亚马逊云科技

版本 1 的用户指南

Amazon Command Line Interface



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Command Line Interface: 版本 1 的用户指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Amazon Web Services 文档中描述的 Amazon Web Services 服务或功能可能因区域而异。要查看适用于中国区域的差异，请参阅 中国的 Amazon Web Services 服务入门 (PDF)。

Table of Contents

..... xiv
关于 Amazon CLI 1
关于 Amazon CLI 版本 1 1
SDK 主要版本的维护和支持 2
关于 Amazon Web Services 2
关于示例 2
其他文档和资源 3
Amazon CLI 文档和资源 3
其他 Amazon SDK 和工具 4
安装 Amazon CLI 5
Python 版本要求 5
Amazon Linux 6
先决条件 6
pip 7
yum 8
Amazon CLI 安装和卸载错误故障排除 8
Linux 9
先决条件 9
使用捆绑安装程序安装和卸载 10
使用 pip 安装和卸载 15
使用 Snapcraft 进行安装和卸载 17
将 Amazon CLI 版本 1 可执行文件添加到命令行路径 19
Amazon CLI 安装和卸载错误故障排除 20
macOS 20
先决条件 21
使用捆绑安装程序安装和卸载 21
使用 pip 进行安装和更新 26
Amazon CLI 安装和卸载错误故障排除 30
Windows 30
使用 MSI 安装程序进行安装、更新和卸载 30
使用 Python 和 pip 进行安装、更新和卸载 32
将 Amazon CLI 可执行文件添加到命令行路径 33
Amazon CLI 安装和卸载错误故障排除 35
Virtualenv 35

先决条件 35
在虚拟环境中进行安装和更新 36
Amazon CLI 安装和卸载错误故障排除 37
配置 Amazon CLI 38
配置和凭证优先顺序 38
此部分中的其他主题 39
配置设置 39
配置和凭证文件的格式 40
配置设置存储在何处？ 45
使用命名配置文件 45
使用命令设置和查看配置设置 46
设置新的配置和凭证命令示例 47
支持的 config 文件设置 50
环境变量 65
如何设置环境变量 65
Amazon CLI 支持的环境变量 67
命令行选项 76
如何使用命令行选项 76
Amazon CLI 支持的全局命令行选项 76
命令行选项的常见用法 79
命令完成 80
工作原理 80
在 Linux 或 macOS 上配置命令完成 81
在 Windows 上配置命令完成 84
重试 85
可用重试模式 86
配置重试模式 88
查看重试日志 89
HTTP 代理 90
使用示例 90
向代理进行身份验证 91
对 Amazon EC2 实例使用代理 92
故障排除 92
了解如何查看、监控和管理 SageMaker 端点。 92
为单个命令设置端点 93
为所有 Amazon Web Services 服务设置全局端点 93

设置成为所有 Amazon Web Services 服务使用 FIPS 端点 94
设置成为所有 Amazon Web Services 服务使用双堆栈端点 95
设置特定于服务的端点 96
基于账户的端点 100
端点配置和设置优先级 102
身份验证和访问凭证 103
配置和凭证优先顺序 103
此部分中的其他主题 104
短期凭证 104
IAM 角色 105
先决条件 106
IAM 角色的用法概览 106
配置和使用角色 107
使用 MFA 109
跨账户角色和外部 ID 111
指定角色会话名称以便于审核 111
通过 Web 身份代入角色 112
清除缓存的凭证 113
IAM 用户 113
步骤 1：创建您的 IAM 用户 114
步骤 2：获取您的访问密钥 114
步骤 3：配置 Amazon CLI 114
Amazon EC2 元数据 116
先决条件 116
配置 Amazon EC2 元数据的配置文件 116
外部凭证 117
使用 Amazon CLI 120
获取帮助 120
内置 Amazon CLI 帮助命令 121
Amazon CLI 参考指南 125
API 文档 126
错误故障排除 126
其他帮助 126
命令结构 126
命令结构 127
Wait 命令 128

指定参数值 129
通用参数类型 130
含字符串的引号 134
文件参数 138
生成 CLI 骨架模板 142
速记语法 149
控制命令输出 151
敏感输出 152
服务器端与客户端输出选项 152
输出格式 153
分页 159
筛选输出 162
返回代码 185
别名 186
先决条件 186
步骤 1：创建别名文件 187
步骤 2：创建别名 188
步骤 3：调用别名 191
别名存储库示例 193
资源 194
排查错误 194
首先尝试的一般故障排除 195
找不到命令错误 203
“aws --version”命令返回的版本与您安装的版本不同 205
卸载 Amazon CLI 后，“aws --version”命令返回一个版本 206
Amazon CLI 处理了参数名称不完整的命令 207
访问被拒绝错误 208
凭证无效和密钥错误 209
签名与错误不匹配 211
找不到 Windows 控制台错误 212
SSL 证书错误 212
JSON 无效错误 214
其他资源 215
Amazon CLI 示例 216
引导式命令示例 216
DynamoDB 217

Amazon EC2 220
Amazon Glacier 238
IAM 244
Amazon S3 248
Amazon SNS 266
命令示例 268
ACM 275
API Gateway 286
API Gateway HTTP 和 WebSocket API 351
API Gateway Management API 401
App Mesh 403
App Runner 447
Amazon AppConfig 481
Application Auto Scaling 514
Application Discovery Service 531
Application Signals 537
AppRegistry 558
Athena 569
Auto Scaling 601
Auto Scaling Plans 667
Amazon Backup 675
Amazon Batch 680
Amazon Budgets 695
Amazon Chime 706
Cloud Control API 749
Amazon Cloud Map 755
Amazon Cloud9 791
Amazon CloudFormation 799
CloudFront 860
Amazon CloudSearch 964
CloudTrail 965
CloudWatch 982
CloudWatch Logs 1016
CloudWatch 网络监控 1022
CloudWatch Observability Access Monitor 1034
CloudWatch Observability Admin 1045

CloudWatch Synthetics 1050
CodeArtifact 1068
CodeBuild 1095
CodeCommit 1157
CodeDeploy 1229
CodeGuru Reviewer 1268
CodePipeline 1286
AWS CodeStar Notifications 1317
CodeConnections 1328
Amazon Cognito Identity 1336
Amazon Cognito 身份提供者 1341
Amazon Comprehend 1486
Amazon Comprehend Medical 1619
Amazon Config 1654
Amazon Connect 1677
Amazon 成本和使用情况报告 1693
Cost Explorer Service 1695
Firehose 1703
Amazon Data Lifecycle Manager 1706
Amazon Data Pipeline 1712
DataSync 1722
DAX 1726
Detective 1744
Device Farm 1755
Amazon Direct Connect 1760
Amazon Directory Service 1809
Amazon Directory Service 数据 1812
Amazon DMS 1837
Amazon DocumentDB 1880
DynamoDB 1936
DynamoDB Streams 2029
Amazon EC2 2036
Amazon EC2 Instance Connect 2690
Amazon ECR 2691
Amazon ECR Public 2721
Amazon ECS 2749

Amazon EFS 2872
Amazon EKS 2880
Elastic Beanstalk 2976
ELB – 版本 1 3005
ELB – 版本 2 3032
ElastiCache 3085
MediaStore 3189
Amazon EMR 3205
Amazon EMR on EKS 3253
EventBridge 3254
EventBridge Pipes 3260
Firewall Manager 3268
Amazon FIS 3278
Amazon GameLift Servers 3296
Amazon Glacier 3328
Global Accelerator 3350
Amazon Glue 3388
GuardDuty 3409
Amazon Health 3428
HealthImaging 3435
HealthLake 3462
HealthOmics 3475
IAM 3541
IAM Access Analyzer 3676
Image Builder 3711
Incident Manager 3752
Incident Manager Contacts 3773
Amazon Inspector 3796
Amazon IoT 3843
Amazon IoT Analytics 4017
Device Advisor 4044
Amazon IoT data 4058
Amazon IoT Events 4061
Amazon IoT Events-Data 4085
Amazon IoT Greengrass 4110
Amazon IoT Greengrass V2 4193

Amazon IoT Jobs SDK 发布 4218
Amazon IoT SiteWise 4221
Amazon IoT Things Graph 4269
Amazon IoT Wireless 4295
Amazon IVS 4331
Amazon IVS 聊天功能 4373
Amazon IVS 实时直播功能 4386
Amazon Kendra 4438
Kinesis 4447
Amazon KMS 4465
Lake Formation 4532
Lambda 4583
License Manager 4624
Lightsail 4636
Macie 4760
Amazon Managed Grafana 4765
MediaConnect 4767
MediaConvert 4782
MediaLive 4807
MediaPackage 4812
MediaPackage VOD 4827
MediaStore Data Plane 4839
MediaTailor 4844
MemoryDB 4849
Amazon MSK 4885
网络流量监测仪 4894
Network Manager 4910
OpenSearch Service 4946
Organizations 4960
Amazon Outposts 4997
Amazon Payment Cryptography 5001
Amazon Payment Cryptography 数据面板 5021
Amazon Pinpoint 5030
Amazon Polly 5052
Amazon 价目表 5058
Amazon 私有 CA 5063

Amazon Proton 5070
Amazon RDS 5082
Amazon RDS 数据服务 5274
Amazon RDS 性能详情 5278
Amazon Redshift 5292
Amazon Rekognition 5369
Amazon RAM 5444
资源探索器 5467
资源组 5489
资源组标记 API 5502
Route 53 5505
Route 53 域注册 5519
Route 53 配置文件 5544
Route 53 Resolver 5556
Amazon S3 5599
Amazon S3 控件 5688
Secrets Manager 5703
Security Hub CSPM 5731
Security Lake 5806
Amazon Serverless Application Repository 5840
Service Catalog 5842
服务配额 5872
Amazon SES 5882
盾牌 5894
签署人 5910
Snowball Edge 5920
Amazon SNS 5921
Amazon SQS 5942
Storage Gateway 5962
Amazon STS 5965
Amazon Web Services 支持 5975
Amazon SWF 5987
Systems Manager 6003
Amazon Textract 6174
Amazon Transcribe 6185
Amazon Translate 6226

Trusted Advisor 6227
Verified Permissions 6247
VPC Lattice 6270
Amazon WAF Classic 6305
Amazon WAF Classic Regional 6310
Amazon WAFV2 6316
WorkDocs 6360
Amazon WorkMail 6392
Amazon WorkMail Message Flow 6415
WorkSpaces 6417
X-Ray 6432
Bash 脚本示例 6449
Amazon Batch 6450
Amazon Cloud Map 6461
CloudFront 6484
DynamoDB 6492
Amazon EC2 6591
HealthImaging 6747
IAM 6756
Amazon KMS 6820
Lightsail 6822
Amazon S3 6830
Amazon STS 6862
安全性 6865
数据保护 6865
数据加密 6866
身份和访问管理 6867
受众 6867
使用身份进行身份验证 6867
使用策略管理访问 6868
Amazon Web Services 服务如何与 IAM 协同工作 6870
对 Amazon 身份和访问进行问题排查 6870
合规性验证 6872
韧性 6872
基础设施安全性 6872
强制实施最低 TLS 版本 6873

确定当前支持的协议 6873
编译 OpenSSL 和 Python 6875
文档历史记录 6877

本文档仅适用于 Amazon CLI 版本 1。

我们已宣布即将终止对 Amazon CLI 版本 1 的支持。建议您迁移到 Amazon CLI 版本 2。有关日期、其它详细信息以及如何迁移的信息，请参阅公告。有关 Amazon CLI 版本 2 的相关文档，请参阅版本 2 用户指南。

什么是 Amazon Command Line Interface 版本 1？

> **Note**
>
> Amazon CLI 版本 1 不是 Amazon CLI 的最新版本。Amazon CLI 版本 2 中引入的某些功能无法向后兼容版本 1，您必须升级才能访问这些功能。版本 1 中有一些可能需要您更改脚本的“重大”更改。有关版本 2 中的重大更改的列表，请参阅 Amazon CLI 版本 2 用户指南 中的重大更改。

Amazon Command Line Interface (Amazon CLI) 是一种开源工具，让您能够在命令行 Shell 中使用命令与 Amazon 服务进行交互。仅需最少的配置，即可使用 Amazon CLI 开始运行命令，以便从终端程序中的命令提示符实现与基于浏览器的 Amazon Web Services 管理控制台所提供的功能等同的功能：

- Linux Shell – 使用常见 Shell 程序（例如 `bash`、`zsh` 和 `tcsh`）在 Linux 或 macOS 中运行命令。
- Windows 命令行 – 在 Windows 上，在 Windows 命令提示符处或在 PowerShell 中运行命令。
- 远程 – 通过远程终端程序（如 PuTTY 或 SSH）在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上运行命令，或者使用 Amazon Systems Manager 运行命令。

Amazon Web Services 管理控制台 中的所有 IaaS（基础设施即服务）Amazon管理和访问函数均可以在 Amazon API 和 Amazon CLI 中获取。新的 Amazon IaaS 功能和服务在启动时或在 180 天启动期内通过 API 和 CLI 提供全部 Amazon Web Services 管理控制台 功能。

Amazon CLI 提供对 Amazon 服务的公共 API 的直接访问。您可以使用 Amazon CLI 探索服务的功能，可以开发 Shell 脚本来管理资源。除了低级别的 API 等效命令，多项 Amazon 服务还为 Amazon CLI 提供了自定义项。自定义项可能包括更高级别的命令，可简化具有复杂 API 的服务的使用。

关于 Amazon CLI 版本 1

Amazon CLI 版本 1 是 Amazon CLI 的原始版，我们将继续支持该版本。但是，Amazon CLI 版本 2 中引入的主要新功能可能不会向后移植到 Amazon CLI 版本 1 中。要使用这些功能，您必须安装 Amazon CLI 版本 2。Amazon CLI 版本 1 使用 SDK for Python 构建，因此需要安装 Python 的兼容版本。

要安装 Amazon CLI 版本 1，请参阅 安装 Amazon CLI。

要检查当前安装的版本，请使用以下命令：

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

有关版本历史记录，请参阅 GitHub 上的 Amazon CLI 版本 1 更改日志。

SDK 主要版本的维护和支持

有关维护和支持 SDK 主要版本及其基础依赖关系的信息，请参阅 Amazon SDK 和工具参考指南中的以下内容：

- Amazon SDK 和工具维护策略
- Amazon SDK 和工具版本支持矩阵

关于 Amazon Web Services

Amazon Web Services (Amazon) 是数字基础设施服务的集合，开发人员可在开发应用程序时加以利用。这些服务包括计算、存储、数据库、分析和应用程序同步（消息收发和排队）。Amazon使用即用即付服务模式。您只需为您或您的应用程序使用的服务付费。此外，Amazon还提供免费使用套餐，以便让Amazon作为原型制作和实验平台更易实现。在此套餐中，低于某种使用水平的服务是免费的。有关 Amazon 成本和免费套餐的更多信息，请参阅 Amazon 免费套餐。要获取 Amazon 账户，请访问 Amazon 主页并选择 Create an Amazon Account（创建 Amazon 账户）。

关于《Amazon CLI 用户指南》中的示例

本指南中 Amazon Command Line Interface (Amazon CLI) 示例的格式是使用下列约定进行设置的：

- 提示 – 命令提示符使用 Linux 提示符并显示为 ($)。对于 Windows 特定的命令，C:\> 用作提示。请勿在键入命令时包含提示符。
- 目录 – 当必须从特定目录执行命令时，目录名称将显示在提示符符号之前。
- 用户输入 – 您在命令行处输入的命令文本采用 **`user input`** 格式。
- 可替换文本 – 变量文本（包括您选择的资源的名称，或您必须包含在命令中的由Amazon服务生成的 ID）采用的格式为*#####*。在多行命令中或需要特定键盘输入的命令中，键盘命令也可显示为可替换文本。
- 输出 – Amazon 服务返回的输出显示在用户输入下方，采用 `computer output` 格式。

例如，以下 **`aws configure`** 命令示例显示了用户输入、可替换文本和输出：

1. 在命令行输入 **aws configure**，然后按 Enter 键。
2. Amazon CLI 输出文本行，提示您输入其他信息。
3. 依次输入每个访问密钥，然后按 Enter。
4. 然后，以显示的格式输入 Amazon 区域名称，按 Enter，然后最后一次按 Enter 以跳过输出格式设置。
5. 最终 Enter 命令将显示为可替换文本，因为这一行没有用户输入。

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: ENTER
```

以下示例显示带输出的简单命令。要使用此示例，请输入命令的完整文本（提示符后突出显示的文本），然后按 Enter。安全组的名称 *my-sg* 可替换为所需的安全组名称。JSON 文档（包括大括号）是输出。如果您将 CLI 配置为以文本或表格式进行输出，输出的格式将有差异。JSON 是默认输出格式。

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group"
{
    "GroupId": "sg-903004f8"
}
```

有关 Amazon CLI 的其他文档和资源

Amazon CLI 文档和资源

除了本用户指南外，在您使用 Amazon CLI 时还提供以下有价值的在线资源。

- Amazon CLI 版本 1 参考指南
- Amazon CLI Bash 脚本代码示例存储库。开源 bash 脚本示例。Bash 脚本示例托管在 GitHub 上的 Amazon 代码示例存储库中。
- Amazon CLI GitHub 存储库 您可以在 GitHub 上查看和分流 Amazon CLI 的源代码。加入 GitHub 上的用户社区，提供反馈、请求功能和提交自己的文章。这包括查看和提供有关 Amazon CLI 文档的命令示例。
- Amazon CLI 别名示例存储库您可以在 GitHub 上查看和分流 Amazon CLI 别名示例。

- Amazon CLI 版本 1 更新日志
- Amazon CLI 版本 2 更新日志

其他 Amazon SDK 和工具

根据您的使用案例，您可能希望选择符合您需求的某个 Amazon SDK 和工具：

- Amazon SDK 和工具参考指南
- 适用于 C++ 的 Amazon SDK
- 适用于 Go 的 Amazon SDK
- 适用于 Java 的 Amazon SDK
- 适用于 JavaScript 的 Amazon SDK
- 适用于 Kotlin 的 Amazon SDK
- 适用于 .NET 的 Amazon SDK
- Amazon SDK for Python (Boto)
- 适用于 PHP 的 Amazon SDK
- Amazon Tools for PowerShell
- 适用于 Ruby 的 Amazon SDK
- 适用于 Rust 的 Amazon SDK
- 适用于 SAP ABAP 的 Amazon SDK
- 适用于 Swift 的 Amazon SDK
- Amazon Amplify

安装、更新和卸载 Amazon CLI

本主题提供用于安装、更新和卸载 Amazon Command Line Interface (Amazon CLI) 原始版本的链接。当前支持 Amazon CLI 版本 1，但添加到 Amazon CLI 版本 2 的新功能可能未添加到 Amazon CLI 版本 1。要使用这些功能，您必须安装 Amazon CLI 版本 2。有关如何安装版本 2 的信息，请参阅安装 Amazon CLI 版本 2。

Amazon CLI 安装、更新和卸载说明：

- Python 版本要求
- 在 Linux 上安装、更新和卸载 Amazon CLI 版本 1
- 在 Linux 上安装、更新和卸载 Amazon CLI 版本 1
- 在 macOS 上安装、更新和卸载 Amazon CLI 版本 1
- 在 Windows 上安装、更新和卸载 Amazon CLI 版本 1
- 在虚拟环境中安装和更新 Amazon CLI 版本 1

Python 版本要求

Amazon CLI 版本 1 使用 SDK for Python 构建，因此需要安装 Python 的兼容版本。

Python 版本支持矩阵

Amazon CLI version	支持的 Python 版本
1.39.0 – 当前	Python 3.9+
1.32.0 – 1.38.38	Python 3.8+
1.27.0 – 1.31.x	Python 3.7+
1.20.0 – 1.26.x	Python 3.6+
1.19.0 — 1.19.x	Python 2.7+、Python 3.6+
1.17 – 1.18.x	Python 2.7+、Python 3.4+
1.0 – 1.16.x	Python 2.6 及更早版本，Python 3.3 及更早版本

有关 Amazon CLI 最新版本的信息，请参阅 GitHub 上的 Amazon CLI 版本 2 更改日志。

在 Linux 上安装、更新和卸载 Amazon CLI 版本 1

Amazon CLI 版本 1 已预安装在 Amazon Linux 和 Amazon Linux 2 上。使用以下命令检查当前安装的版本。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

根据您创建 Amazon Linux 实例的时间，使用下列程序包管理器之一来预安装 Amazon CLI 版本 1：

- pip
- yum

先决条件

您必须已安装 Python 3.8 或更高版本。有关安装说明，请参阅 Python 的初学者指南 中的下载 Python 页面。

Python 版本支持矩阵

Amazon CLI version	支持的 Python 版本
1.39.0 – 当前	Python 3.9+
1.32.0 – 1.38.38	Python 3.8+
1.27.0 – 1.31.x	Python 3.7+
1.20.0 – 1.26.x	Python 3.6+
1.19.0 — 1.19.x	Python 2.7+、Python 3.6+
1.17 – 1.18.x	Python 2.7+、Python 3.4+
1.0 – 1.16.x	Python 2.6 及更早版本，Python 3.3 及更早版本

使用 pip 进行安装、更新和卸载

大多数 Amazon Linux 实例都使用 pip 来预安装 Amazon CLI 版本 1。

使用 pip 在 Amazon Linux 上安装或更新 Amazon CLI 版本 1

要为当前用户安装最新版本的 Amazon CLI 版本 1，请按以下说明操作。

1. 如果您安装了 Python 3 或更高版本，我们建议您使用 pip3。使用 `pip3 install` 安装或更新至 Amazon CLI 版本 1 的最新版本。如果您在 Python 虚拟环境 (venv) 中运行命令，则不需要使用 `--user` 选项。

   ```
   $ pip3 install --upgrade --user awscli
   ```

2. 确保包含 `aws` 的文件夹是您的 `PATH` 变量的一部分。

   a. 在您的用户目录中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 `echo $SHELL`。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash – `.bash_profile`、`.profile` 或 `.bash_login`
   - Zsh – `.zshrc`
   - Tcsh – `.tcshrc`、`.cshrc` 或 `.login`

   b. 在配置文件脚本末尾添加与以下示例类似的导出命令。

   ```
   export PATH=$HOME/.local/bin:$PATH
   ```

   此命令将路径（在本示例中为 `$HOME/.local/bin`）插入到现有 `$PATH` 变量的前面。

   c. 将配置文件重新加载到当前会话中，以使更改生效。

   ```
   $ source ~/.bash_profile
   ```

3. 要验证是否正在运行新版本，请使用 `aws --version` 命令。

   ```
   $ aws --version
   aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
   ```

使用 pip 卸载 Amazon CLI 版本 1

如果需要卸载 Amazon CLI，请使用 `pip uninstall`。

```
$ pip3 uninstall awscli
```

使用 yum 进行安装、更新和卸载

大多数 Amazon Linux 2 实例都使用 yum 来预安装 Amazon CLI 版本 1。

使用 yum 在 Amazon Linux 上安装或更新 Amazon CLI 版本 1

要安装 Amazon Linux 上提供的 Amazon CLI 版本 1 的最新版本，请运行以下命令。

```
$ sudo yum install awscli
```

要更新到 Amazon Linux 上提供的 Amazon CLI 版本 1 的最新版本，请运行以下命令。

```
$ sudo yum update awscli
```

要验证是否正在运行更高的版本，请使用 `aws --version` 命令。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

使用 yum 卸载 Amazon CLI 版本 1

要卸载 Amazon CLI，请使用 `yum remove`。

```
$ sudo yum remove awscli
```

Amazon CLI 安装和卸载错误故障排除

如果您在安装或卸载 Amazon CLI 后遇到问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “找不到命令错误”、the section called ““aws --version”命令返回的版本与您安装的版本不同”和the section called “卸载 Amazon CLI 后，“aws --version”命令返回一个版本”。

在 Linux 上安装、更新和卸载 Amazon CLI 版本 1

您可以使用 Amazon Command Line Interface 程序包管理器或捆绑安装程序安装 Amazon CLI (pip) 版本 1 及其对大多数 Linux 发行版的依赖项。

尽管其他程序包管理器（如 awscli 和 apt）的存储库中提供了 yum 程序包，但它们不是由 Amazon 生产、管理和支持的。我们建议您仅从本指南中记载的官方 Amazon CLI 发行点安装 Amazon。

Sections

- 先决条件
- 在 Linux 上使用捆绑安装程序安装和卸载 Amazon CLI 版本 1
- 使用 pip 安装和卸载 Amazon CLI 版本 1
- 使用 Snapcraft 安装和卸载 Amazon CLI 版本 1
- 将 Amazon CLI 版本 1 可执行文件添加到命令行路径
- Amazon CLI 安装和卸载错误故障排除

先决条件

您必须已安装 Python 3.8 或更高版本。有关安装说明，请参阅 Python 的初学者指南 中的下载 Python 页面。

Python 版本支持矩阵

Amazon CLI version	支持的 Python 版本
1.39.0 – 当前	Python 3.9+
1.32.0 – 1.38.38	Python 3.8+
1.27.0 – 1.31.x	Python 3.7+
1.20.0 – 1.26.x	Python 3.6+
1.19.0 — 1.19.x	Python 2.7+、Python 3.6+
1.17 – 1.18.x	Python 2.7+、Python 3.4+
1.0 – 1.16.x	Python 2.6 及更早版本，Python 3.3 及更早版本

在 Linux 上使用捆绑安装程序安装和卸载 Amazon CLI 版本 1

在 Linux 或 macOS 上，可以使用捆绑安装程序来安装 Amazon CLI 的版本 1。捆绑安装程序包含所有依赖项，并可以离线使用。

> **Note**
>
> 捆绑安装程序不支持安装到包含空格的路径。

主题

- 使用捆绑安装程序（带有 sudo）安装 Amazon CLI 版本 1
- 使用捆绑安装程序（不带有 sudo）安装 Amazon CLI 版本 1
- 卸载 Amazon CLI 版本 1 捆绑安装程序

使用捆绑安装程序（带有 **sudo**）安装 Amazon CLI 版本 1

以下步骤使您能够从任何版本的 Linux 或 macOS 上的命令行安装 Amazon CLI 版本 1。

以下是可剪切和粘贴以作为一组命令运行的安装命令的摘要，各个命令的具体解释见下文。

对于最新版本的 Amazon CLI，请使用以下命令块：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
unzip awscli-bundle.zip
sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```

对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 `awscli-bundle-1.16.312.zip`，这会生成以下命令：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
bundle.zip"
unzip awscli-bundle.zip
sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```

使用捆绑安装程序，在命令行中执行以下步骤来安装 Amazon CLI 版本 1。

要使用捆绑安装程序安装 Amazon CLI 版本 1

1. 使用以下方法之一下载 Amazon CLI 版本 1 捆绑安装程序。

- 使用 `curl` 命令下载。

  对于最新版本的 Amazon CLI，请使用以下命令块：

  ```
  $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
  ```

  对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 `awscli-bundle-1.16.312.zip`，这会生成以下命令：

  ```
  $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-bundle.zip"
  ```

- 使用直接链接下载。

  对于最新版本的 Amazon CLI：https://s3.amazonaws.com/aws-cli/awscli-bundle.zip

  对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 `awscli-bundle-1.16.312.zip`，这会生成以下 URL：https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip

2. 从程序包中提取文件。如果没有 `unzip` 来提取文件，请使用 Linux 发行版的内置程序包管理器进行安装。

   ```
   $ unzip awscli-bundle.zip
   ```

3. 运行安装程序。安装程序在 Amazon CLI 中安装 `/usr/local/aws`，并在 `aws` 目录中创建符号链接 `/usr/local/bin`。使用 `-b` 选项创建符号链接将免除在用户的 `$PATH` 变量中指定安装目录的需要。这应该能让所有用户通过在任何目录下输入 Amazon CLI 来调用 `aws`。

   ```
   $ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
   ```

   默认情况下，安装脚本在系统默认版本的 Python 下运行。如果已安装 Python 的备选版本并希望使用该版本安装 Amazon CLI，请使用该版本按 Python 可执行文件的绝对路径运行安装脚本，如下所示。

   ```
   $ sudo /usr/local/bin/python3.7 awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
   ```

4. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

使用捆绑安装程序（不带有 **sudo**）安装 Amazon CLI 版本 1

如果您没有 sudo 权限，或打算仅为当前用户安装 Amazon CLI，则可使用先前命令的修改版本。前两个命令是相同的。

对于最新版本的 Amazon CLI，请使用以下命令块：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
unzip awscli-bundle.zip
./awscli-bundle/install -b ~/bin/aws
```

对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下命令：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
bundle.zip"
unzip awscli-bundle.zip
./awscli-bundle/install -b ~/bin/aws
```

要为当前用户安装 Amazon CLI 版本 1

1. 使用以下方式之一下载 Amazon CLI 版本 1 捆绑安装程序。

   - 使用 curl 命令下载。

     对于最新版本的 Amazon CLI，请使用以下命令块：

     ```
     $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-
     bundle.zip"
     ```

     对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下命令：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
bundle.zip"
```

- 使用直接链接下载。

  对于最新版本的 Amazon CLI：https://s3.amazonaws.com/aws-cli/awscli-bundle.zip

  对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-*1.16.312*.zip，这会生成以下 URL：https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip

2. 使用 unzip 从程序包中提取文件。如果没有 unzip，请使用 Linux 发行版的内置程序包管理器进行安装。

```
$ unzip awscli-bundle.zip
```

3. 运行安装程序。安装程序在 Amazon CLI 中安装 /usr/local/aws，并在 aws 目录中创建符号链接 /usr/local/bin。此命令使用 -b 参数以指定安装程序放置 aws 符号链接文件的目录。您必须具有对指定文件夹的写入权限。

```
$ ./awscli-bundle/install -b ~/bin/aws
```

这会将 Amazon CLI 安装到默认位置 (~/.local/lib/aws) 并在 ~/bin/aws 中创建符号链接 (symlink)。确保您的 ~/bin 环境变量中包含 PATH，以使该符号链接生效。

```
$ echo $PATH | grep ~/bin     // See if $PATH contains ~/bin (output will be empty
 if it doesn't)
$ export PATH=~/bin:$PATH     // Add ~/bin to $PATH if necessary
```

4. 确保包含 Amazon CLI 版本 1 的目录是您的 PATH 变量的一部分。

   a. 在您的用户文件夹中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 echo $SHELL。

```
$ ls -a ~
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

   - Bash – .bash_profile、.profile 或 .bash_login
   - Zsh – .zshrc

- Tcsh – .tcshrc、.cshrc 或 .login

b. 在配置文件脚本末尾添加与以下示例类似的导出命令。

```
export PATH=~/.local/bin:$PATH
```

此命令将路径（在本示例中为 ~/.local/bin）插入到现有 PATH 变量的前面。

c. 将配置文件重新加载到当前会话中，以使更改生效。

```
$ source ~/.bash_profile
```

5. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

卸载 Amazon CLI 版本 1 捆绑安装程序

1. 如果使用捆绑安装程序安装了 Amazon CLI，请按照以下说明进行操作。除了可选的符号链接之外，捆绑安装程序不会将任何内容放在安装目录之外，所以卸载十分简单，就是直接删除这两个项目。

```
$ sudo rm -rf /usr/local/aws
$ sudo rm -rf /usr/local/bin/aws
```

2. （可选）删除 .aws 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

.aws 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *~/.aws/*。如果您的用户对此目录具有写入权限，则无需使用 sudo。

```
$ sudo rm -r ~/.aws/
```

使用 pip 安装和卸载 Amazon CLI 版本 1

主题

- 安装 pip
- 使用 pip 安装和更新 Amazon CLI 版本 1
- 使用 pip 卸载 Amazon CLI

安装 pip

如果尚未安装 pip，可以使用 Python 打包权威机构 提供的脚本进行安装。运行 pip --version 可查看您的 Linux 版本是否已包含 Python 和 pip。如果您安装了 Python 3 或更高版本，我们建议您使用 pip3 命令。

1. 使用 curl 命令下载安装脚本。以下命令使用 -O（大写字母“O”）参数指定下载的文件将使用与远程主机上相同的名称存储在当前的目录中。

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   ```

2. 使用 python 或 python3 命令运行脚本以下载并安装最新版本的 pip 和其他必需的支持包。当您包含 --user 开关时，脚本将 pip 安装到路径 ~/.local/bin。

   ```
   $ python3 get-pip.py --user
   ```

3. 确保包含 pip 的目录是您的 PATH 变量的一部分。

   a. 在您的用户文件夹中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 echo $SHELL。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash – .bash_profile、.profile 或 .bash_login
   - Zsh – .zshrc
   - Tcsh – .tcshrc、.cshrc 或 .login

b. 在配置文件脚本末尾添加与以下示例类似的导出命令。

```
export PATH=~/.local/bin:$PATH
```

此命令将路径（在本示例中为 ~/.local/bin）插入到现有 PATH 变量的前面。

c. 将配置文件重新加载到当前会话中，以使更改生效。

```
$ source ~/.bash_profile
```

4. 要验证 pip 是 pip3 否已正确安装，请运行以下命令。

```
$ pip3 --version
pip 24.0 from ~/.local/lib/python3.7/site-packages (python 3.7)
```

使用 pip 安装和更新 Amazon CLI 版本 1

1. 使用 pip 或 pip3 命令安装或更新 Amazon CLI。如果您使用的是 Python 3 或更高版本，我们建议您使用 pip3 命令。使用 --user 开关时，pip 将 Amazon CLI 安装到 ~/.local/bin。

   对于最新版本的 Amazon CLI，请使用以下命令块：

   ```
   $ pip3 install awscli --upgrade --user
   ```

   对于特定版本的 Amazon CLI，在文件名后附加两个等号 = 和版本号。在本示例中，版本 *1.16.312* 的文件名为 *==1.16.312*，这会生成以下命令：

   ```
   $ pip3 install awscli==1.16.312 --upgrade --user
   ```

   > **Note**
   >
   > 为您的终端使用适当的引用规则。要使用 = 字符，您可能需要使用单引号或双引号适当地进行转义。以下示例使用单引号进行转义：
   >
   > ```
   > $ pip3 install 'awscli==1.16.312' --upgrade --user
   > ```

2. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

使用 pip 卸载 Amazon CLI

1. 如果您使用 pip 安装了 Amazon CLI 版本 1，则还必须使用 pip 进行卸载。

   ```
   $ pip uninstall awscli
   ```

   如果您使用的是 Python 版本 2 或 3，则可能需要使用 pip2 或 pip3 命令。使用 aws --version 命令确定与您安装的 Amazon CLI 版本 1 关联的 Python 版本。

   ```
   $ pip3 uninstall awscli
   ```

   您可能需要重新启动命令提示符窗口或电脑才能删除所有文件。
2. （可选）删除 .aws 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

   > **Warning**
   >
   > 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

   .aws 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *~/.aws/*。如果您的用户对此目录具有写入权限，则无需使用 sudo。

   ```
   $ sudo rm -r ~/.aws/
   ```

使用 Snapcraft 安装和卸载 Amazon CLI 版本 1

主题

- 安装 snap
- 使用 snap 安装和更新 Amazon CLI 版本 1

- 使用 snap 卸载 Amazon CLI

安装 snap

如果尚未安装 snap，可以使用 Canonical Snapcraft 提供的说明进行安装。运行 snap version 可查看您的 Linux 版本是否已包含 snap。

1. 在您的平台上安装 Snapcraft。有关安装 Snapcraft 的信息，请参阅《Snap 文档》中的安装进程守护程序。
2. 重新启动系统，以便正确更新 PATH 变量。如果您遇到安装问题，请按照《Snap 文档》中修复常见问题中的步骤操作。
3. 要验证 snap 是否已正确安装，请运行以下命令：

```
$ snap version
```

使用 snap 安装和更新 Amazon CLI 版本 1

1. 为 Amazon CLI 版本 1 运行以下 snap install 命令

```
$ snap install aws-cli --channel=v1/stable --classic
```

根据您的权限，您可能需要将 sudo 添加到命令中。

```
$ sudo snap install aws-cli --channel=v1/stable --classic
```

2. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

使用 snap 卸载 Amazon CLI

1. 如果您使用 snap 安装了 Amazon CLI 版本 1，则还必须使用 snap 进行卸载。

```
$ snap remove aws-cli
```

您可能需要重新启动命令提示符窗口或电脑才能删除所有文件。

2. （可选）删除 .aws 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

.aws 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *~/.aws/*。如果您对此目录具有写入权限，则无需使用 sudo。

```
$ sudo rm -r ~/.aws/
```

将 Amazon CLI 版本 1 可执行文件添加到命令行路径

在使用 pip 或 snap 进行安装后，可能需要将 aws 可执行文件添加到操作系统的 PATH 环境变量中。

您可以运行以下命令验证 pip 已将 Amazon CLI 安装到哪个文件夹中。

```
$ which aws
/home/username/.local/bin/aws
```

您可以将此路径 ~/.local/bin/ 作为参考，因为在 Linux 中 /home/username 对应于 ~。

如果您忽略了 --user 开关且未在用户模式下安装，可执行文件可能位于 Python 安装的 bin 文件夹中。如果您不知道 Python 的安装位置，请运行此命令。

```
$ which python
/usr/local/bin/python
```

输出可能是符号链接的路径，而不是实际的可执行文件。运行 ls -al 以查看所指向的路径。

```
$ ls -al /usr/local/bin/python
/usr/local/bin/python -> ~/.local/Python/3.6/bin/python3.6
```

pip 将程序安装到 Python 应用程序所在的文件夹中。将此文件夹添加到 PATH 变量。

修改您的 **PATH** 变量

1. 在您的用户目录中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 echo $SHELL。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash – .bash_profile、.profile 或 .bash_login
   - Zsh – .zshrc
   - Tcsh – .tcshrc、.cshrc 或 .login
2. 向配置文件脚本中添加导出命令。

   ```
   export PATH=~/.local/bin:$PATH
   ```

   在本示例中，此命令将路径 ~/.local/bin 添加到当前 PATH 变量中。
3. 将更新的配置文件加载到当前会话中。

   ```
   $ source ~/.bash_profile
   ```

Amazon CLI 安装和卸载错误故障排除

如果您在安装或卸载 Amazon CLI 后遇到问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “找不到命令错误”、the section called ““aws --version”命令返回的版本与您安装的版本不同”和the section called “卸载 Amazon CLI 后，“aws --version”命令返回一个版本”。

在 macOS 上安装、更新和卸载 Amazon CLI 版本 1

您可以使用捆绑安装程序或 pip 安装 Amazon Command Line Interface (Amazon CLI) 版本 1 及其对 macOS 的依赖项。

小节目录

- 先决条件

- 使用捆绑安装程序在 macOS 上安装、更新和卸载 Amazon CLI 版本 1
- 使用 pip 安装、更新和卸载 Amazon CLI 版本 1
- Amazon CLI 安装和卸载错误故障排除

先决条件

必须先确保已安装 Python 3.8 或更高版本，然后才能在 macOS 上安装 Amazon CLI 版本 1。有关安装说明，请参阅 Python 的初学者指南 中的下载 Python 页面。

Python 版本支持矩阵

Amazon CLI version	支持的 Python 版本
1.39.0 – 当前	Python 3.9+
1.32.0 – 1.38.38	Python 3.8+
1.27.0 – 1.31.x	Python 3.7+
1.20.0 – 1.26.x	Python 3.6+
1.19.0 — 1.19.x	Python 2.7+、Python 3.6+
1.17 – 1.18.x	Python 2.7+、Python 3.4+
1.0 – 1.16.x	Python 2.6 及更早版本，Python 3.3 及更早版本

使用捆绑安装程序在 macOS 上安装、更新和卸载 Amazon CLI 版本 1

在 Linux 或 macOS 上，可以使用捆绑安装程序来安装 Amazon Command Line Interface (Amazon CLI) 的版本 1。捆绑安装程序包含所有依赖项，并可以离线使用。

捆绑安装程序不支持安装到包含空格的路径。

主题

- 使用捆绑安装程序（带有 sudo）安装 Amazon CLI 版本 1
- 使用捆绑安装程序（不带有 sudo）安装 Amazon CLI 版本 1

- 卸载 Amazon CLI 版本 1 捆绑安装程序

使用捆绑安装程序（带有 **sudo**）安装 Amazon CLI 版本 1

以下步骤使您能够从任何版本的 macOS 上的命令行安装 Amazon CLI 版本 1。

以下是可剪切和粘贴以作为一组命令运行的安装命令的摘要。

对于最新版本的 Amazon CLI，请使用以下命令块：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
unzip awscli-bundle.zip
sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```

对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 `awscli-bundle-1.16.312.zip`，这会生成以下命令：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
bundle.zip"
unzip awscli-bundle.zip
sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```

要使用捆绑安装程序安装 Amazon CLI 版本 1

1. 使用以下方式之一下载 Amazon CLI 版本 1 捆绑安装程序：

   - 使用 `curl` 命令下载。

     对于最新版本的 Amazon CLI，请使用以下命令块：

     ```
     $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-
     bundle.zip"
     ```

     对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 `awscli-bundle-1.16.312.zip`，这会生成以下命令：

     ```
     $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
     bundle.zip"
     ```

   - 使用直接链接下载。

对于最新版本的 Amazon CLI：https://s3.amazonaws.com/aws-cli/awscli-bundle.zip

对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下 URL：https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip

2. 从程序包中提取（解压缩）文件。如果没有 unzip，请使用 macOs 发行版的内置程序包管理器进行安装。

```
$ unzip awscli-bundle.zip
```

3. 运行安装程序。安装程序在 Amazon CLI 中安装 /usr/local/aws，并在 aws 文件夹中创建符号链接 /usr/local/bin。使用 -b 选项创建符号链接将免除在用户的 $PATH 变量中指定安装文件夹的需要。这应该能让所有用户通过在任何目录下输入 aws 来调用 Amazon CLI。

```
$ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```

默认情况下，安装脚本在系统默认版本的 Python 下运行。如果已安装 Python 的可选版本并希望使用该版本安装 Amazon CLI，请使用该版本按 Python 可执行文件的绝对路径运行安装脚本，如下所示。

```
$ sudo /usr/local/bin/python3.7 awscli-bundle/install -i /usr/local/aws -b /usr/
local/bin/aws
```

4. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

使用捆绑安装程序（不带有 **sudo**）安装 Amazon CLI 版本 1

如果您没有 sudo 权限，或打算仅为当前用户安装 Amazon CLI，则可使用先前命令的修改版本。前两个命令是相同的。

对于最新版本的 Amazon CLI，请使用以下命令块：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
```

```
unzip awscli-bundle.zip
./awscli-bundle/install -b ~/bin/aws
```

对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下命令：

```
$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
bundle.zip"
unzip awscli-bundle.zip
./awscli-bundle/install -b ~/bin/aws
```

为当前用户安装 Amazon CLI 版本 1

1. 使用以下方法之一下载 Amazon CLI 版本 1 捆绑安装程序：

   - 使用 curl 命令下载。

     对于最新版本的 Amazon CLI，请使用以下命令块：

     ```
     $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-
     bundle.zip"
     ```

     对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下命令：

     ```
     $ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip" -o "awscli-
     bundle.zip"
     ```

   - 使用直接链接下载。

     对于最新版本的 Amazon CLI：https://s3.amazonaws.com/aws-cli/awscli-bundle.zip

     对于特定版本的 Amazon CLI，在文件名后附加一个连字符和版本号。在本示例中，版本 *1.16.312* 的文件名为 awscli-bundle-1.16.312.zip，这会生成以下 URL：https://s3.amazonaws.com/aws-cli/awscli-bundle-1.16.312.zip

2. 从程序包中提取文件。如果没有 unzip，请使用 Linux 发行版的内置程序包管理器进行安装。

   ```
   $ unzip awscli-bundle.zip
   ```

3. 运行安装程序。安装程序在 Amazon CLI 中安装 /usr/local/aws，并在 aws 目录中创建符号链接 /usr/local/bin。此命令使用 -b 参数以指定安装程序放置 aws 符号链接文件的目录。您必须具有对指定目录的写入权限。

```
$ ./awscli-bundle/install -b ~/bin/aws
```

这会将 Amazon CLI 安装到默认位置 (~/.local/lib/aws) 并在 ~/bin/aws 中创建符号链接 (symlink)。确保您的 ~/bin 环境变量中包含 $PATH，以使该符号链接生效。

```
$ echo $PATH | grep ~/bin     // See if $PATH contains ~/bin (output will be empty
 if it doesn't)
$ export PATH=~/bin:$PATH     // Add ~/bin to $PATH if necessary
```

4. 确保安装 Amazon CLI 版本 1 的文件夹是 $PATH 变量的一部分。

   a. 在您的用户文件夹中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 echo $SHELL。

```
$ ls -a ~
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

   - Bash – .bash_profile、.profile 或 .bash_login
   - Zsh – .zshrc
   - Tcsh – .tcshrc、.cshrc 或 .login

   b. 在配置文件脚本末尾添加与以下示例类似的导出命令。

```
export PATH=~/.local/bin:$PATH
```

此命令将路径（在本示例中为 ~/.local/bin）插入到现有 PATH 变量的前面。

   c. 将配置文件重新加载到当前会话中，以使更改生效。

```
$ source ~/.bash_profile
```

5. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

如果出现错误，请参阅排查 Amazon CLI 错误。

卸载 Amazon CLI 版本 1 捆绑安装程序

1. 捆绑的安装程序将除可选符号链接之外的所有内容放在安装目录中，因此要卸载，您只需删除这两个项目即可。

```
$ sudo rm -rf /usr/local/aws
$ sudo rm /usr/local/bin/aws
```

2. （可选）删除 .aws 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

.aws 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *~/.aws/*。如果您的用户对此目录具有写入权限，则无需使用 sudo。

```
$ sudo rm ~/.aws/
```

使用 pip 安装、更新和卸载 Amazon CLI 版本 1

您可以直接使用 pip 安装 Amazon CLI。

主题

- 安装 pip
- 使用 pip 安装和更新 Amazon CLI
- 将 Amazon CLI 版本 1 可执行文件添加到 macOS 命令行路径
- 使用 pip 卸载 Amazon CLI

安装 pip

如果尚未安装 pip，可以使用 Python 打包权威机构 提供的脚本进行安装。运行 `pip --version` 可查看您的 Linux 版本是否已包含 Python 和 pip。如果您安装了 Python 3 或更高版本，我们建议您使用 pip3 命令。

1. 使用 curl 命令下载安装脚本。以下命令使用 -O（大写字母“O”）参数指定下载的文件将使用与远程主机上相同的名称存储在当前的文件夹中。

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   ```

2. 使用 python 或 python3 命令运行脚本以下载并安装最新版本的 pip 和其他必需的支持包。当您包含 --user 开关时，脚本将 pip 安装到路径 ~/.local/bin。

   ```
   $ python3 get-pip.py --user
   ```

使用 pip 安装和更新 Amazon CLI

1. 使用 pip 或 pip3 命令安装 Amazon CLI。如果您使用的是 Python 3 或更高版本，我们建议您使用 pip3 命令。

   对于最新版本的 Amazon CLI，请使用以下命令块：

   ```
   $ pip3 install awscli --upgrade --user
   ```

   对于特定版本的 Amazon CLI，在文件名后附加两个等号 = 和版本号。在本示例中，版本 *1.16.312* 的文件名为 *==1.16.312*，这会生成以下命令：

   ```
   $ pip3 install awscli==1.16.312 --upgrade --user
   ```

   > **Note**
   >
   > 为您的终端使用适当的引用规则。要使用 = 字符，您可能需要使用单引号或双引号适当地进行转义。以下示例使用单引号进行转义：
   >
   > ```
   > $ pip3 install 'awscli==1.16.312' --upgrade --user
   > ```

2. 验证 Amazon CLI 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Darwin/23.3.0 botocore/1.18.6
```

如果未找到该程序，请将它添加到命令行路径。

将 Amazon CLI 版本 1 可执行文件添加到 macOS 命令行路径

在使用 pip 进行安装后，可能需要将 aws 程序添加到操作系统的 PATH 环境变量中。程序的位置取决于 Python 的安装位置。

Example Amazon CLI 安装位置 - 带 Python 3.6 和 **pip**（用户模式）的 macOS

```
~/Library/Python/3.7/bin
```

将上面示例中的版本替换为您的 Python 版本。

如果您不知道 Python 的安装位置，请运行 `which python`。

```
$ which python
/usr/local/bin/python
```

输出可能是符号链接的路径，而不是实际的程序。运行 `ls -al` 以查看所指向的路径。

```
$ ls -al /usr/local/bin/python
~/Library/Python/3.7/bin/python3.7
```

pip 将程序安装到 Python 应用程序所在的文件夹中。将此文件夹添加到 PATH 变量。

修改您的 **PATH** 变量

1. 在您的用户目录中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 `echo $SHELL`。

```
$ ls -a ~
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

   - Bash – `.bash_profile`、`.profile` 或 `.bash_login`

- Zsh – `.zshrc`
- Tcsh – `.tcshrc`、`.cshrc` 或 `.login`

2. 向配置文件脚本中添加导出命令。

```
export PATH=~/.local/bin:$PATH
```

在本示例中，此命令将路径 `~/.local/bin` 添加到当前 `PATH` 变量中。

3. 将更新的配置文件加载到当前会话中。

```
$ source ~/.bash_profile
```

使用 pip 卸载 Amazon CLI

1. 如果您使用 `pip` 安装了 Amazon CLI 版本 1，则还必须使用 `pip` 进行卸载。

```
$ pip uninstall awscli
```

如果您使用的是 Python 版本 2 或 3，则可能需要使用 `pip2` 或 `pip3` 命令。使用 `aws --version` 命令确定与您安装的 Amazon CLI 版本 1 关联的 Python 版本。

```
$ pip3 uninstall awscli
```

您可能需要重新启动命令提示符窗口或电脑才能删除所有文件。

2. （可选）删除 `.aws` 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

`.aws` 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *~/.aws/*。如果您的用户对此目录具有写入权限，则无需使用 `sudo`。

```
$ sudo rm -rf ~/.aws/
```

Amazon CLI 安装和卸载错误故障排除

如果您在安装或卸载 Amazon CLI 后遇到问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “找不到命令错误”、the section called ““aws --version”命令返回的版本与您安装的版本不同”和the section called “卸载 Amazon CLI 后，“aws --version”命令返回一个版本”。

在 Windows 上安装、更新和卸载 Amazon CLI 版本 1

可以在 Windows 上使用独立安装程序（建议）或 pip（一种适用于 Python 的程序包管理器）来安装 Amazon Command Line Interface (Amazon CLI) 的版本 1。

键入命令时，请勿包含提示符符号 (C:\>)。程序列表中包含这些符号是为了区分您键入的命令与 Amazon CLI 返回的输出。除非是特定于 Windows 的命令，否则本指南其余部分使用通用提示符符号 ($)。

主题

- 使用 MSI 安装程序来安装、更新和卸载 Amazon CLI 版本 1
- 在 Windows 上使用 Python 和 pip 安装、更新和卸载 Amazon CLI 版本 1
- 将 Amazon CLI 版本 1 可执行文件添加到命令行路径
- Amazon CLI 安装和卸载错误故障排除

使用 MSI 安装程序来安装、更新和卸载 Amazon CLI 版本 1

Windows XP 或更高版本支持 Amazon CLI 版本 1。对于 Windows 用户，MSI 安装程序包提供了一种熟悉而方便的方式来安装 Amazon CLI 版本 1，且无需安装其他任何必备软件。

使用 MSI 安装程序来安装和更新 Amazon CLI 版本 1

查看 GitHub 上的版本页面，了解何时发布了最新版本。更新发布后，您必须重复安装过程以获取最新版本的 Amazon CLI 版本 1。

1. 下载相应的 MSI 安装程序：

   - 适用于 Windows（64 位）的 Amazon CLI MSI 安装程序：https://s3.amazonaws.com/aws-cli/AWSCLI64PY3.msi
   - 适用于 Windows（32 位）的 Amazon CLI MSI 安装程序：https://s3.amazonaws.com/aws-cli/AWSCLI32PY3.msi

- 适用于 Windows 的 Amazon CLI 合并安装文件：https://s3.amazonaws.com/aws-cli/AWSCLISetup.exe（同时包括 32 位和 64 位 MSI 安装程序，并自动安装正确的版本）

2. 运行下载的 MSI 安装程序或设置文件。
3. 按照屏幕上的说明进行操作。默认情况下，Amazon CLI 版本 1 安装到 `C:\Program Files\Amazon\AWSCLI`（64 位版本）或 `C:\Program Files (x86)\Amazon\AWSCLI`（32 位版本）。
4. 要确认安装，请在命令提示符下使用 `aws --version` 命令（打开 Start（开始）菜单并搜索 `cmd` 以启动命令提示符）。

```
C:\> aws --version
aws-cli/1.40.41 Python/3.11.6 Windows/10 botocore/1.18.6
```

如果 Windows 无法找到该程序，您需要关闭并重新打开命令提示符以刷新该路径，或手动将安装目录添加到您的 PATH 环境变量。

卸载 Amazon CLI 版本 1

要使用以下卸载说明，您需要已经使用 MSI 安装程序或安装文件安装了 Amazon CLI 版本 1。

1. 通过执行以下操作之一打开程序和功能：

   - 打开控制面板，然后选择程序和功能。
   - 打开命令提示符，然后运行以下命令。

   ```
   C:\> appwiz.cpl
   ```

2. 选择名为 Amazon Command Line Interface 的条目，然后选择 Uninstall（卸载）启动卸载程序。
3. 确认您要卸载 Amazon CLI。
4. （可选）删除 `.aws` 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

`.aws` 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *`%UserProfile%\.aws`*。

```
$ rmdir %UserProfile%\.aws
```

在 Windows 上使用 Python 和 pip 安装、更新和卸载 Amazon CLI 版本 1

Python Software Foundation 为包含 pip 的 Windows 提供了安装程序。

先决条件

您必须已安装 Python 3.8 或更高版本。有关安装说明，请参阅 Python 的初学者指南 中的下载 Python 页面。

使用 pip 安装和更新 Amazon CLI 版本 1

1. 要安装 Amazon CLI 版本 1，请使用 pip3 命令（如果使用 Python 版本 3 或更高版本）或 pip 命令。

   对于最新版本的 Amazon CLI，请使用以下命令块：

   ```
   C:\> pip3 install awscli --upgrade --user
   ```

   对于特定版本的 Amazon CLI，在文件名中附加一个小于号 < 和版本号。在本示例中，版本 *1.16.312* 的文件名为 *<1.16.312*，这会生成以下命令：

   ```
   C:\> pip3 install awscli<1.16.312 --upgrade --user
   ```

2. 验证 Amazon CLI 版本 1 是否已正确安装。如果没有响应，请参阅 将 Amazon CLI 版本 1 可执行文件添加到命令行路径 部分。

   ```
   C:\> aws --version
   aws-cli/1.40.41 Python/3.11.6 Windows/10 botocore/1.18.6
   ```

使用 pip 卸载 Amazon CLI 版本 1

1. 如果您使用 pip 安装了 Amazon CLI 版本 1，则还必须使用 pip 进行卸载。

   ```
   C:\> pip uninstall awscli
   ```

如果您使用的是 Python 版本 2 或 3，则可能需要使用 pip2 或 pip3 命令。使用 aws --version 命令确定与您安装的 Amazon CLI 版本 1 关联的 Python 版本。

```
C:\> pip3 uninstall awscli
```

您可能需要重新启动命令提示符窗口或电脑才能删除所有文件。

2. （可选）删除 .aws 文件夹中的共享 Amazon SDK 和 Amazon CLI 设置信息。

> **Warning**
>
> 这些配置和凭证设置跨所有 Amazon SDK 和 Amazon CLI 进行共享。如果删除此文件夹，则您系统上的任何 Amazon SDK 都无法访问它们。

.aws 文件夹的默认位置因平台而异，默认情况下，该文件夹位于 *%UserProfile%\.aws*。

```
$ rmdir %UserProfile%\.aws
```

将 Amazon CLI 版本 1 可执行文件添加到命令行路径

在使用 pip 安装 Amazon CLI 版本 1 后，将 aws 程序添加到操作系统的 PATH 环境变量中。通过 MSI 安装，这应该会自动发生。但是，如果在安装它后未运行 aws 命令，则可能需要手动设置它。

1. 使用 where 命令查找 aws 文件位置。默认情况下，where 命令显示在系统的 PATH 中找到指定程序的位置。

```
C:\> where aws
```

所显示的路径取决于平台和安装 Amazon CLI 所采用的方法。包含版本号的文件夹名称可能有所不同。这些示例反映所使用的是 Python 版本 3.7。根据需要将版本替换为您正在使用的版本号。典型路径包括：

- Python 3 和 **pip3** – C:\Program Files\Python37\Scripts\
- Windows 较早版本上的 Python 3 和 **pip3** --user 选项 – %USERPROFILE%\AppData\Local\Programs\Python\Python37\Scripts

- Windows 10 上的 Python 3 和 **pip3** --user 选项 – `%USERPROFILE%\AppData\Roaming\Python\Python37\Scripts`
- MSI 安装程序（64 位）– `C:\Program Files\Amazon\AWSCLI\bin`
- MSI 安装程序（32 位）– `C:\Program Files (x86)\Amazon\AWSCLI\bin`

根据是否返回文件路径，使用以下步骤。

A file path is returned

```
C:\> where aws
C:\Program Files\Amazon\AWSCLI\bin\aws.exe
```

您可以通过运行以下命令找到安装 aws 程序的位置。

```
C:\> where c:\ aws
C:\Program Files\Python37\Scripts\aws
```

A file path is NOT returned

如果 where 命令返回以下错误，这表示程序并不在系统 PATH 下，因此您无法通过输入其名称来运行它。

```
C:\> where c:\ aws
INFO: Could not find files for the given pattern(s).
```

在这种情况下，请运行带有 where 参数的 /R *path* 命令，以告诉它搜索所有文件夹，然后手动添加路径。使用命令行或文件资源管理器发现它在电脑上的安装位置。

```
C:\> where /R c:\ aws
c:\Program Files\Amazon\AWSCLI\bin\aws.exe
c:\Program Files\Amazon\AWSCLI\bincompat\aws.cmd
c:\Program Files\Amazon\AWSCLI\runtime\Scripts\aws
c:\Program Files\Amazon\AWSCLI\runtime\Scripts\aws.cmd
...
```

2. 按 Windows 键并输入 **environment variables**。
3. 选择 Edit environment variables for your account（编辑您账户的环境变量）。
4. 选择 PATH，然后选择 Edit（编辑）。

5. 将找到的路径添加到 Variable value（变量值）字段中，例如 ***C:\Program Files\Amazon\AWSCLI\bin\aws.exe***。
6. 选择 OK（确定）两次以应用新设置。
7. 关闭任何运行的命令提示符并重新打开命令提示符窗口。

Amazon CLI 安装和卸载错误故障排除

如果您在安装或卸载 Amazon CLI 后遇到问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “找不到命令错误”、the section called ““aws --version”命令返回的版本与您安装的版本不同”和the section called “卸载 Amazon CLI 后，“aws --version”命令返回一个版本”。

在虚拟环境中安装和更新 Amazon CLI 版本 1

您可以通过在虚拟环境中安装 Amazon Command Line Interface (Amazon CLI) 的版本 1，避免与其他 pip 程序包之间的版本要求冲突。

主题

- 先决条件
- 在虚拟环境中安装和更新 Amazon CLI 版本 1
- Amazon CLI 安装和卸载错误故障排除

先决条件

- Python 3.8 或更高版本。有关安装说明，请参阅 Python 的初学者指南 中的下载 Python 页面。

  Python 版本支持矩阵

Amazon CLI version	支持的 Python 版本
1.39.0 – 当前	Python 3.9+
1.32.0 – 1.38.38	Python 3.8+
1.27.0 – 1.31.x	Python 3.7+

Amazon CLI version	支持的 Python 版本
1.20.0 – 1.26.x	Python 3.6+
1.19.0 — 1.19.x	Python 2.7+、Python 3.6+
1.17 – 1.18.x	Python 2.7+、Python 3.4+
1.0 – 1.16.x	Python 2.6 及更早版本，Python 3.3 及更早版本

- `pip` 或 `pip3` 已安装。

在虚拟环境中安装和更新 Amazon CLI 版本 1

1. 使用 `pip` 安装 `virtualenv`。

```
$ pip install --user virtualenv
```

2. 创建虚拟环境并命名它。

```
$ virtualenv ~/cli-ve
```

或者，您也可以使用 `-p` 选项指定默认版本以外的 Python 版本。

```
$ virtualenv -p /usr/bin/python37 ~/cli-ve
```

3. 激活新虚拟环境。

Linux 或 macOS

```
$ source ~/cli-ve/bin/activate
```

Windows

```
$ %USERPROFILE%\cli-ve\Scripts\activate
```

提示符更改为显示您的虚拟环境处于活动状态。

```
(cli-ve)~$
```

4. 将 Amazon CLI 版本 1 安装到您的虚拟环境中或进行更新。

```
(cli-ve)~$ pip install --upgrade awscli
```

5. 验证 Amazon CLI 版本 1 是否已正确安装。

```
$ aws --version
aws-cli/1.40.41 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

6. 您可以使用 `deactivate` 命令退出虚拟环境。不管何时启动新会话，都必须重新激活环境。

Amazon CLI 安装和卸载错误故障排除

如果您在安装或卸载 Amazon CLI 后遇到问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “找不到命令错误”、the section called ““aws --version”命令返回的版本与您安装的版本不同”和the section called “卸载 Amazon CLI 后，“aws --version”命令返回一个版本”。

配置 Amazon CLI 设置

本部分介绍如何配置 Amazon Command Line Interface (Amazon CLI) 用于与 Amazon 交互的设置。这些功能包括：

- 凭证将标识 API 的调用方。访问凭证用于加密向 Amazon 服务器发出的请求，以确认您的身份并检索相关的权限策略。这些权限将决定您可以执行的操作。有关如何设置凭证的信息，请参阅 身份验证和访问凭证。
- 其他配置详细信息，用于说明 Amazon CLI 如何处理请求，例如默认输出格式和默认 Amazon 区域。

> **Note**
>
> Amazon要求所有传入的请求都进行加密签名。Amazon CLI 为您执行该操作。“签名”包括日期/时间戳。因此，您必须确保正确设置电脑的日期和时间。否则，如果签名中的日期/时间与 Amazon 服务认定的日期/时间相差太远，Amazon 会拒绝请求。

配置和凭证优先顺序

凭证和配置设置位于不同位置（例如，系统或用户环境变量、本地 Amazon 配置文件）或在命令行上显式声明为参数。某些位置优先于其他位置。Amazon CLI 凭证和配置设置的优先顺序如下：

1. 命令行选项 – 覆盖任何其他位置的设置，例如 `--region`、`--output` 和 `--profile` 参数。
2. 环境变量 – 您可以在系统的环境变量中存储值。
3. 代入角色 – 通过配置或 `assume-role` 命令代入 IAM 角色的权限。
4. 使用 Web 身份代入角色 – 通过配置或 `assume-role-with-web-identity` 命令使用 Web 身份代入 IAM 角色的权限。
5. 凭证文件 – 在运行命令 `aws configure` 时，将更新 `credentials` 和 `config` 文件。`credentials` 文件位于 `~/.aws/credentials`（在 Linux 或 macOS 上）或 `C:\Users\USERNAME\.aws\credentials`（在 Windows 上）。
6. 自定义流程 – 从外部来源获取您的凭证。
7. 配置文件 – 在运行命令 `aws configure` 时，将更新 `credentials` 和 `config` 文件。`config` 文件位于 `~/.aws/config`（在 Linux 或 macOS 上）或 `C:\Users\USERNAME\.aws\config`（在 Windows 上）。

8. 容器凭证 – 您可以将 IAM 角色与每个 Amazon Elastic Container Service (Amazon ECS) 作业定义关联。之后，该任务的容器就可以使用该角色的临时凭证。有关更多信息，请参阅 Amazon Elastic Container Service 开发人员指南中的任务的 IAM 角色。
9. Amazon EC2 实例配置文件凭证 – 您可以将 IAM 角色与每个 Amazon Elastic Compute Cloud (Amazon EC2) 实例关联。之后，在该实例上运行的代码就可以使用该角色的临时凭证。凭证通过 Amazon EC2 元数据服务提供。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 角色和《IAM 用户指南》中的使用实例配置文件。

此部分中的其他主题

- the section called “配置设置”
- the section called “环境变量”
- the section called “命令行选项”
- the section called “ 命令完成”
- the section called “重试”
- the section called “HTTP 代理”

Amazon CLI 中的配置和凭证文件设置

您可以将常用的配置设置和凭证保存在由 Amazon CLI 维护的文件中。

这些文件将分成 `profiles`。默认情况下，Amazon CLI 将使用在名为 `default` 的配置文件中找到的设置。要使用备用设置，您可以创建和引用其他配置文件。

您可以通过设置某个支持的环境变量或使用命令行参数来覆盖个别设置。有关配置设置优先顺序的更多信息，请参阅配置 Amazon CLI 设置。

> **Note**
>
> 有关如何设置凭证的信息，请参阅 身份验证和访问凭证。

主题

- 配置和凭证文件的格式
- 配置设置存储在何处？

- 使用命名配置文件
- 使用命令设置和查看配置设置
- 设置新的配置和凭证命令示例
- 支持的 config 文件设置

配置和凭证文件的格式

config 和 credentials 文件将归入各个节中。节包括 profiles 和 services。节是一个命名的设置集合，它一直持续到遇到另一个节定义行为止。可将多个配置文件和节存储在 config 和 credentials 文件中。

这些文件是使用以下格式的纯文本文件：

- 节名称用方括号 [] 括起来，例如 [default]、[profile *user1*] 和 [sso-session]。
- 部分中的所有条目均采用 setting_name=value 的一般形式。
- 可以通过以井号字符（#）开头来注释掉行。

config 和 credentials 文件包含以下节类型：

- 节类型：profile
- 节类型：services

节类型：**profile**

配置文件节名称使用以下格式，具体取决于文件：

- Config 文件：[default] [profile *user1*]
- 凭证文件：[default] [*user1*]

  在 credentials 文件中创建条目时，请勿 使用单词 profile。

每个配置文件都可以指定不同的凭证，还可以指定不同的 Amazon 区域和输出格式。在 config 文件中命名配置文件时，请包括前缀词“profile”，但不要将它包括在 credentials 文件中。

以下示例显示指定了两个配置文件、区域和输出的 credentials 和 config 文件。第一个 [default] 配置文件在运行未指定配置文件的 Amazon CLI 命令时使用。第二个在运行有 --profile user1 参数的 Amazon CLI 命令时使用。

Short-term credentials

此示例介绍来自 Amazon Identity and Access Management 的短期凭证。有关更多信息，请参阅 the section called “短期凭证”。

凭证文件

```
[default]
aws_access_key_id=ASIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token
 = IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRINGEXAMPLE

[user1]
aws_access_key_id=ASIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
aws_session_token
 = fcZib3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRINGEXAMPLE
```

Config 文件

```
[default]
region=us-west-2
output=json

[profile user1]
region=us-east-1
output=text
```

IAM role

此示例介绍代入 IAM 角色。使用 IAM 角色的配置文件从另一个配置文件提取凭证，然后应用 IAM 角色权限。在以下示例中，`default` 是凭证的源配置文件，`user1` 将借用相同的凭证，然后代入新角色。有关更多信息，请参阅 the section called “IAM 角色”。

凭证文件

`credentials` 文件取决于源配置文件使用的身份验证。在以下示例中，源配置文件使用短期凭证。

```
[default]
```

```
aws_access_key_id=ASIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token
 = IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRINGEXAMPLE
```

Config 文件

```
[default]
region=us-west-2
output=json

[profile user1]
role_arn=arn:aws:iam::777788889999:role/user1role
source_profile=default
role_session_name=session_user1
region=us-east-1
output=text
```

Amazon EC2 instance metadata credentials

此示例介绍从托管 Amazon EC2 实例元数据中获得的凭证。有关更多信息，请参阅 the section called “Amazon EC2 元数据”。

凭证文件

该 `credentials` 文件不用于此身份验证方法。

Config 文件

```
[default]
role_arn=arn:aws:iam::123456789012:role/defaultrole
credential_source=Ec2InstanceMetadata
region=us-west-2
output=json

[profile user1]
role_arn=arn:aws:iam::777788889999:role/user1role
credential_source=Ec2InstanceMetadata
region=us-east-1
output=text
```

Long-term credentials

> **Warning**
>
> 为了避免安全风险，在开发专用软件或处理真实数据时，请勿使用 IAM 用户进行身份验证，而是使用与身份提供者的联合身份验证，例如 Amazon IAM Identity Center。

此示例介绍来自 Amazon Identity and Access Management 的长期凭证。有关更多信息，请参阅 the section called “IAM 用户”。

凭证文件

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

[user1]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

Config 文件

```
[default]
region=us-west-2
output=json

[profile user1]
region=us-east-1
output=text
```

有关更多信息以及其他授权和凭证方法，请参阅 the section called “IAM 用户”。

节类型：**services**

`services` 节是一组为 Amazon Web Services 服务 请求配置自定义端点的设置。然后，配置文件将链接到 `services` 节。

```
[profile dev]
```

```
services = my-services
```

services 节按 <SERVICE> = 行分成多个小节，其中 <SERVICE> 是 Amazon Web Services 服务标识键。Amazon Web Services 服务 标识符基于 API 模型的 serviceId，不过需要将所有空格替换为下划线，并将所有字母小写。有关 services 节中要使用的所有服务标识符密钥的列表，请参阅在 Amazon CLI 中使用端点。服务标识符密钥后面是嵌套的设置，每个设置单独成行，缩进两个空格。

以下示例会配置端点，用于在 *dev* 配置文件中使用的 *my-services* 节中向服务发出的请求。后面紧跟的任何缩进行都包含在该小节中，并适用于该服务。

```
[profile dev]
services = my-services

[services my-services]
dynamodb =
  endpoint_url = http://localhost:8000
```

有关特定于服务的端点的更多信息，请参阅 在 Amazon CLI 中使用端点。

如果您的配置文件具有基于角色的凭证，而这些凭证是通过 IAM 代入角色功能的 source_profile 参数配置的，则开发工具包仅使用所指定配置文件的服务配置。它不使用关联有角色的配置文件。例如，使用以下共享 config 文件：

```
[profile A]
credential_source = Ec2InstanceMetadata
endpoint_url = https://profile-a-endpoint.aws/

[profile B]
source_profile = A
role_arn = arn:aws:iam::123456789012:role/roleB
services = profileB

[services profileB]
ec2 =
  endpoint_url = https://profile-b-ec2-endpoint.aws
```

如果您使用配置文件 B 并在代码中调用 Amazon EC2，则端点将解析为 https://profile-b-ec2-endpoint.aws。如果您的代码向其他任何服务发出请求，则端点解析将不遵循任何自定义逻辑。该端点不会解析到配置文件 A 中定义的全局端点。要使全局端点对配置文件 B 生效，您需要直接在配置文件 B 中设置 endpoint_url。

配置设置存储在何处？

Amazon CLI 将使用 `aws configure` 指定的敏感凭证信息存储在主目录中名为 `credentials` 的文件夹中名为 `.aws` 的本地文件中。使用 `aws configure` 指定的较不敏感的配置选项存储在名为 `config` 的本地文件中，该文件也存储在主目录的 `.aws` 文件夹中。

> **在 config 文件中存储凭证**
>
> 您可以将所有配置文件设置保留在一个文件中，因为 Amazon CLI 可以从 `config` 文件中读取凭证。如果两个文件中都有共享相同名称的配置文件的凭证，则凭证文件中的密钥优先。我们建议将凭证保存在 `credentials` 文件中。这些文件也被各种语言软件开发工具包（SDK）使用。如果除了 Amazon CLI 之外，您还使用某个开发工具包，请确认凭证是否应存储在其自己的文件中。

主目录位置因操作系统而异，但在 Windows 中使用环境变量 `%UserProfile%` 引用，在基于 Unix 的系统中使用 `$HOME` 或 `~`（波形符）引用。通过将 `AWS_CONFIG_FILE` 和 `AWS_SHARED_CREDENTIALS_FILE` 环境变量设置为另一个本地路径，可以为文件指定非默认位置。有关详细信息，请参阅 为 Amazon CLI 配置环境变量。

当您使用共享配置文件指定 Amazon Identity and Access Management（IAM）角色时，Amazon CLI 将调用 Amazon STS `AssumeRole` 操作来检索临时凭证。随后，这些凭证将存储起来（存储在 `~/.aws/cli/cache` 中）。后续 Amazon CLI 命令将使用缓存的临时凭证，直到它们过期，这时 Amazon CLI 将自动刷新这些凭证。

使用命名配置文件

如果未明确定义配置文件，则使用 `default` 配置文件。

要使用命名配置文件，请向您的命令添加 `--profile` *`profile-name`* 选项。以下示例列出了使用 `user1` 配置文件中定义的凭证和设置的所有 Amazon EC2 实例。

```
$ aws ec2 describe-instances --profile user1
```

要为多个命令使用一个命名配置文件，可以通过将 `AWS_PROFILE` 环境变量设置为默认配置文件来避免在每个命令中指定配置文件。您可以使用 `--profile` 参数来覆盖此设置。

Linux or macOS

```
$ export AWS_PROFILE=user1
```

Windows

```
C:\> setx AWS_PROFILE user1
```

使用 set 设置环境变量会更改使用的值，直到当前命令提示符会话结束，或者直到您将该变量设置为其他值。

使用 setx 设置环境变量会更改运行命令后创建的所有命令 Shell 中的值。这不会 影响运行命令时已在运行的任何命令 Shell。关闭并重新启动命令 Shell 可查看这一更改的效果。

设置环境变量会更改默认配置文件，直到 Shell 会话结束或直到您将该变量设置为其他值。通过将环境变量放在 shell 的启动脚本中，可使环境变量在未来的会话中继续有效。有关更多信息，请参阅 为 Amazon CLI 配置环境变量。

使用命令设置和查看配置设置

可通过多种方法使用命令来查看和设置配置设置。

aws configure

运行此命令可快速设置和查看凭证、区域和输出格式。以下示例显示了示例值。

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

aws configure set

您可以使用 aws configure set 设置任何凭证或配置设置。使用 --profile 设置指定要查看或修改的配置文件。

例如，以下命令设置名为 region 的配置文件中的 integ。

```
$ aws configure set region us-west-2 --profile integ
```

要删除某个设置，可在文本编辑器中从 config 和 credentials 文件中手动删除该设置。

aws configure get

您可以检索已使用 aws configure get 设置的任何凭证或配置设置。使用 --profile 设置指定要查看或修改的配置文件。

例如，以下命令检索名为 region 的配置文件中的 integ 设置。

```
$ aws configure get region --profile integ
us-west-2
```

如果输出为空，不会显式设置该设置，并将使用默认值。

aws configure list

要列出配置数据，请使用 aws configure list 命令。此命令列出配置文件以及用于指定的配置文件的访问密钥、密钥和区域配置信息。对于每个配置项目，它会显示值、检索配置值的位置以及配置变量名称。

例如，如果您在环境变量中提供 Amazon Web Services 区域，则此命令会显示您配置的区域的名称、该值来自环境变量以及环境变量的名称。

对于角色和 IAM Identity Center 等临时凭证方法，此命令显示临时缓存的访问密钥并显示秘密访问密钥。

```
$ aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************ABCD  shared-credentials-file
secret_key     ****************ABCD  shared-credentials-file
    region                 us-west-2             env    AWS_DEFAULT_REGION
```

设置新的配置和凭证命令示例

以下示例说明如何使用为不同的身份验证方法指定的凭证、区域和输出配置默认配置文件。

Console sessions as short-term credentials

此示例允许您将现有的控制台凭证与 `aws login` 命令一起使用。此登录方法可用于在初始账户设置期间创建的根凭证、IAM 用户或您的身份提供商提供的联合身份。有关更多信息，请参阅使用控制台凭证登录以进行 Amazon 本地开发。

```
$ aws login
No Amazon region has been configured. The Amazon region is the geographic location
 of your Amazon resources.

If you've used Amazon before and already have resources in your account, tell us
 which region they were
created in. If you haven't created resources in your account before, you can pick
 the region closest to you:
https://docs.aws.amazon.com/global-infrastructure/latest/regions/aws-regions.html.
 You are able to change
the region in the CLI at any time with the command `aws configure set region
 NEW_REGION`.

Amazon Region [us-east-1]: us-east-1
Attempting to open the login page for `us-east-1` in your default browser. If the
 browser does not open,
use the following URL to complete your login:

https://us-east-1.signin.aws.amazon.com/v1/authorize<abbreviated>

If you cannot connect to this URL, make sure that you have specified a valid region.

Waiting for login...

Logged in with role `arn:aws:sts::012345678910:user/iam-user`, and configured
 profile
`default` to use `us-east-1`. This session will expire on October 14, 2025 at 2:04
 PST. After this time, you
can renew your session with `aws login`.
```

Short-term credentials

此示例介绍来自 Amazon Identity and Access Management 的短期凭证。aws 配置向导用于设置初始值，然后 `aws configure set` 命令分配所需的最后一个值。有关更多信息，请参阅 the section called “短期凭证”。

```
$ aws configure
```

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
$ aws configure set
 aws_session_token fcZib3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONG
```

IAM role

此示例介绍代入 IAM 角色。使用 IAM 角色的配置文件从另一个配置文件提取凭证，然后应用 IAM 角色权限。在以下示例中，default 是凭证的源配置文件，user1 将借用相同的凭证，然后代入新角色。此过程没有向导，因此，将使用 aws configure set 命令设置每个值。有关更多信息，请参阅 the section called “IAM 角色”。

```
$ aws configure set role_arn arn:aws:iam::123456789012:role/defaultrole
$ aws configure set source_profile default
$ aws configure set role_session_name session_user1
$ aws configure set region us-west-2
$ aws configure set output json
```

Amazon EC2 instance metadata credentials

此示例介绍从托管 Amazon EC2 实例元数据中获得的凭证。此过程没有向导，因此，将使用 aws configure set 命令设置每个值。有关更多信息，请参阅 the section called “Amazon EC2 元数据”。

```
$ aws configure set role_arn arn:aws:iam::123456789012:role/defaultrole
$ aws configure set credential_source Ec2InstanceMetadata
$ aws configure set region us-west-2
$ aws configure set output json
```

Long-term credentials

> **Warning**
>
> 为了避免安全风险，在开发专用软件或处理真实数据时，请勿使用 IAM 用户进行身份验证，而是使用与身份提供者的联合身份验证，例如 Amazon IAM Identity Center。

此示例介绍来自 Amazon Identity and Access Management 的长期凭证。有关更多信息，请参阅 the section called “IAM 用户”。

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

支持的 config 文件设置

主题

- 全局设置
- S3 自定义命令设置

config 文件支持以下设置。将使用指定（或默认）配置文件中列出的值，除非它们被具有相同名称的环境变量或具有相同名称的命令行选项覆盖。有关哪些顺序设置优先的更多信息，请参阅配置 Amazon CLI 设置

全局设置

account_id_endpoint_mode

指定是否使用基于 Amazon 账户的端点 ID 来调用受支持的 Amazon Web Services 服务。有关基于账户的端点的更多信息，请参阅the section called “基于账户的端点”。

此设置可以设为以下值：

- （默认）**preferred**：端点应包含账户 ID（如果有）。
- **disabled**：已解析的端点不包含账户 ID。
- **required**：端点必须包含账户 ID。如果账户 ID 不可用，SDK 会引发错误。

可以被 AWS_ACCOUNT_ID_ENDPOINT_MODE 环境变量覆盖。要使用基于账户的端点，必须在 AWS_ACCOUNT_ID 环境变量或 aws_account_id 设置中设置 ID。

```
account_id_endpoint_mode = preferred
```

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使

用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

api_versions

某些 Amazon 服务维护多个 API 版本以支持向后兼容性。默认情况下，Amazon CLI 命令使用最新的可用 API 版本。您可以通过在 api_versions 文件中包含 config 设置来指定要用于配置文件的 API 版本。

这是一个“嵌套”设置，后跟一个或多个缩进行，每行标识一个 Amazon 服务和要使用的 API 版本。请参阅每项服务的文档以了解可用的 API 版本。

以下示例显示如何为两种 Amazon 服务指定 API 版本。这些 API 版本仅用于在包含这些设置的配置文件下运行的命令。

```
api_versions =
    ec2 = 2015-03-01
    cloudfront = 2015-09-017
```

此设置没有等效的环境变量或命令行参数。

aws_access_key_id

指定用作凭证一部分的对命令请求进行身份验证的 Amazon 访问密钥。虽然它可以存储在 config 文件中，但我们建议您将其存储在 credentials 文件中。

可以被 AWS_ACCESS_KEY_ID 环境变量覆盖。您不能将访问密钥 ID 指定为命令行选项。

```
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
```

aws_account_id

指定要用于调用受支持的 Amazon Web Services 服务的基于 Amazon 账户的端点 ID。有关基于账户的端点的更多信息，请参阅the section called “基于账户的端点”。

可以被 AWS_ACCOUNT_ID 环境变量覆盖。AWS_ACCOUNT_ID_ENDPOINT_MODE 环境变量或 account_id_endpoint_mode 设置必须设置为 preferred 或 required，才能使用此设置。

```
aws_account_id = 123456789EXAMPLE
```

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

aws_secret_access_key

指定用作凭证一部分的对命令请求进行身份验证的 Amazon 私有密钥。虽然它可以存储在 config 文件中，但我们建议您将其存储在 credentials 文件中。

可以被 AWS_SECRET_ACCESS_KEY 环境变量覆盖。您不能将私有访问密钥指定为命令行选项。

```
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

aws_session_token

指定 Amazon 会话令牌。只有在手动指定临时安全凭证时才需要会话令牌。虽然它可以存储在 config 文件中，但我们建议您将其存储在 credentials 文件中。

可以被 AWS_SESSION_TOKEN 环境变量覆盖。您不能将会话令牌指定为命令行选项。

```
aws_session_token = AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

ca_bundle

指定用于验证 SSL 证书的 CA 证书捆绑包（具有 .pem 扩展名的文件）。

可以被 AWS_CA_BUNDLE 环境变量或 --ca-bundle 命令行选项覆盖。

```
ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem
```

cli_follow_urlparam

指定 Amazon CLI 是否尝试跟踪命令行参数中以 http:// 或 https:// 开头的 URL 链接。启用后，将检索到的内容而不是 URL 用作参数值。

- true – 这是默认值。指定后，将抓取任何以 `http://` 或 `https://` 开头的字符串参数，并将任何下载的内容用作该命令的参数值。
- false – 如果指定，Amazon CLI 不会将以 `http://` 或 `https://` 开头的参数字符串值与其他字符串区别对待。

该条目没有等效的环境变量或命令行选项。

```
cli_follow_urlparam = false
```

cli_history

默认情况下禁用。此设置启用 Amazon CLI 的命令历史记录。启用此设置后，Amazon CLI 记录 `aws` 命令的历史记录。

```
cli_history = enabled
```

您可以使用 `aws history list` 命令列出您的历史记录，然后使用 `aws history show` 命令中生成的 `command_ids` 获取详细信息。有关更多信息，请参阅《Amazon CLI 参考指南》中的 [aws history]。

cli_timestamp_format

指定时间戳值的输出格式。可以指定以下任一值：

- iso8601 – Amazon CLI 版本 2 的默认值。如果指定，Amazon CLI 根据 [ISO 8601] 对输出中的所有时间戳进行重新格式化。ISO 8601 格式的时间戳与以下示例类似。以下示例演示了如何通过用 T 分隔日期和时间，并在时间后包含 Z 来格式化时间。

  ```
  YYYY-MM-DDThh:mm:ssZ
  ```

  以下示例显示了使用先前格式的时间戳。

  ```
  2024-05-08T15:16:43Z
  ```

- wire – Amazon CLI 版本 1 的默认值。如果指定，Amazon CLI 按原样显示在 HTTP 查询响应中收到的所有时间戳值。

该设置没有等效的环境变量或命令行选项。此设置不更改时间戳输入，只更改输出格式。

```
cli_timestamp_format = iso8601
```

credential_process

指定 Amazon CLI 运行的外部命令，以生成或检索用于该命令的身份验证凭证。命令必须以特定格式返回凭证。有关如何使用该设置的更多信息，请参阅在 Amazon CLI 中使用外部进程获取凭证。

该条目没有等效的环境变量或命令行选项。

```
credential_process = /opt/bin/awscreds-retriever --username susan
```

credential_source

在 Amazon EC2 实例或容器中使用，指定 Amazon CLI 在何处可以找到要用于代入通过 `role_arn` 参数指定的角色的凭证。不能在同一配置文件中同时指定 `source_profile` 和 `credential_source`。

此参数具有三个值：

- Environment – 指定 Amazon CLI 从环境变量检索源凭证。
- Ec2InstanceMetadata – 指定 Amazon CLI 将使用附加到 EC2 实例配置文件的 IAM 角色以获取源凭证。
- EcsContainer – 指定 Amazon CLI 将附加到 ECS 容器的 IAM 角色用作源凭证。

```
credential_source = Ec2InstanceMetadata
```

duration_seconds

指定角色会话的最大持续时间（以秒为单位）。该值的范围在 900 秒（15 分钟）到角色的最大会话持续时间设置之间。此参数为可选参数，默认情况下，该值设置为 3600 秒。

endpoint_url

指定用于所有服务请求的端点。如果在 `config` 文件的 `services` 节中使用此设置，则该端点仅用于指定的服务。有关更多信息，请参阅 the section called “为所有 Amazon Web Services 服务设置全局端点”。

以下示例使用全局端点 `http://localhost:1234` 和用于 Amazon S3 的特定于服务的端点 `http://localhost:4567`。

```
[profile dev]
endpoint_url = http://localhost:1234
services = s3-specific
```

```
[services s3-specific]
s3 =
  endpoint_url = http://localhost:4567
```

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

ignore_configure_endpoint_urls

如果启用，则 Amazon CLI 会忽略 config 文件中指定的所有自定义端点配置。有效值为 **true** 和 **false**。

```
ignore_configure_endpoint_urls = true
```

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

external_id

指定第三方用于在其客户账户中代入角色的唯一标识符。这将映射到 ExternalId 操作中的 AssumeRole 参数。仅当角色的信任策略为 ExternalId 指定值时，才需要此参数。有关更多信息，请参阅《IAM 用户指南》中的如何在向第三方授予对 Amazon 资源的访问权限时使用外部 ID。

max_attempts

指定 Amazon CLI 重试处理程序使用的最大重试次数值，其中初始调用计入您提供的 max_attempts 值。

您可以使用 AWS_MAX_ATTEMPTS 环境变量覆盖此值。

```
max_attempts = 3
```

mfa_serial

代入角色时要使用的 MFA 设备的标识号。仅当代入角色的信任策略包含需要 MFA 身份验证的条件，此值才是必需的。该值可以是硬件设备（例如 GAHT12345678）的序列号，也可以是虚拟 MFA 设备（例如 arn:aws:iam::123456789012:mfa/*user*）的 Amazon 资源名称（ARN）。

output

指定使用该配置文件请求的命令的默认输出格式。您可以指定以下任意值：

- **json** – 输出采用 JSON 字符串的格式。
- **text** – 输出采用多个制表符分隔字符串值行的格式。这对于将输出传递到文本处理器（如 grep、sed 或 awk）很有用。
- **table** – 输出采用表格形式，使用字符 +|- 以形成单元格边框。它通常以“人性化”格式呈现信息，这种格式比其他格式更容易阅读，但从编程方面来讲不是那么有用。

可以被 AWS_DEFAULT_OUTPUT 环境变量或 --output 命令行选项覆盖。

```
output = table
```

parameter_validation

指定 Amazon CLI 客户端在将参数发送到 Amazon 服务端点之前是否尝试验证参数。

- true – 这是默认值。指定后，Amazon CLI 将执行命令行参数的本地验证。
- false – 如果指定，Amazon CLI 在将命令行参数发送到 Amazon 服务端点前不对其进行验证。

该条目没有等效的环境变量或命令行选项。

```
parameter_validation = false
```

region

对于使用该配置文件请求的命令，指定要将请求发送到的 Amazon Web Services 区域。

- 您可以指定可用于所选服务的任何区域代码，有关区域代码的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon 区域和端点。
- 通过 aws_global，您可以为不仅支持区域端点，还支持全局端点的服务指定全局端点，例如 Amazon Security Token Service（Amazon STS）和 Amazon Simple Storage Service（Amazon S3）。

您可以使用 、AWS_DEFAULT_REGION 环境变量或 --region 命令行选项覆盖此值。

```
region = us-west-2
```

request_checksum_calculation

指定何时计算请求有效载荷的校验和，并提供以下选项：

- when_supported – (默认) 当操作在其服务模型中指定校验和算法或需要请求校验和时，将计算请求有效载荷校验和。
- when_required – 当操作需要请求校验和或用户提供由 Amazon Web Services 服务 建模的 requestAlgorithmMember 时，将计算请求有效载荷校验和。

```
request_checksum_calculation = when_supported
```

环境变量 AWS_REQUEST_CHECKSUM_CALCULATION 将覆盖此设置。

response_checksum_validation

指定何时对响应有效载荷执行校验和验证，并提供以下选项：

- when_supported – (默认) 当操作在其服务模型中指定 Amazon CLI 支持的响应算法时，将执行响应有效载荷校验和验证。
- when_required：当操作在其服务模型中指定 Amazon CLI 支持的响应算法，并且您在操作输出中将建模的 requestValidationModeMember 设置为 ENABLED 时，将执行响应有效载荷校验和验证。

```
response_checksum_validation = when_supported
```

环境变量 AWS_RESPONSE_CHECKSUM_VALIDATION 将覆盖此设置。

retry_mode

指定 Amazon CLI 使用哪种重试模式。有三种重试模式可用：standard、legacy (默认) 和 adaptive。有关重试的更多信息，请参阅Amazon CLI 中的 Amazon CLI 重试次数 。

您可以使用 AWS_RETRY_MODE 环境变量覆盖此值。

```
retry_mode = standard
```

role_arn

指定要用于运行 Amazon CLI 命令的 IAM 角色的 Amazon 资源名称（ARN）。此外，还必须指定以下参数之一以标识有权代入此角色的凭证：

- source_profile
- credential_source

```
role_arn = arn:aws:iam::123456789012:role/role-name
```

环境变量 AWS_ROLE_ARN 将覆盖此设置。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

role_session_name

指定要附加到角色会话的名称。此值在 RoleSessionName 调用 Amazon CLI 操作时将提供给 AssumeRole 参数，并成为代入角色用户 ARN 的一部分：arn:aws:sts::*123456789012*:assumed-role/*role_name*/*role_session_name*。此参数为可选参数。如果未提供此值，则将自动生成会话名称。此名称显示在与此会话关联的条目的 Amazon CloudTrail 日志中。

```
role_session_name = maria_garcia_role
```

环境变量 AWS_ROLE_SESSION_NAME 将覆盖此设置。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

services

指定要用于您的配置文件的服务配置。

```
[profile dev-s3-specific-and-global]
endpoint_url = http://localhost:1234
services = s3-specific

[services s3-specific]
s3 =
  endpoint_url = http://localhost:4567
```

有关更多信息，请参阅 the section called “services” 中的 services 节。

环境变量 AWS_ROLE_SESSION_NAME 将覆盖此设置。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

sdk_ua_app_id

一个 Amazon Web Services 账户可以供多个客户应用程序来调用 Amazon Web Services 服务。应用程序 ID 标识哪个源应用程序使用 Amazon Web Services 服务 进行了一组调用。AmazonSDK 和服务不会使用或解释此值，除非将其显示在客户通信中。例如，此值可以包含在操作电子邮件中，以唯一标识您的哪个应用程序与通知相关联。

应用程序 ID 是一个字符串，最大长度为 50 个字符。允许使用字母、数字和以下特殊字符：！ $ % & * + - . , ^ _ ` | ~。默认情况下，不分配任何值。

```
sdk_ua_app_id = prod1
```

使用 AWS_SDK_UA_APP_ID 环境变量可以覆盖此设置。您不能将此值设置为命令行参数。

sigv4a_signing_region_set

使用逗号分隔列表指定在通过 SigV4a 进行签名时要使用的区域。如果未设置此变量，Amazon CLI 会使用 Amazon Web Services 服务 所使用的默认值。如果 Amazon Web Services 服务 没有默认值，则请求签名将使用 * 作为值，在所有区域内生效。

```
sigv4a_signing_region_set = us-west-2, us-east-1
```

有关 SigV4a 的更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的 Amazon 签名版本 4。

使用 AWS_SIGV4A_SIGNING_REGION_SET 环境变量可以覆盖此设置。您不能将此值设置为命令行参数。

source_profile

指定包含长期凭证的命名配置文件，Amazon CLI 可使用这些凭证代入通过 `role_arn` 参数指定的角色。不能在同一配置文件中同时指定 `source_profile` 和 `credential_source`。

```
source_profile = production-profile
```

sts_regional_endpoints

指定 Amazon CLI 如何确定 Amazon CLI 客户端用于与 Amazon Security Token Service (Amazon STS) 通信的 Amazon 服务终端节点。Amazon CLI 版本 1 的默认值为 `legacy`。

您可以指定以下两个值之一：

- **legacy** – 对以下Amazon区域使用全局 STS 端点 sts.amazonaws.com：ap-northeast-1、ap-south-1、ap-southeast-1、ap-southeast-2、aws-global、ca-central-1、eu-central-1、eu-north-1、eu-west-1、eu-west-2、eu-west-3、sa-east-1、us-east-1、us-east-2、us-west-1 和 us-west-2。所有其他区域自动使用其各自的区域端点。
- **regional** – Amazon CLI 始终使用当前配置的区域的 Amazon STS 端点。例如，如果客户端配置为使用 us-west-2，则对 Amazon STS 进行的所有调用都针对区域端点 sts.us-west-2.amazonaws.com 而非全局 sts.amazonaws.com 端点。要在启用此设置时向全局终端节点发送请求，您可以将区域设置为 aws-global。

使用 AWS_STS_REGIONAL_ENDPOINTS 环境变量可以覆盖此设置。您不能将此值设置为命令行参数。

use_dualstack_endpoint

允许使用双堆栈端点发送 Amazon 请求。要详细了解支持 IPv4 和 IPv6 流量的双堆栈端点，请参阅 Amazon Simple Storage Service 用户指南中的使用 Amazon S3 双堆栈端点。双堆栈端点适用于某些区域。如果不存在适用于服务和/或 Amazon Web Services 区域的双堆栈端点，则请求将失败。有效的设置为 true 和 false。默认情况下，将禁用该功能。有关更多信息，请参阅 the section called “设置成为所有 Amazon Web Services 服务使用双堆栈端点”。

该设置与 use_accelerate_endpoint 设置互斥。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

use_fips_endpoint

在某些 Amazon Web Services 区域，部分 Amazon 服务提供支持美国联邦信息处理标准 (FIPS) 140-2 的端点。当 Amazon 服务支持 FIPS 时，此设置指定 Amazon CLI 应使用哪个 FIPS 端点。与标准 Amazon 端点不同，FIPS 端点使用符合 FIPS 140-2 的 TLS 软件库。与美国政府有业务来往的企业可能需要使用这些端点。有关更多信息，请参阅the section called “设置成为所有 Amazon Web Services 服务使用 FIPS 端点”。

如果启用此设置，但不存在适用于您所在 Amazon Web Services 区域中的服务的 FIPS 端点，则 Amazon 命令可能会失败。在这种情况下，请使用 `--endpoint-url` 选项手动指定要在命令中使用的端点，或者使用特定于服务的端点。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

`web_identity_token_file`

指定一个文件的路径，该文件包含由身份提供者提供的 OAuth 2.0 访问令牌或 OpenID Connect ID 令牌。Amazon CLI 加载此文件的内容，并将其作为 `WebIdentityToken` 参数传递给 `AssumeRoleWithWebIdentity` 操作。

环境变量 `AWS_WEB_IDENTITY_TOKEN_FILE` 将覆盖此设置。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

`tcp_keepalive`

指定 Amazon CLI 客户端是否使用 TCP keep-alive 数据包。

该条目没有等效的环境变量或命令行选项。

```
tcp_keepalive = false
```

S3 自定义命令设置

Amazon S3 支持多项配置 Amazon CLI 如何执行 Amazon S3 操作的设置。一些设置适用于 `s3api` 和 `s3` 命名空间中的所有 S3 命令。其他的则专门用于抽象常见操作的 S3“自定义”命令，而不仅仅是对 API 操作的一对一映射。`aws s3` 传输命令 `cp`、`sync`、`mv` 和 `rm` 具有可用于控制 S3 传输的其他设置。

可以通过在 `s3` 文件中指定 `config` 嵌套设置来配置所有这些选项。每个设置在其自己的行上缩进。

> **Note**
>
> 这些设置完全是可选的。即使不配置这些设置中的任何一个，您也应该能够成功使用 aws s3 传输命令。提供这些设置是为了让您能够调整性能或匹配运行这些 aws s3 命令的特定环境。

这些设置都在 config 文件中的顶层 s3 键下设置，如以下 development 配置文件示例所示。

```
[profile development]
s3 =
  max_concurrent_requests = 20
  max_queue_size = 10000
  multipart_threshold = 64MB
  multipart_chunksize = 16MB
  max_bandwidth = 50MB/s
  use_accelerate_endpoint = true
  addressing_style = path
```

以下设置适用于 s3 或 s3api 命名空间中的任何 S3 命令。

addressing_style

指定要使用的寻址样式。这将控制存储桶名称是位于主机名还是 URL 中。有效值包括 path、virtual 和 auto。默认值为 auto。

构造 Amazon S3 端点的样式有两种。第一种称为 virtual，它将存储桶名称包含为主机名的一部分。例如：https://*bucketname*.s3.amazonaws.com。另一种为 path 样式，您将存储桶名称视为 URI 中的路径，例如 https://s3.amazonaws.com/*bucketname*。CLI 中的默认值是使用 auto，它尝试尽可能使用 virtual 样式，但在需要时回退到 path 样式。例如，如果您的存储桶名称与 DNS 不兼容，则存储桶名称不能是主机名的一部分，而必须位于路径中。使用 auto 时，CLI 将检测这种情况并自动切换到 path 样式。如果将寻址方式设置为 path，您必须确保在 Amazon CLI 中配置的 Amazon 区域与存储桶的区域匹配。

payload_signing_enabled

指定是否对 sigv4 负载进行 SHA256 签名。默认情况下，使用 HTTPS 时，将对流式上传（UploadPart 和 PutObject）禁用该设置。默认情况下，对于流式上传（UploadPart 和 PutObject），此设置为 false，但仅限存在 ContentMD5（默认生成）并且端点使用 HTTPS 时。

如果设置为 true，则 S3 请求接收 SHA256 校验和形式的额外内容验证（替您计算并包含在请求签名中）。如果设置为 false，则不计算校验和。禁用该设置可减少校验和计算产生的性能开销。

use_accelerate_endpoint

为所有 s3 和 s3api 命令使用 Amazon S3 加速端点。默认值为 False。该设置与 use_dualstack_endpoint 设置互斥。

如果设置为 true，Amazon CLI 会将所有 Amazon S3 请求定向到 s3-accelerate.amazonaws.com 的 S3 Accelerate 端点。要使用该端点，您必须让您的存储桶使用 S3 Accelerate。使用存储桶寻址的虚拟样式发送所有请求：*my-bucket*.s3-accelerate.amazonaws.com。不会将任何 ListBuckets、CreateBucket 和 DeleteBucket 请求发送到 S3 加速端点，因为该端点不支持这些操作。如果将任何 --endpoint-url 或 https://s3-accelerate.amazonaws.com 命令的 http://s3-accelerate.amazonaws.com 参数设置为 s3 或 s3api，也可以设置该行为。

use_dualstack_endpoint

支持使用双堆栈端点来发送 s3 和 s3api 请求。要详细了解支持 IPv4 和 IPv6 流量的双堆栈端点，请参阅 Amazon Simple Storage Service 用户指南中的使用 Amazon S3 双堆栈端点。双堆栈端点适用于某些区域。如果不存在适用于服务和/或 Amazon Web Services 区域的双堆栈端点，则请求将失败。有效的设置为 true 和 false。默认情况下，将禁用该功能。有关更多信息，请参阅 the section called “设置成为所有 Amazon Web Services 服务使用双堆栈端点”。

该设置与 use_accelerate_endpoint 设置互斥。

以下设置仅适用于 s3 命名空间命令集中的命令。

max_bandwidth

指定向 Amazon S3 上传数据和从其下载数据可使用的最大带宽。默认为无限制。

这限制了 S3 命令可用于向 Amazon S3 传输数据和从 Amazon S3 传输数据的最大带宽。该值仅适用于上传和下载；它不适用于复制或删除。值以每秒字节数表示。该值可以指定为：

- 一个整数。例如，1048576 将最大带宽使用率设置为每秒 1 兆字节。
- 一个整数，后跟速率后缀。可以使用以下格式指定速率后缀：KB/s、MB/s 或 GB/s。例如，300KB/s 和 10MB/s。

通常，我们建议您先尝试通过降低 max_concurrent_requests 来降低带宽使用率。如果这样做没有充分地将带宽使用率限制到所需速率，您可以使用 max_bandwidth 设置进一步限制

带宽使用率。这是因为 max_concurrent_requests 控制当前运行的线程数。如果您先降低 max_bandwidth 但保持较高的 max_concurrent_requests 设置，则可能导致线程不得不进行不必要的等待。这可能造成过多的资源消耗和连接超时。

max_concurrent_requests

指定最大并发请求数。默认值是 10。

aws s3 传输命令是多线程的。在任意给定时间，都可以运行多个 Amazon S3 请求。例如，当您使用命令 aws s3 cp localdir s3://bucket/ --recursive 将文件上传到 S3 存储桶时，Amazon CLI 可以并行上传文件 localdir/file1、localdir/file2 和 localdir/file3。设置 max_concurrent_requests 指定可同时运行的最大传输操作数。

您可能由于以下原因而需要更改该值：

- 减小该值 – 在某些环境中，默认的 10 个并发请求可能会占用过多的系统资源。这可能导致连接超时或系统响应速度变慢。减小该值可减少 S3 传输命令消耗的资源。但不利后果是 S3 传输可能需要更长时间才能完成。如果使用了限制带宽的工具，则可能需要减小该值。
- 增大该值 – 在某些情况下，您可能希望 Amazon S3 传输根据需要使用尽可能多的网络带宽，以尽可能快地完成任务。在这种情况下，默认的并发请求数可能不足以使用所有可用的网络带宽。增大该值可缩短完成 Amazon S3 传输所需的时间。

max_queue_size

指定作业队列中的最大任务数。默认值是 1000。

Amazon CLI 在内部使用这样一种模型：将 Amazon S3 任务排队，然后由数量受 max_concurrent_requests 限制的使用者执行。任务通常映射到单个 Amazon S3 操作。例如，任务可以是 PutObjectTask、GetObjectTask 或 UploadPartTask。任务添加到队列的速度可能比使用者完成任务的速度快得多。为避免无限制增长，作业队列大小设置了特定大小的上限。该设置用于更改该最大数量的值。

您通常不需要更改该设置。该设置还对应于 Amazon CLI 知道需要运行的任务数。这意味着，默认情况下 Amazon CLI 只能查看前 1000 个任务。增大该值意味着 Amazon CLI 可更快得知所需任务的总数（假设排队速度快于任务完成速度）。但不利后果是更大的 max_queue_size 需要更多的内存。

multipart_chunksize

指定 Amazon CLI 用于单个文件的分段传输的块大小。默认值为 8 MB，最少为 5 MB。

当文件传输超出 multipart_threshold 时，Amazon CLI 将文件分成该大小的块。可以使用与 multipart_threshold 相同的语法指定该值，即整数形式的字节数，或使用大小和后缀。

`multipart_threshold`

指定 Amazon CLI 用于单个文件的分段传输的大小阈值。默认值为 8 MB。

上传、下载或复制文件时，如果文件超出该大小，Amazon S3 命令将切换到分段操作。您可以通过以下两种方式之一指定该值：

- 文件大小（以字节为单位）。例如：`1048576`。
- 文件大小及大小后缀。您可以使用 `KB`、`MB`、`GB` 或 `TB`。例如，`10MB`、`1GB`。

> **Note**
>
> S3 可能会对可用于分段操作的有效值施加约束。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的 S3 分段上传文档。

为 Amazon CLI 配置环境变量

环境变量提供了另一种指定配置选项和凭证的方法并且可用于编写脚本。

选项的优先顺序

- 如果您使用本主题中描述的某个环境变量指定选项，则它将在配置文件中覆盖从配置文件加载的任何值。
- 如果您通过在 Amazon CLI 命令行上使用参数指定选项，则它将在配置文件中覆盖相应环境变量或配置文件中的任何值。

有关优先顺序以及 Amazon CLI 如何确定使用哪些凭证的更多信息，请参阅配置 Amazon CLI 设置。

主题

- 如何设置环境变量
- Amazon CLI 支持的环境变量

如何设置环境变量

下面的示例介绍您如何可以为默认用户配置环境变量。

Linux or macOS

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-west-2
```

设置环境变量会更改使用的值，直到 Shell 会话结束或直到您将该变量设置为其他值。通过在 shell 的启动脚本中设置变量，可使变量在未来的会话中继续有效。

Windows Command Prompt

为所有会话设置

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> setx AWS_DEFAULT_REGION us-west-2
```

使用 `setx` 设置环境变量会更改当前命令提示符会话和运行该命令后创建的所有命令提示符会话中使用的值。它不 影响在运行该命令时已经运行的其他命令 shell。您可能需要重启终端来加载设置。

仅为当前会话设置

使用 `set` 设置环境变量会更改使用的值，直到当前命令提示符会话结束，或者直到您将该变量设置为其他值。

```
C:\> set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
C:\> set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> set AWS_DEFAULT_REGION=us-west-2
```

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
PS C:\> $Env:AWS_DEFAULT_REGION="us-west-2"
```

如果在 PowerShell 提示符下设置环境变量（如前面的示例所示），则仅保存当前会话持续时间的值。要在所有 PowerShell 和命令提示符会话中使环境变量设置保持不变，请使用控制面板中的系统应用程序来存储该变量。或者，您可以通过将其添加到 PowerShell 配置文件来为将来的所有 PowerShell 会话设置该变量。有关存储环境变量或跨会话保存它们的更多信息，请参阅 PowerShell 文档。

Amazon CLI 支持的环境变量

Amazon CLI 支持以下环境变量。

AWS_ACCESS_KEY_ID

指定与 IAM 账户关联的 Amazon 访问密钥。

如果已定义此环境变量，它将覆盖配置文件设置 aws_access_key_id 的值。您不能使用命令行选项来指定访问密钥 ID。

AWS_ACCOUNT_ID

指定要用于调用受支持的 Amazon Web Services 服务的基于 Amazon 账户的端点 ID。有关基于账户的端点的更多信息，请参阅the section called “基于账户的端点”。

该设置会覆盖 aws_account_id 设置。AWS_ACCOUNT_ID_ENDPOINT_MODE 环境变量或 account_id_endpoint_mode 设置必须设置为 preferred 或 required，才能使用此设置。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_ACCOUNT_ID_ENDPOINT_MODE

指定是否使用基于 Amazon 账户的端点 ID 来调用受支持的 Amazon Web Services 服务。有关基于账户的端点的更多信息，请参阅the section called “基于账户的端点”。

此设置可以设为以下值：

- （默认）**preferred**：端点应包含账户 ID（如果有）。
- **disabled**：已解析的端点不包含账户 ID。
- **required**：端点必须包含账户 ID。如果账户 ID 不可用，SDK 会引发错误。

该设置会覆盖 account_id_endpoint_mode 设置。要使用基于账户的端点，必须在 AWS_ACCOUNT_ID 环境变量或 aws_account_id 设置中设置 ID。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_CA_BUNDLE

指定要用于 HTTPS 证书验证的证书捆绑包的路径。

如果已定义此环境变量，它将覆盖配置文件设置 `ca_bundle` 的值。您可以使用 `--ca-bundle` 命令行参数覆盖此环境变量。

AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS

如果在使用 `s3 mv` 命令时源存储桶和目标存储桶相同，则可以将源文件或对象移到其自身上，这可能导致源文件或对象被意外删除。`AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS` 环境变量和 `--validate-same-s3-paths` 选项指定是验证接入点 ARN 还是验证您的 Amazon S3 源或目标 URI 中的接入点别名。

> **Note**
>
> 针对 `s3 mv` 的路径验证需要额外的 API 调用。

AWS_CLI_UPGRADE_DEBUG_MODE

启用升级调试模式以引导从 Amazon CLI v1 升级到 Amazon CLI v2 的过程。有效值为 `true` 和 `false`。

如果设置为 `true`，则 Amazon CLI 将检测在 Amazon CLI v2 中有突破性更改的更新功能的使用情况，并针对检测到的可能受影响的每种使用情况输出警告。您可以使用此模式为升级到 v2 准备环境和 Amazon CLI 使用情况。

您可以使用 `--v2-debug` 命令行参数覆盖此环境变量。

有关 Amazon CLI v1 到 v2 升级调试模式的信息，请参阅使用升级调试模式将 Amazon CLI 版本 1 升级到 Amazon CLI 版本 2。

AWS_CONFIG_FILE

指定 Amazon CLI 用于存储配置文件的文件的位置。默认路径为 ~/.aws/config。

您不能在命名配置文件设置中或使用命令行参数来指定此值。

AWS_DATA_PATH

加载 Amazon CLI 数据时要在 ~/.aws/models 的内置搜索路径之外检查的其他目录的列表。设置此环境变量将指示在回滚到内置搜索路径前要先检查的其他目录。应使用 os.pathsep 字符（在 Linux 上为 ：，在 Windows 上为 ；）隔开多个条目。

AWS_DEFAULT_OUTPUT

指定要使用的输出格式。

如果已定义此环境变量，它将覆盖配置文件设置 output 的值。您可以使用 --output 命令行参数覆盖此环境变量。

AWS_DEFAULT_REGION

Default region name 标识默认情况下您要将请求发送到的服务器所在的 Amazon 区域。通常是离您最近的区域，但可以是任意区域。例如，您可以键入 us-west-2 以使用美国西部（俄勒冈）。除非在命令中另行指定，否则这是所有后续请求将发送到的区域。

> **Note**
>
> 使用 Amazon 时，必须明确指定或通过设置原定设置区域来指定 Amazon CLI 区域。有关可用区域的列表，请参阅区域和端点。Amazon CLI 使用的区域指示符与您在 Amazon Web Services 管理控制台 URL 和服务端点中看到的名称相同。

如果已定义此环境变量，它将覆盖 region 配置文件设置的值。您可以使用 --region 命令行参数来覆盖此环境变量。

AWS_EC2_METADATA_DISABLED

禁用 Amazon EC2 实例元数据服务（IMDS）。

如果设置为 true，则不会从 IMDS 请求用户凭证或配置（如区域）。

AWS_ENDPOINT_URL

指定用于所有服务请求的端点。有关更多信息，请参阅 the section called “为所有 Amazon Web Services 服务设置全局端点”。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_ENDPOINT_URL_<SERVICE>

指定用于特定服务的自定义端点，在该服务中 <SERVICE> 替换为 Amazon Web Services 服务 标识符。例如，Amazon DynamoDB 具有 DynamoDB 的 serviceId。对于此服务，端点 URL 环境变量为 AWS_ENDPOINT_URL_DYNAMODB。

有关特定于服务的所有环境变量的列表，请参阅特定于服务的标识符列表。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_IGNORE_CONFIGURED_ENDPOINT_URLS

如果启用，则 Amazon CLI 将忽略所有自定义端点配置。有效值为 **true** 和 **false**。有关更多信息，请参阅 the section called “为所有 Amazon Web Services 服务设置全局端点”。

端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。有关端点优先级，请参阅the section called “端点配置和设置优先级”。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_MAX_ATTEMPTS

指定 Amazon CLI 重试处理程序使用的最大重试次数值，其中初始调用计入您提供的值。有关重试的更多信息，请参阅Amazon CLI 中的 Amazon CLI 重试次数 。

如果已定义此环境变量，它将覆盖配置文件设置 `max_attempts` 的值。

AWS_METADATA_SERVICE_NUM_ATTEMPTS

尝试在已配置 IAM 角色的 Amazon EC2 实例上检索凭证时，Amazon CLI 会在停止前尝试从实例元数据服务检索一次凭证。如果您知道您的命令将在 Amazon EC2 实例上运行，则可增大此值，以使 Amazon CLI 在放弃前重试多次。

AWS_METADATA_SERVICE_TIMEOUT

与实例元数据服务的连接超时前等待的秒数。尝试在已配置 IAM 角色的 Amazon EC2 实例上检索凭证时，默认情况下，与实例元数据服务的连接将在 1 秒后超时。如果您知道您正在已配置 IAM 角色的 Amazon EC2 实例上运行，如有必要，可增大此值。

AWS_PROFILE

指定包含要使用的凭证和选项的 Amazon CLI 配置文件的名称。可以是存储在 `credentials` 或 `config` 文件中的配置文件的名称，也可以是值 `default`，后者使用默认配置文件。

如果您定义了此环境变量，它将在配置文件中覆盖使用名为 `[default]` 的配置文件的行为。您可以使用 `--profile` 命令行参数覆盖此环境变量。

AWS_REQUEST_CHECKSUM_CALCULATION

指定何时计算请求有效载荷的校验和，并提供以下选项：

- `when_supported` – （默认）当操作在其服务模型中指定校验和算法或需要请求校验和时，将计算请求有效载荷校验和。
- `when_required` – 当操作需要请求校验和或用户提供由 Amazon Web Services 服务 建模的 `requestAlgorithmMember` 时，将计算请求有效载荷校验和。

如果已定义此环境变量，它将覆盖配置文件设置 `request_checksum_calculation` 的值。

AWS_RESPONSE_CHECKSUM_VALIDATION

指定何时对响应有效载荷执行校验和验证，并提供以下选项：

- `when_supported` – （默认）当操作在其服务模型中指定 Amazon CLI 支持的响应算法时，将执行响应有效载荷校验和验证。

- when_required：当操作在其服务模型中指定 Amazon CLI 支持的响应算法，并且您在操作输出中将建模的 requestValidationModeMember 设置为 ENABLED 时，将执行响应有效载荷校验和验证。

如果已定义此环境变量，它将覆盖配置文件设置 response_checksum_validation 的值。

AWS_RETRY_MODE

指定 Amazon CLI 使用哪种重试模式。有三种重试模式可用：旧模式（默认模式）、标准模式和自适应模式。有关重试的更多信息，请参阅Amazon CLI 中的 Amazon CLI 重试次数 。

如果已定义此环境变量，它将覆盖配置文件设置 retry_mode 的值。

AWS_ROLE_ARN

指定要用于运行 Amazon CLI 命令的 IAM 角色的 Amazon Resource Name（ARN）以及 Web 身份提供者。

结合使用 AWS_WEB_IDENTITY_TOKEN_FILE 和 AWS_ROLE_SESSION_NAME 环境变量。

如果已定义此环境变量，它将覆盖配置文件设置 role_arn 的值。不能将角色会话名称指定为命令行参数。

> **Note**
>
> 此环境变量仅适用于使用 Web 身份提供者的代入角色，而不适用于常规代入角色提供商配置。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

AWS_ROLE_SESSION_NAME

指定要附加到角色会话的名称。此值在 RoleSessionName 调用 Amazon CLI 操作时将提供给 AssumeRole 参数，并成为代入角色用户 ARN 的一部分：arn:aws:sts::*123456789012*:assumed-role/*role_name*/*role_session_name*。此参数为可选参数。如果未提供此值，则将自动生成会话名称。此名称显示在与此会话关联的条目的 Amazon CloudTrail 日志中。

如果已定义此环境变量，它将覆盖配置文件设置 role_session_name 的值。

结合使用 AWS_ROLE_ARN 和 AWS_WEB_IDENTITY_TOKEN_FILE 环境变量。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

> **Note**
>
> 此环境变量仅适用于使用 Web 身份提供者的代入角色，而不适用于常规代入角色提供商配置。

AWS_SDK_UA_APP_ID

一个 Amazon Web Services 账户可以供多个客户应用程序来调用 Amazon Web Services 服务。应用程序 ID 标识哪个源应用程序使用 Amazon Web Services 服务 进行了一组调用。AmazonSDK 和服务不会使用或解释此值，除非将其显示在客户通信中。例如，此值可以包含在操作电子邮件中，以唯一标识您的哪个应用程序与通知相关联。

默认情况下，没有值。

应用程序 ID 是一个字符串，最大长度为 50 个字符。允许使用字母、数字和以下特殊字符：

```
! $ % & * + - . , ^ _ ` | ~
```

如果已定义此环境变量，它将覆盖配置文件设置 sdk_ua_app_id 的值。您不能将应用程序 ID 指定为命令行选项。

AWS_SECRET_ACCESS_KEY

指定与访问密钥关联的私有密钥。这基本上是访问密钥的“密码”。

如果已定义此环境变量，它将覆盖配置文件设置 aws_secret_access_key 的值。您不能将秘密访问密钥 ID 指定为命令行选项。

AWS_SESSION_TOKEN

指定在使用您直接从 Amazon STS 操作中检索的临时安全凭证时需要的会话令牌值。有关更多信息，请参阅 Amazon CLI 命令引用中的代入角色命令的输出部分。

如果已定义此环境变量，它将覆盖配置文件设置 aws_session_token 的值。

AWS_SHARED_CREDENTIALS_FILE

指定 Amazon CLI 用于存储访问密钥的文件的位置。默认路径为 ~/.aws/credentials。

您不能在命名配置文件设置中或使用命令行参数来指定此值。

AWS_SIGV4A_SIGNING_REGION_SET

使用逗号分隔列表指定在通过 SigV4a 进行签名时要使用的区域。如果未设置此变量，Amazon CLI 会使用 Amazon Web Services 服务 所使用的默认值。如果 Amazon Web Services 服务 没有默认值，则请求签名将使用 * 作为值，在所有区域内生效。

有关 SigV4a 的更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的 Amazon 签名版本 4。

如果已定义此环境变量，它将覆盖配置文件设置 `sigv4a_signing_region_set` 的值。

AWS_STS_REGIONAL_ENDPOINTS

指定 Amazon CLI 如何确定 Amazon CLI 客户端用于与 Amazon Security Token Service (Amazon STS) 通信的 Amazon 服务端点。Amazon CLI 版本 1 的默认值为 `legacy`。

您可以指定以下两个值之一：

- **`legacy`** – 对以下Amazon区域使用全局 STS 端点 `sts.amazonaws.com`：`ap-northeast-1`、`ap-south-1`、`ap-southeast-1`、`ap-southeast-2`、`aws-global`、`ca-central-1`、`eu-central-1`、`eu-north-1`、`eu-west-1`、`eu-west-2`、`eu-west-3`、`sa-east-1`、`us-east-1`、`us-east-2`、`us-west-1` 和 `us-west-2`。所有其他区域自动使用其各自的区域端点。
- **`regional`** – Amazon CLI 始终使用当前配置的区域的 Amazon STS 端点。例如，如果客户端配置为使用 `us-west-2`，则对 Amazon STS 进行的所有调用都针对区域端点 `sts.us-west-2.amazonaws.com` 而非全局 `sts.amazonaws.com` 端点。要在启用此设置时向全局端点发送请求，您可以将区域设置为 `aws-global`。

AWS_USE_DUALSTACK_ENDPOINT

允许使用双堆栈端点发送 Amazon 请求。要详细了解支持 IPv4 和 IPv6 流量的双堆栈端点，请参阅 Amazon Simple Storage Service 用户指南中的使用 Amazon S3 双堆栈端点。双堆栈端点适用于某些区域。如果不存在适用于服务和/或 Amazon Web Services 区域的双堆栈端点，则请求将失败。默认情况下，将禁用该功能。有关更多信息，请参阅 the section called “设置成为所有 Amazon Web Services 服务使用双堆栈端点”。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使

用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_USE_FIPS_ENDPOINT

在某些 Amazon Web Services 区域，部分 Amazon 服务提供支持美国联邦信息处理标准 (FIPS) 140-2 的端点。当 Amazon 服务支持 FIPS 时，此设置指定 Amazon CLI 应使用哪个 FIPS 端点。与标准 Amazon 端点不同，FIPS 端点使用符合 FIPS 140-2 的 TLS 软件库。与美国政府有业务来往的企业可能需要使用这些端点。有关更多信息，请参阅the section called “设置成为所有 Amazon Web Services 服务使用 FIPS 端点”。

如果启用此设置，但不存在适用于您所在 Amazon Web Services 区域中的服务的 FIPS 端点，则 Amazon 命令可能会失败。在这种情况下，请使用 `--endpoint-url` 选项手动指定要在命令中使用的端点，或者使用特定于服务的端点。

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

AWS_WEB_IDENTITY_TOKEN_FILE

指定一个文件的路径，该文件包含由身份提供者提供的 OAuth 2.0 访问令牌或 OpenID Connect ID 令牌。Amazon CLI 加载此文件的内容，并将其作为 `WebIdentityToken` 参数传递给 `AssumeRoleWithWebIdentity` 操作。

结合使用 `AWS_ROLE_ARN` 和 `AWS_ROLE_SESSION_NAME` 环境变量。

如果已定义此环境变量，它将覆盖配置文件设置 `web_identity_token_file` 的值。

有关使用 Web 身份的更多信息，请参阅 the section called “通过 Web 身份代入角色”。

> **Note**
>
> 此环境变量仅适用于使用 Web 身份提供者的代入角色，而不适用于常规代入角色提供商配置。

Amazon CLI 中的命令行选项

在 Amazon CLI 中，命令行选项是您可用来覆盖该单个命令的默认设置配置设置、任何相应的配置文件设置或环境变量设置的全局参数。虽然您可以指定要使用的配置文件，但无法使用命令行选项直接指定凭证。

主题

- 如何使用命令行选项
- Amazon CLI 支持的全局命令行选项
- 命令行选项的常见用法

如何使用命令行选项

大多数命令行选项都是简单的字符串，例如，以下示例中的配置文件名 `profile1`：

```
$ aws s3 ls --profile profile1
amzn-s3-demo-bucket1
amzn-s3-demo-bucket2
...
```

每个带参数的选项都需要一个空格或等号 (=) 将参数与选项名称分开。如果参数值为包含空格的字符串，则必须使用引号将参数引起来。有关参数类型和参数格式的详细信息，请参阅 在 Amazon CLI 中指定参数值。

Amazon CLI 支持的全局命令行选项

在 Amazon CLI 中，您可以使用以下命令行选项来覆盖该单个命令的原定设置配置设置、任何相应的配置文件设置或环境变量设置。

--ca-bundle ***<string>***

指定验证 SSL 证书时要使用的证书颁发机构 (CA) 证书捆绑包。

如果已定义，此选项将覆盖配置文件设置 `ca_bundle` 和 `AWS_CA_BUNDLE` 环境变量的值。

--cli-connect-timeout ***<integer>***

指定最大套接字连接时间（以秒为单位）。如果该值设置为零 (0)，则套接字连接将无限等待（阻塞），不会超时。默认值为 60 秒。

--cli-read-timeout *<integer>*

指定最大套接字读取时间（以秒为单位）。如果该值设置为零 (0)，则套接字读取将无限等待（阻塞），不会超时。默认值为 60 秒。

--color *<string>*

指定对彩色输出的支持。有效值包括 on、off 和 auto。原设定值为 auto。

--debug

启用调试日志记录的布尔开关。默认情况下，Amazon CLI 提供有关命令输出中命令结果的任何成功或失败的清理信息。--debug 选项提供完整的 Python 日志。这包括有关命令操作的额外 stderr 诊断信息，这些信息在排查命令提供意外结果的原因时非常有用。为了轻松查看调试日志，我们建议将日志发送到文件，这样可以更轻松地搜索信息。您可以使用以下方法之一实现这一点。

要仅发送 stderr 诊断信息，请附加 2> debug.txt，其中 debug.txt 是您要用于调试文件的名称：

```
$ aws servicename commandname options --debug 2> debug.txt
```

要同时发送输出信息和 stderr 诊断信息，请附加 &> debug.txt，其中 debug.txt 是您要用于调试文件的名称：

```
$ aws servicename commandname options --debug &> debug.txt
```

--endpoint-url *<string>*

指定要将请求发送到的 URL。对于大多数命令，Amazon CLI 会根据所选服务和指定的 Amazon 区域自动确定 URL。但是，某些命令需要您指定账户专用 URL。您还可以配置一些 Amazon 服务直接在您的私有 VPC 中托管端点（然后可能需要指定该端点）。

以下命令示例使用自定义 Amazon S3 端点 URL。

```
$ aws s3 ls --endpoint-url http://localhost:4567
```

> **端点优先级**
>
> 端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 按特定顺序检查这些端点设置，并使

用优先级最高的端点设置。有关端点优先级列表，请参阅the section called “端点配置和设置优先级”。

--no-paginate

一个布尔开关，它禁用多次调用，Amazon CLI 自动发出这些调用来接收创建输出分页的所有命令结果。这意味着只显示您的输出的第一页。

--no-sign-request

对 Amazon 服务端点的 HTTP 请求禁用签名的布尔开关。这可避免加载凭证。

--no-verify-ssl

默认情况下，Amazon CLI 在与 Amazon 服务通信时使用 SSL。对于每个 SSL 连接和调用，Amazon CLI 都会验证 SSL 证书。使用此选项会覆盖验证 SSL 证书的默认行为。

> **Warning**
>
> 此选项不是 最佳做法。如果您使用 `--no-verify-ssl`，则您的客户端和 Amazon 服务之间的流量将不再受到保护。这意味着您的流量存在安全风险，容易受到中间人攻击。如果您遇到证书问题，最好解决这些问题。有关证书故障排除步骤，请参阅 the section called “SSL 证书错误”。

--output ***<string>***

指定用于该命令的输出格式。您可以指定以下任意值：

- **`json`** – 输出采用 JSON 字符串的格式。
- **`text`** – 输出采用多个制表符分隔字符串值行的格式。这对于将输出传递到文本处理器（如 `grep`、`sed` 或 `awk`）很有用。
- **`table`** – 输出采用表格形式，使用字符 +|- 以形成单元格边框。它通常以“人性化”格式呈现信息，这种格式比其他格式更容易阅读，但从编程方面来讲不是那么有用。

--profile ***<string>***

指定用于该命令的命名配置文件。要设置其他命名配置文件，可以在 `aws configure` 命令中使用 `--profile` 选项。

```
$ aws configure --profile <profilename>
```

--query *<string>*

指定用于筛选响应数据的 JMESPath 查询。有关更多信息，请参阅 在 Amazon CLI 中筛选输出。

--region *<string>*

指定要将该命令的 Amazon 请求发送到的 Amazon 区域。有关您可以指定的所有区域的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon 区域和端点。

--v2-debug

启用 v1 到 v2 升级调试模式。在此模式下，Amazon CLI将检测在 Amazon CLI v2 中有突破性更改的更新功能的使用情况，并针对检测到的可能受影响的每种使用情况输出警告。您可以使用此模式为升级到 v2 准备环境和 Amazon CLI 使用情况。

此选项将覆盖 AWS_CLI_UPGRADE_DEBUG_MODE 环境变量。

有关 Amazon CLI v1 到 v2 升级调试模式的信息，请参阅使用升级调试模式将 Amazon CLI 版本 1 升级到 Amazon CLI 版本 2。

--version

显示正在运行的 Amazon CLI 程序的当前版本的布尔开关。

命令行选项的常见用法

常见的命令行选项用法包括在编写脚本时检查多个 Amazon 区域中的资源，以及更改输出格式使其易于阅读或使用。在以下示例中，我们对每个区域运行 describe-instances 命令，直到我们找到实例所在的区域。

```
$ aws ec2 describe-instances --output table --region us-west-1
-------------------
|DescribeInstances|
+-----------------+
$ aws ec2 describe-instances --output table --region us-west-2
-----------------------------------------------------------------------------
|                             DescribeInstances                             |
+---------------------------------------------------------------------------+
||                               Reservations                              ||
```

```
|+------------------------------------+------------------------------------+|
||  OwnerId                           |  012345678901                      ||
||  ReservationId                     |  r-abcdefgh                        ||
|+------------------------------------+------------------------------------+|
|||                               Instances                                |||
||+-------------------------+-----------------------------------------------+||
|||  AmiLaunchIndex         |  0                                            |||
|||  Architecture           |  x86_64                                       |||
...
```

在 Amazon CLI 中配置命令完成

Amazon Command Line Interface（Amazon CLI）包含一个与 Bash 兼容的命令完成功能，让您可以使用 Tab 键完成部分输入的命令。在大多数系统上，您需要手动配置此功能。

主题

- 工作原理
- 在 Linux 或 macOS 上配置命令完成
- 在 Windows 上配置命令完成

工作原理

当您部分输入命令、参数或选项时，命令完成功能会自动完成您的命令或显示建议的命令列表。要提示命令完成，请输入命令的一部分并按完成键（在大多数 Shell 中，它通常为 *Tab* 键）。

以下示例显示了可以使用命令完成的不同方法：

- 部分输入命令，然后按 *Tab* 键以显示建议的命令列表。

```
$ aws dynamodb dTAB
delete-backup                       describe-global-table
delete-item                         describe-global-table-settings
delete-table                        describe-limits
describe-backup                     describe-table
describe-continuous-backups         describe-table-replica-auto-scaling
describe-contributor-insights       describe-time-to-live
describe-endpoints
```

- 部分输入参数，然后按 *Tab* 键显示建议的参数列表。

```
$ aws dynamodb delete-table --TAB
--ca-bundle              --endpoint-url           --profile
--cli-connect-timeout    --generate-cli-skeleton  --query
--cli-input-json         --no-paginate            --region
--cli-read-timeout       --no-sign-request        --table-name
--color                  --no-verify-ssl          --version
--debug                  --output
```

- 输入参数并按 *Tab* 键以显示建议的资源值列表。此功能仅在 Amazon CLI 版本 2 中可用。

```
$ aws dynamodb delete-table --table-name TAB
Table 1                  Table 2                  Table 3
```

在 Linux 或 macOS 上配置命令完成

要在 Linux 或 macOS 上配置命令完成，您必须知道所使用的 Shell 的名称和 `aws_completer` 脚本的位置。

> **Note**
>
> 默认情况下，在运行 Amazon Linux 的 Amazon EC2 实例上自动配置和启用命令完成。

主题

- 确认完成标签的文件夹在您的路径中
- 启用命令完成
- 验证命令完成

确认完成标签的文件夹在您的路径中

要让Amazon完成标签成功运行，`aws_completer` 必须在您的 Shell 路径中。`which` 命令可以检查完成标签是否在您的路径中。

```
$ which aws_completer
/usr/local/bin/aws_completer
```

如果 which 命令找不到完成标签，则按照以下步骤将完成标签的文件夹添加到您的路径中。

步骤 1：定位Amazon完成标签

Amazon 完成标签的位置可能随所用安装方法而异。

- 程序包管理器 – pip、yum、brew 和 apt-get 等程序通常在标准路径位置安装Amazon完成标签（或其符号链接）。
  - 如果您使用没有 pip 参数的--user，则默认路径为 /usr/local/bin/aws_completer。
  - 如果您使用包含 pip 参数的--user，则默认路径为 /home/*username*/.local/bin/aws_completer。
- 捆绑安装程序 – 如果您使用捆绑安装程序，则默认路径为 /usr/local/bin/aws_completer。

如果所有其他操作都失败，您可以使用 find 命令在您的文件系统中搜索Amazon完成标签。

```
$ find / -name aws_completer
/usr/local/bin/aws_completer
```

步骤 2：识别 Shell

要识别您正在使用的 Shell，可以使用以下命令之一。

- echo $SHELL – 显示 Shell 的程序文件名称。这通常会与所使用的 Shell 的名称匹配，除非您在登录后启动了不同的 Shell。

```
$ echo $SHELL
/bin/bash
```

- ps – 显示为当前用户运行的进程。其中之一是 Shell。

```
$ ps
  PID TTY          TIME CMD
 2148 pts/1    00:00:00 bash
 8756 pts/1    00:00:00 ps
```

步骤 3：将完成标签添加到您的路径中

1. 在您的用户文件夹中查找 Shell 的配置文件脚本。

```
$ ls -a ~/
```

```
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

- Bash– .bash_profile、.profile 或 .bash_login
- Zsh– .zshrc
- Tcsh – .tcshrc、.cshrc 或 .login

2. 在配置文件脚本末尾添加与以下示例类似的导出命令。将 */usr/local/bin/* 替换为您在上一部分中找到的文件夹。

```
export PATH=/usr/local/bin/:$PATH
```

3. 将配置文件重新加载到当前会话中，以使更改生效。将 *.bash_profile* 替换为您在第一部分中找到的 shell 脚本的名称。

```
$ source ~/.bash_profile
```

启用命令完成

确认完成标签在您的路径中后，通过运行正在使用的 Shell 的相应命令来启用命令完成。您可以将命令添加到 Shell 的配置文件中，以便在每次打开一个新 Shell 时运行它。在每个命令中，将 */usr/local/bin/* 路径替换为 确认完成标签的文件夹在您的路径中 中在您的系统上找到的那个。

- **bash** – 使用内置命令 complete。

```
$ complete -C '/usr/local/bin/aws_completer' aws
```

  将之前的命令添加到 ~/.bashrc 中，以便在每次打开一个新外壳程序时运行它。您的 ~/.bash_profile 应指定源 ~/.bashrc，以确保该命令也在登录 Shell 中运行。

- **zsh** – 要运行命令完成功能，您需要在 ~/.zshrc 配置文件脚本的末尾添加以下自动加载行来运行 bashcompinit。

```
$ autoload bashcompinit && bashcompinit
$ autoload -Uz compinit && compinit
```

  要启用命令完成，请使用内置命令 complete。

```
$ complete -C '/usr/local/bin/aws_completer' aws
```

将之前的命令添加到 ~/.zshrc 中，以便在每次打开一个新外壳程序时运行它。

- **tcsh** – tcsh 的完成采用字类型和样式来定义完成行为。

```
> complete aws 'p/*/`aws_completer`/'
```

将之前的命令添加到 ~/.tcshrc 中，以便在每次打开一个新外壳程序时运行它。

启用命令完成后，验证命令完成 正在工作。

验证命令完成

启用命令完成后，重新加载 Shell，输入部分命令并按 Tab 查看可用命令。

```
$ aws sTAB
s3              ses             sqs             sts             swf
s3api           sns             storagegateway  support
```

在 Windows 上配置命令完成

> **Note**
>
> 有关 PowerShell 如何处理其完成情况（包括各种完成密钥）的信息，请参阅 Microsoft PowerShell Docs 中的 about_Tab_Expansion。

要在 Windows 上启用 PowerShell 的命令完成，请在 PowerShell 中完成以下步骤。

1. 使用以下命令打开你的 $PROFILE。

```
PS C:\> Notepad $PROFILE
```

如果没有 $PROFILE，请使用以下命令创建用户配置文件。

```
PS C:\> if (!(Test-Path -Path $PROFILE ))
{ New-Item -Type File -Path $PROFILE -Force }
```

有关 PowerShell 配置文件的详细信息，请参阅 Microsoft Docs 网站上的如何在 Windows PowerShell ISE 中使用配置文件。

2. 要启用命令完成，请将以下代码块添加到您的配置文件中，保存，然后关闭文件。

```
Register-ArgumentCompleter -Native -CommandName aws -ScriptBlock {
    param($commandName, $wordToComplete, $cursorPosition)
        $env:COMP_LINE=$wordToComplete
        if ($env:COMP_LINE.Length -lt $cursorPosition){
            $env:COMP_LINE=$env:COMP_LINE + " "
        }
        $env:COMP_POINT=$cursorPosition
        aws_completer.exe | ForEach-Object {
            [System.Management.Automation.CompletionResult]::new($_, $_,
 'ParameterValue', $_)
        }
        Remove-Item Env:\COMP_LINE
        Remove-Item Env:\COMP_POINT
}
```

3. 启用命令完成功能后，重新加载 Shell，输入命令的一部分并按 Tab 可循环浏览可用命令。

```
$ aws sTab
```

```
$ aws s3
```

要查看完成后可用的所有命令，请输入命令的一部分并按 Ctrl + 空格键。

```
$ aws sCtrl + Space
s3              ses             sqs             sts             swf
s3api           sns             storagegateway  support
```

Amazon CLI 中的 Amazon CLI 重试次数

本主题介绍在 Amazon CLI 中，可能由于意外问题而导致 Amazon 服务调用失败。这些调用可能发生在服务器端，也可能是由于您尝试调用的 Amazon 服务存在速率限制而失败。这些类型的故障通常不需要特殊处理，并且通常在短暂的等待时间段后会自动重新发出调用。Amazon CLI 提供了许多功能，在出现这些类型的错误或异常的情况下，帮助重试对 Amazon 服务进行的客户端调用。

主题

- 可用重试模式
- 配置重试模式
- 查看重试日志

可用重试模式

Amazon CLI 有多种模式可供选择，具体取决于您的版本：

- 标准重试模式
- 传统重试模式（默认）
- 自适应重试模式

标准重试模式

Standard 模式是一组跨 Amazon SDK 的标准重试规则，其功能比传统模式更多，并且是 Amazon CLI 的默认模式。标准模式为 Amazon CLI 版本 2 而创建，并已反向移植到 Amazon CLI 版本 1。标准模式的功能包括：

- 最大重试次数的默认值为 2，总共可发出 3 次调用尝试。此值可通过 max_attempts 配置参数覆盖。
- 对以下更加广泛的错误/异常列表重试操作：
  - 瞬时错误/异常
    - RequestTimeout
    - RequestTimeoutException
    - PriorRequestNotComplete
    - ConnectionError
    - HTTPClientError
  - 服务端限制错误和异常：
    - Throttling
    - ThrottlingException
    - ThrottledException
    - RequestThrottledException
    - TooManyRequestsException

    - ProvisionedThroughputExceededException
    - TransactionInProgressException
    - RequestLimitExceeded
    - BandwidthLimitExceeded
    - LimitExceededException
    - RequestThrottled
    - SlowDown
    - EC2ThrottledException

- 对非描述性的瞬时错误代码进行重试。具体来说，这些 HTTP 状态代码包括：500、502、503、504。
- 任何重试都将包含基准因子为 2 的指数回退，最长回退时间为 20 秒。

传统重试模式（默认）

Legacy 模式使用旧的重试处理程序，其功能有限，其中包括：

- 最大重试次数的默认值为 4，总共可发出 5 次调用尝试。此值可通过 max_attempts 配置参数覆盖。
- DynamoDB 的最大重试次数的原定设置值为 9，总共可发出 10 次调用尝试。此值可通过 max_attempts 配置参数覆盖。
- 重试以下有限数量的错误/异常：
    - 常规套接字/连接错误：
        - ConnectionError
        - ConnectionClosedError
        - ReadTimeoutError
        - EndpointConnectionError
    - 服务端限制错误和异常：
        - Throttling
        - ThrottlingException
        - ThrottledException
        - RequestThrottledException

    - `ProvisionedThroughputExceededException`
- 对多个 HTTP 状态代码执行重试操作，包括 429、500、502、503、504 和 509。
- 任何重试都将包含基准因子为 2 的指数回退。

自适应重试模式

> **Warning**
>
> 自适应模式是一种试验模式，在特征和行为方面可能会发生变化。

`Adaptive` 重试模式是一种试验性重试模式，包括标准模式的所有功能。除了标准模式功能外，自适应模式还通过使用令牌存储桶和速率限制变量引入了客户端速率限制，这些变量会随着每次重试而动态更新。此模式为客户端重试提供了灵活性，能够适应Amazon服务的错误/异常状态响应。

对于每个新的重试，自适应模式都会根据 Amazon 服务响应中显示的错误、异常或 HTTP 状态代码修改速率限制变量。然后，使用这些速率限制变量来计算客户端的新调用速率。对于Amazon服务的每个异常/错误或不成功 HTTP 响应（在上面的列表中提供），都会在重试时更新速率限制变量，直到达到成功、令牌存储桶用尽或达到配置的最大尝试次数值。

配置重试模式

Amazon CLI 包括各种重试配置，以及创建客户端对象时要考虑的配置方法。

可用配置方法

在 Amazon CLI 中，用户可以通过以下方式配置重试：

- 环境变量
- Amazon CLI 配置文件

用户可以自定义以下重试选项：

- 重试模式 - 指定 Amazon CLI 使用的重试模式。如上所述，有三种重试模式可用：传统模式、标准模式和自适应模式。Amazon CLI 版本 1 的原定设置值为传统模式。
- 最大尝试次数 - 指定 Amazon CLI 重试处理程序使用的最大重试次数值，其中初始调用会计入您提供的值。默认值基于您的重试模式。

在环境变量中定义重试配置

要为 Amazon CLI 定义您的重试配置，请更新操作系统的环境变量。

重试环境变量包括：

- `AWS_RETRY_MODE`
- `AWS_MAX_ATTEMPTS`

有关环境变量的更多信息，请参阅为 Amazon CLI 配置环境变量。

查看重试日志

Amazon CLI 使用 Boto3 的重试方法和日志记录。您可以对任何命令使用 `--debug` 选项来接收调试日志。有关如何使用 `--debug` 选项的更多信息，请参阅Amazon CLI 中的命令行选项。

如果您在调试日志中搜索“retry”，将会找到所需的重试信息。重试操作的客户端日志条目取决于您启用的重试模式。

传统模式：

重试消息是由 botocore.retryhandler 生成的。您将看到以下三个消息之一：

- `No retry needed`
- `Retry needed, action of: <action_name>`
- `Reached the maximum number of retry attempts: <attempt_number>`

标准模式或自适应模式：

重试消息是由 botocore.retries.standard 生成的。您将看到以下三个消息之一：

- `No retrying request`
- `Retry needed, retrying request after delay of: <delay_value>`
- `Retry needed but retry quota reached, not retrying request`

有关 botocore 重试的完整定义文件，请参阅 botocore GitHub 存储库 上的 _retry.json。

在 Amazon CLI 中使用 HTTP 代理

要通过代理服务器访问 Amazon，您可以使用代理服务器使用的 DNS 域名或 IP 地址和端口号配置 HTTP_PROXY 和 HTTPS_PROXY 环境变量。

主题

- 使用示例
- 向代理进行身份验证
- 对 Amazon EC2 实例使用代理
- 故障排除

使用示例

> **Note**
>
> 以下示例显示了全部使用大写字母的环境变量名称。但是，如果使用不同的大小写指定一个变量两次，则优先使用小写字母。建议您只定义每个变量一次，以避免系统混淆和意外行为。

以下示例显示如何使用代理的显式 IP 地址或解析为代理 IP 地址的 DNS 名称。两种情况都可以后跟冒号和应将查询发送到的端口号。

Linux or macOS

```
$ export HTTP_PROXY=http://10.15.20.25:1234
$ export HTTP_PROXY=http://proxy.example.com:1234
$ export HTTPS_PROXY=http://10.15.20.25:5678
$ export HTTPS_PROXY=http://proxy.example.com:5678
```

Windows Command Prompt

为所有会话设置

```
C:\> setx HTTP_PROXY http://10.15.20.25:1234
C:\> setx HTTP_PROXY http://proxy.example.com:1234
C:\> setx HTTPS_PROXY http://10.15.20.25:5678
C:\> setx HTTPS_PROXY http://proxy.example.com:5678
```

使用 `setx` 设置环境变量会更改当前命令提示符会话和运行该命令后创建的所有命令提示符会话中使用的值。它不 影响在运行该命令时已经运行的其他命令 shell。

仅为当前会话设置

使用 `set` 设置环境变量会更改使用的值，直到当前命令提示符会话结束，或者直到您将该变量设置为其他值。

```
C:\> set HTTP_PROXY=http://10.15.20.25:1234
C:\> set HTTP_PROXY=http://proxy.example.com:1234
C:\> set HTTPS_PROXY=http://10.15.20.25:5678
C:\> set HTTPS_PROXY=http://proxy.example.com:5678
```

向代理进行身份验证

> **Note**
>
> Amazon CLI 不支持 NTLM 代理。如果使用 NTLM 或 Kerberos 协议代理，则可以通过身份验证代理（如 Cntlm）进行连接。

Amazon CLI 支持 HTTP 基本身份验证。在代理 URL 中指定用户名和密码，如下所示。

Linux or macOS

```
$ export HTTP_PROXY=http://username:password@proxy.example.com:1234
$ export HTTPS_PROXY=http://username:password@proxy.example.com:5678
```

Windows Command Prompt

为所有会话设置

```
C:\> setx HTTP_PROXY http://username:password@proxy.example.com:1234
C:\> setx HTTPS_PROXY http://username:password@proxy.example.com:5678
```

仅为当前会话设置

```
C:\> set HTTP_PROXY=http://username:password@proxy.example.com:1234
```

```
C:\> set HTTPS_PROXY=http://username:password@proxy.example.com:5678
```

对 Amazon EC2 实例使用代理

如果是在使用附加 IAM 角色启动的 Amazon EC2 实例上配置代理，请确保排除用于访问实例元数据的地址。为此，请将 NO_PROXY 环境变量设置为实例元数据服务的 IP 地址 169.254.169.254。该地址保持不变。

Linux or macOS

```
$ export NO_PROXY=169.254.169.254
```

Windows Command Prompt

为所有会话设置

```
C:\> setx NO_PROXY 169.254.169.254
```

仅为当前会话设置

```
C:\> set NO_PROXY=169.254.169.254
```

故障排除

如果您遇到有关 Amazon CLI 的问题，请参阅the section called “排查错误”以了解故障排除步骤。有关相关性最高的故障排除步骤，请参阅the section called “SSL 证书错误”。

在 Amazon CLI 中使用端点

要通过编程方式连接到某个 Amazon Web Services 服务，您需要使用端点。端点是作为 Amazon Web 服务的入口点的 URL。Amazon Command Line Interface (Amazon CLI) 会自动为 Amazon Web Services 区域中的每项服务使用默认端点，但您可以应 API 请求指定备用端点。

端点主题

- 为单个命令设置端点
- 为所有 Amazon Web Services 服务设置全局端点

- 设置成为所有 Amazon Web Services 服务使用 FIPS 端点
- 设置成为所有 Amazon Web Services 服务使用双堆栈端点
- 设置特定于服务的端点
  - 特定于服务的端点：环境变量
  - 特定于服务的端点：共享 config 文件
  - 特定于服务的端点：特定于服务的标识符列表
- 基于账户的端点
- 端点配置和设置优先级

为单个命令设置端点

要覆盖单个命令的任何端点设置或环境变量，请使用 `--endpoint-url` 命令行选项。以下命令示例使用自定义 Amazon S3 端点 URL。

```
$ aws s3 ls --endpoint-url http://localhost:4567
```

为所有 Amazon Web Services 服务设置全局端点

要将所有服务的请求路由到自定义端点 URL，请使用以下设置之一：

- 环境变量：
  - `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` – 忽略已配置的端点 URL。

    Linux or macOS

    ```
    $ export AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
    ```

    Windows Command Prompt

    为所有会话设置

    ```
    C:\> setx AWS_IGNORE_CONFIGURED_ENDPOINT_URLS true
    ```

    仅为当前会话设置

    ```
    C:\> set AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
    ```

PowerShell

```
PS C:\> $Env:AWS_IGNORE_CONFIGURED_ENDPOINT_URLS="true"
```

- AWS_ENDPOINT_URL – 设置全局端点 URL。

  Linux or macOS

  ```
  $ export AWS_ENDPOINT_URL=http://localhost:4567
  ```

  Windows Command Prompt

  为所有会话设置

  ```
  C:\> setx AWS_ENDPOINT_URL http://localhost:4567
  ```

  仅为当前会话设置

  ```
  C:\> set AWS_ENDPOINT_URL=http://localhost:4567
  ```

  PowerShell

  ```
  PS C:\> $Env:AWS_ENDPOINT_URL="http://localhost:4567"
  ```

- config 文件：
  - ignore_configure_endpoint_urls – 忽略已配置的端点 URL。

    ```
    ignore_configure_endpoint_urls = true
    ```

  - endpoint_url – 设置全局端点 URL。

    ```
    endpoint_url = http://localhost:4567
    ```

特定于服务的端点和 `--endpoint-url` 命令行选项会覆盖所有全局端点。

设置成为所有 Amazon Web Services 服务使用 FIPS 端点

要路由所有服务的请求以使用 FIPS 端点，请使用以下设置之一：

- AWS_USE_FIPS_ENDPOINT 环境变量。

  Linux or macOS

  ```
  $ export AWS_USE_FIPS_ENDPOINT=true
  ```

  Windows Command Prompt

  为所有会话设置

  ```
  C:\> setx AWS_USE_FIPS_ENDPOINT true
  ```

  仅为当前会话设置

  ```
  C:\> set AWS_USE_FIPS_ENDPOINT=true
  ```

  PowerShell

  ```
  PS C:\> $Env:AWS_USE_FIPS_ENDPOINT="true"
  ```

- use_fips_endpoint 文件设置。

  ```
  use_fips_endpoint = true
  ```

在某些 Amazon Web Services 区域，部分 Amazon 服务提供支持美国联邦信息处理标准 (FIPS) 140-2 的端点。当 Amazon 服务支持 FIPS 时，此设置指定 Amazon CLI 应使用哪个 FIPS 端点。与标准 Amazon 端点不同，FIPS 端点使用符合 FIPS 140-2 的 TLS 软件库。与美国政府有业务来往的企业可能需要使用这些端点。

如果启用此设置，但不存在适用于您所在 Amazon Web Services 区域中的服务的 FIPS 端点，则 Amazon 命令可能会失败。在这种情况下，请使用 --endpoint-url 选项手动指定要在命令中使用的端点，或者使用特定于服务的端点。

有关按 Amazon Web Services 区域指定 FIPS 端点的更多信息，请参阅按服务列出的 FIPS 端点。

设置成为所有 Amazon Web Services 服务使用双堆栈端点

要路由所有服务的请求以在可用时使用双堆栈端点，请使用以下设置之一：

- AWS_USE_DUALSTACK_ENDPOINT 环境变量。

Linux or macOS

```
$ export AWS_USE_DUALSTACK_ENDPOINT=true
```

Windows Command Prompt

为所有会话设置

```
C:\> setx AWS_USE_DUALSTACK_ENDPOINT true
```

仅为当前会话设置

```
C:\> set AWS_USE_DUALSTACK_ENDPOINT=true
```

PowerShell

```
PS C:\> $Env:AWS_USE_DUALSTACK_ENDPOINT="true"
```

- use_dualstack_endpoint 文件设置。

```
use_dualstack_endpoint = true
```

允许使用双堆栈端点发送 Amazon 请求。要详细了解支持 IPv4 和 IPv6 流量的双堆栈端点，请参阅 Amazon Simple Storage Service 用户指南中的使用 Amazon S3 双堆栈端点。双堆栈端点适用于某些区域。如果不存在适用于服务和/或 Amazon Web Services 区域的双堆栈端点，则请求将失败。默认情况下，将禁用该功能。

设置特定于服务的端点

特定于服务的端点配置提供了一个选项，可使用您应 Amazon CLI 请求选择的永久端点。这些设置可以灵活地支持本地端点、VPC 端点和第三方本地 Amazon 开发环境。不同的端点可分别用于测试环境和生产环境。您可以为个别 Amazon Web Services 服务指定端点 URL。

可通过以下方式指定特定于服务的端点：

- 单个命令的命令行选项 --endpoint-url。
- 环境变量：

- AWS_IGNORE_CONFIGURED_ENDPOINT_URLS – 忽略所有已配置的端点 URL，除非在命令行中指定。
- AWS_ENDPOINT_URL_<SERVICE> – 指定用于特定服务的自定义端点，在该服务中 <SERVICE> 替换为 Amazon Web Services 服务标识符。有关所有特定于服务的变量，请参阅the section called “特定于服务的标识符列表”。

- config 文件：
  - ignore_configure_endpoint_urls – 忽略所有已配置的端点 URL，除非使用环境变量或在命令行中指定。
  - config 文件的 services 节与 endpoint_url 文件设置相结合。

特定于服务的端点主题：

- 特定于服务的端点：环境变量
- 特定于服务的端点：共享 config 文件
- 特定于服务的端点：特定于服务的标识符列表

特定于服务的端点：环境变量

环境变量会覆盖配置文件中的设置，但不会覆盖命令行上指定的选项。如果您希望所有配置文件在设备上使用相同的端点，请使用环境变量。

以下是特定于服务的环境变量：

- AWS_IGNORE_CONFIGURED_ENDPOINT_URLS – 忽略所有已配置的端点 URL，除非在命令行中指定。

  Linux or macOS

  ```
  $ export AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
  ```

  Windows Command Prompt

  为所有会话设置

  ```
  C:\> setx AWS_IGNORE_CONFIGURED_ENDPOINT_URLS true
  ```

  仅为当前会话设置

```
C:\> set AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
```

PowerShell

```
PS C:\> $Env:AWS_IGNORE_CONFIGURED_ENDPOINT_URLS="true"
```

- AWS_ENDPOINT_URL_<SERVICE> – 指定用于特定服务的自定义端点，在该服务中 <SERVICE> 替换为 Amazon Web Services 服务标识符。有关所有特定于服务的变量，请参阅the section called “特定于服务的标识符列表”。

  以下环境变量示例设置 Amazon Elastic Beanstalk 的端点。

  Linux or macOS

  ```
  $ export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:4567
  ```

  Windows Command Prompt

  为所有会话设置

  ```
  C:\> setx AWS_ENDPOINT_URL_ELASTIC_BEANSTALK http://localhost:4567
  ```

  仅为当前会话设置

  ```
  C:\> set AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:4567
  ```

  PowerShell

  ```
  PS C:\> $Env:AWS_ENDPOINT_URL_ELASTIC_BEANSTALK="http://localhost:4567"
  ```

有关设置环境变量的更多信息，请参阅the section called “环境变量”。

特定于服务的端点：共享 `config` 文件

在共享 `config` 文件中，`endpoint_url` 用在多个节中。要设置特定于服务的端点，请使用嵌套在 `services` 节内服务标识符密钥下的 `endpoint_url` 设置。有关在共享 `config` 文件中定义 `services` 节的详细信息，请参阅 the section called “`services`”。

以下示例使用 services 节为 Amazon S3 配置服务特定的端点 URL 和用于其他所有服务的自定义全局端点：

```
[profile dev1]
endpoint_url = http://localhost:1234
services = s3-specific

[services testing-s3]
s3 =
  endpoint_url = http://localhost:4567
```

单个配置文件可以为多个服务配置端点。以下示例在同一配置文件中为 Amazon S3 和 Amazon Elastic Beanstalk 设置服务特定的端点 URL。

有关 services 节中要使用的所有服务标识符密钥的列表，请参阅特定于服务的标识符列表。

```
[profile dev1]
services = testing-s3-and-eb

[services testing-s3-and-eb]
s3 =
  endpoint_url = http://localhost:4567
elastic_beanstalk =
  endpoint_url = http://localhost:8000
```

“服务配置”节可以在多个配置文件中使用。以下示例有两个配置文件使用相同的 services 定义：

```
[profile dev1]
output = json
services = testing-s3

[profile dev2]
output = text
services = testing-s3

[services testing-s3]
s3 =
  endpoint_url = https://localhost:4567
```

特定于服务的端点：特定于服务的标识符列表

Amazon Web Services 服务标识符基于 API 模型的 `serviceId`，不过需要将所有空格替换为下划线，并将所有字母小写。

下表列出了所有特定于服务的标识符、`config` 文件密钥和环境变量。

基于账户的端点

可以通过以下方式指定基于账户的端点：

- 环境变量
  - AWS_ACCOUNT_ID：指定要用于调用受支持的 Amazon Web Services 服务的基于 Amazon 账户的端点 ID。

    Linux or macOS

    ```
    $ export AWS_ACCOUNT_ID=<account-id>
    ```

    Windows Command Prompt

    为所有会话设置

    ```
    C:\> setx AWS_ACCOUNT_ID <account-id>
    ```

    仅为当前会话设置

    ```
    C:\> set AWS_ACCOUNT_ID=<account-id>
    ```

    PowerShell

    ```
    PS C:\> $Env:AWS_ACCOUNT_ID="<account-id>"
    ```

  - AWS_ACCOUNT_ID_ENDPOINT_MODE：指定是否使用基于 Amazon 账户的端点 ID 来调用受支持的 Amazon Web Services 服务。可以设置为 `preferred`、`disabled` 或 `required`。默认值为 `preferred`。

    Linux or macOS

    ```
    $ export AWS_ACCOUNT_ID_ENDPOINT_MODE=preferred
    ```

Windows Command Prompt

为所有会话设置

```
C:\> setx AWS_ACCOUNT_ID_ENDPOINT_MODE preferred
```

仅为当前会话设置

```
C:\> set AWS_ACCOUNT_ID_ENDPOINT_MODE=preferred
```

PowerShell

```
PS C:\> $Env:AWS_ACCOUNT_ID_ENDPOINT_MODE="preferred"
```

- config 文件：
  - aws_account_id：指定要用于调用受支持的 Amazon Web Services 服务的基于 Amazon 账户的端点 ID。

    ```
    aws_account_id = <account-id>
    ```

  - account_id_endpoint_mode：指定是否使用基于 Amazon 账户的端点 ID 来调用受支持的 Amazon Web Services 服务。可以设置为 preferred、disabled 或 required。首选默认值。

    ```
    account_id_endpoint_mode = preferred
    ```

基于账户的端点通过使用 Amazon 账户 ID 来简化对支持此功能的服务发出的 Amazon Web Services 服务请求的路由，有助于确保高性能和可扩展性。当您使用凭证提供商和支持基于账户的端点的服务时，Amazon CLI 会自动构造和使用基于账户的端点，而不是区域端点。

基于账户的端点使用以下格式，其中 <account-id> 替换为您的 Amazon 账户 ID，而 <region> 替换为您的 Amazon Web Services 区域：

```
https://<account-id>.myservice.<region>.amazonaws.com
```

默认情况下，在 Amazon CLI 中，基于账户的端点模式设置为 preferred。

端点配置和设置优先级

端点配置设置位于多个位置（例如，系统变量或用户环境变量、本地 Amazon 配置文件），或者在命令行上显式声明为参数。Amazon CLI 端点配置设置的优先顺序如下：

1. `--endpoint-url` 命令行选项。
2. 如果启用，则 `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` 全局端点环境变量或配置文件设置 `ignore_configure_endpoint_urls` 将忽略自定义端点。
3. 由特定于服务的环境变量 `AWS_ENDPOINT_URL_<SERVICE>` 提供的值，例如 AWS_ENDPOINT_URL_DYNAMODB。
4. `AWS_USE_DUALSTACK_ENDPOINT`、`AWS_USE_FIPS_ENDPOINT` 和 `AWS_ENDPOINT_URL` 环境变量提供的值。
5. `AWS_ACCOUNT_ID_ENDPOINT_MODE` 环境变量使用 `AWS_ACCOUNT_ID` 环境变量或 `aws_account_id` 设置中的账户 ID 设置为 `preferred` 或 `required`。
6. 共享 `config` 文件 `services` 节中的 `endpoint_url` 设置提供的特定于服务的端点值。
7. 共享 `config` 文件的 `profile` 中的 `endpoint_url` 设置提供的值。
8. `use_dualstack_endpoint`、`use_fips_endpoint` 和 `endpoint_url` 设置。
9. `account_id_endpoint_mode` 设置使用 `AWS_ACCOUNT_ID` 环境变量或 `aws_account_id` 设置中的账户 ID 设置为 `preferred` 或 `required`。
10. 最后使用相应的 Amazon Web Services 服务 的任何默认端点 URL。有关每个区域可用的标准服务端点的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon 区域和端点。

Amazon CLI 身份验证和访问凭证

在使用 Amazon 服务进行开发时，必须确定 Amazon CLI 如何使用 Amazon 进行身份验证。要为 Amazon CLI 配置用于编程访问的凭证，请选择下列选项之一。这些选项按推荐顺序排列。

身份验证类型	用途	说明
IAM 用户短期凭证	使用 IAM 用户短期凭证，它比长期凭证更安全。如果您的凭证遭到泄露，则在凭证过期之前的使用时间有限。	the section called “短期凭证”
Amazon EC2 实例上的 IAM 。	通过分配给 Amazon EC2 实例的角色，使用 Amazon EC2 实例元数据查询临时凭证。	the section called “Amazon EC2 元数据”
代入角色以获得权限	将另一个凭证方法配对并代入一个角色以临时访问您的用户可能无法访问的 Amazon Web Services 服务。	the section called “IAM 角色”
IAM 用户长期凭证	（不推荐）使用不会过期的长期凭证。	the section called “IAM 用户”
IAM 外部存储	（不推荐）将另一个凭证方法配对，但存储位于 Amazon CLI 外部的凭证值。此方法的安全性取决于存储凭证的外部位置。	the section called “外部凭证”

配置和凭证优先顺序

凭证和配置设置位于不同位置（例如，系统或用户环境变量、本地 Amazon 配置文件）或在命令行上显式声明为参数。某些位置优先于其他位置。Amazon CLI 凭证和配置设置的优先顺序如下：

1. 命令行选项 – 覆盖任何其他位置的设置，例如 `--region`、`--output` 和 `--profile` 参数。
2. 环境变量 – 您可以在系统的环境变量中存储值。

3. 代入角色：通过配置或 `assume-role` 命令代入 IAM 角色的权限。
4. 使用 Web 身份代入角色：通过配置或 `assume-role-with-web-identity` 命令使用 Web 身份代入 IAM 角色的权限。
5. 凭证文件 – 在运行命令 `aws configure` 时，将更新 `credentials` 和 `config` 文件。`credentials` 文件位于 `~/.aws/credentials`（在 Linux 或 macOS 上）或 `C:\Users\USERNAME\.aws\credentials`（在 Windows 上）。
6. 自定义流程 – 从外部来源获取您的凭证。
7. 配置文件 – 在运行命令 `aws configure` 时，将更新 `credentials` 和 `config` 文件。`config` 文件位于 `~/.aws/config`（在 Linux 或 macOS 上）或 `C:\Users\USERNAME\.aws\config`（在 Windows 上）。
8. 容器凭证 – 您可以将 IAM 角色与每个 Amazon Elastic Container Service (Amazon ECS) 作业定义关联。之后，该任务的容器就可以使用该角色的临时凭证。有关更多信息，请参阅 Amazon Elastic Container Service 开发人员指南中的任务的 IAM 角色。
9. Amazon EC2 实例配置文件凭证 – 您可以将 IAM 角色与每个 Amazon Elastic Compute Cloud (Amazon EC2) 实例关联。之后，在该实例上运行的代码就可以使用该角色的临时凭证。凭证通过 Amazon EC2 元数据服务提供。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 角色和《IAM 用户指南》中的使用实例配置文件。

此部分中的其他主题

- the section called “短期凭证”
- the section called “IAM 角色”
- the section called “IAM 用户”
- the section called “Amazon EC2 元数据”
- the section called “外部凭证”

在 Amazon CLI 中使用短期凭证进行身份验证

建议将您的 SDK 或工具配置为使用带延长会话持续时间选项的 IAM Identity Center 身份验证。但是，您可以复制和使用 Amazon 访问门户中提供的临时凭证。在这些临时凭证过期后，将需要复制新凭证。您可以在配置文件中使用临时凭证，也可以将其用作系统属性和环境变量的值。

1. 登录到 Amazon 访问门户。
2. 按照以下说明操作，从 Amazon 访问门户中复制 IAM 角色凭证。

1. 对于链接的说明中的步骤 2，选择可授予访问权限以满足您的开发需求的 Amazon 账户和 IAM 角色名称。此角色的名称通常类似于 PowerUserAccess 或 Developer。
2. 对于步骤 4，选择将配置文件添加到您的 Amazon 凭证文件选项并复制内容。

3. 创建或打开共享 `credentials` 文件。此文件在 Linux 和 macOS 系统上为 `~/.aws/credentials`，在 Windows 上为 `%USERPROFILE%\.aws\credentials`。有关更多信息，请参阅 the section called “配置设置”。
4. 将以下文本添加到共享 `credentials` 文件中。将示例值替换为您复制的凭证。

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token =
 IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRINGEXAMPLE
```

5. 将您的首选默认区域和格式添加到共享 `config` 文件中。

```
[default]
region=us-west-2
output=json

[profile user1]
region=us-east-1
output=text
```

当 SDK 创建服务客户端时，它将访问这些临时凭证并将它们用于每个请求。在步骤 2a 中选择的 IAM 角色的设置决定了临时凭证的有效时间。最长持续时间为 12 小时。

每次您的凭证到期时都重复这些步骤。

在 Amazon CLI 中使用 IAM 角色

Amazon Identity and Access Management（IAM）角色是一种授权工具，可让用户获得额外（或不同）的权限或者获取使用其他 Amazon 账户执行操作的权限。

主题

- 先决条件
- IAM 角色的用法概览

- 配置和使用角色
- 使用多重验证
- 跨账户角色和外部 ID
- 指定角色会话名称以便于审核
- 通过 Web 身份代入角色
- 清除缓存的凭证

先决条件

要运行 iam 命令，您需要安装和配置 Amazon CLI。这包括设置已配置的配置文件，因为假设一个角色与其他凭证方法配对。有关更多信息，请参阅安装 Amazon CLI。

IAM 角色的用法概览

通过在 ~/.aws/config 文件中为某个 IAM 角色定义配置文件，您可以配置 Amazon Command Line Interface (Amazon CLI) 以使用该 IAM 角色。

以下示例显示了一个名为 marketingadmin 的角色配置文件。如果使用 --profile marketingadmin（或使用 AWS_PROFILE 环境变量指定它）运行命令，则 Amazon CLI 使用在单独的配置文件 user1 中定义的凭证代入 Amazon Resource Name (ARN) 为 arn:aws:iam::*123456789012*:role/*marketingadminrole* 的角色。您可以运行分配给该角色的权限所允许的任何操作。

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
source_profile = user1
```

然后，您可以指定一个指向单独的命名配置文件的 source_profile，此配置文件包含用户凭证及使用该角色的权限。在上一个示例中，marketingadmin 配置文件使用 user1 配置文件中的凭证。当您指定某个 Amazon CLI 命令将使用配置文件 marketingadmin 时，Amazon CLI 会自动查找链接的 user1 配置文件的凭证，并使用它们为指定的 IAM 角色请求临时凭证。CLI 在后台使用 sts:AssumeRole 操作来完成该操作。然后，使用这些临时凭证来运行请求的 Amazon CLI 命令。指定的角色必须附加有允许运行请求的 Amazon CLI 命令的 IAM 权限策略。

要从 Amazon Elastic Compute Cloud (Amazon EC2) 实例或 Amazon Elastic Container Service (Amazon ECS) 容器中运行 Amazon CLI 命令，您可以使用附加到实例配置文件或容器的 IAM 角色。如果未指定配置文件或未设置环境变量，则将直接使用该角色。这让您能够避免在实例上存储

长时间生存的访问密钥。您也可以使用这些实例或容器角色仅获取其他角色的凭证。为此，请使用 `credential_source`（而不是 `source_profile`）指定如何查找凭证。`credential_source` 属性支持以下值：

- `Environment` – 从环境变量检索源凭证。
- `Ec2InstanceMetadata` – 使用附加到 Amazon EC2 实例配置文件的 IAM 角色。
- `EcsContainer` – 使用附加到 Amazon ECS 容器的 IAM 角色。

以下示例显示通过引用 Amazon EC2 实例配置文件来使用同一个 `marketingadminrole` 角色。

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
credential_source = Ec2InstanceMetadata
```

当您调用角色时，您可以要求使用其他选项，例如，使用多重身份验证和外部 ID（供第三方公司用于访问其客户的资源）。也可以指定更易于在 Amazon CloudTrail 日志中进行审核的唯一角色会话名称。

配置和使用角色

在使用指定 IAM 角色的配置文件运行命令时，Amazon CLI 将使用源配置文件的凭证调用 Amazon Security Token Service (Amazon STS) 并为指定角色请求临时凭证。源配置文件中的用户必须具有为指定配置文件中的角色调用 `sts:assume-role` 的权限。该角色必须具有允许源配置文件中的用户使用角色的信任关系。检索角色的临时凭证然后使用临时凭证的过程通常称为代入角色。

您可以通过执行《Amazon Identity and Access Management 用户指南》中的创建向 IAM 用户委派权限的角色下的过程，在 IAM 中创建一个您希望用户代入的具有该权限的角色。如果该角色和源配置文件的 用户在同一个账户中，在配置角色的信任关系时，您可以输入自己的账户 ID。

在创建角色后，请修改信任关系以允许 用户担任该角色。

以下示例显示了一个可附加到角色的信任策略。该策略允许账户 123456789012 中的任何用户代入该角色，前提是该账户的管理员向该用户显式授予了 `sts:AssumeRole` 权限。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

信任策略不会实际授予权限。账户管理员必须通过附加具有适当权限的策略才能将代入角色的权限委派给各个用户。以下示例显示了一个可附加到用户的策略，该策略仅允许用户代入 `marketingadminrole` 角色。有关授予用户代入角色的访问权限的更多信息，请参阅 IAM 用户指南中的向用户授予切换角色的权限。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/marketingadminrole"
    }
  ]
}
```

用户无需拥有附加权限，即可使用角色配置文件运行 Amazon CLI 命令。相反，运行命令的权限来自附加到角色 的权限。您可以将权限策略附加到角色，以指定可以针对哪些 Amazon 资源执行哪些操作。有关向角色附加权限（与向用户附加权限的操作相同）的更多信息，请参阅《IAM 用户指南》中的更改 IAM 用户的权限。

既然您已正确地配置角色配置文件、角色权限、角色信任关系和用户权限，那么就可以通过调用 `--profile` 选项在命令行中使用该角色了。例如，下面的内容使用附加到 `ls` 角色（由本主题开头的示例定义）的权限调用 Amazon S3 `marketingadmin` 命令。

```
$ aws s3 ls --profile marketingadmin
```

要对多个调用使用角色，您可以从命令行设置当前会话的 AWS_PROFILE 环境变量。定义该环境变量后，就不必对每个命令都指定 --profile 选项。

Linux 或 macOS

```
$ export AWS_PROFILE=marketingadmin
```

Windows

```
C:\> setx AWS_PROFILE marketingadmin
```

有关配置用户和角色的更多信息，请参阅《IAM 用户指南》中的 IAM 身份（用户、用户组和角色）和 IAM 角色。

使用多重验证

为了提高安全性，当用户尝试使用角色配置文件进行调用时，您可以要求用户提供从多重验证 (MFA) 设备（一种 U2F 设备）或移动应用程序生成的一次性密钥。

首先，您可以选择将与 IAM 角色有关的信任关系修改为需要 MFA。这可以防止任何人在未首先使用 MFA 进行身份验证的情况下使用该角色。有关示例，请参阅下面示例中的 Condition 行。此策略允许名为 anika 的用户代入策略所附加的角色，但前提是用户使用 MFA 进行身份验证。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/anika" },
      "Action": "sts:AssumeRole",
      "Condition": { "Bool": { "aws:multifactorAuthPresent": true } }
    }
  ]
}
```

其次，为角色配置文件添加一行，用来指定用户的 MFA 设备的 ARN。以下示例 config 文件条目显示两个角色配置文件，它们都使用用户 anika 的访问密钥来请求角色 cli-role 的临时凭证。用户 anika 有权代入角色，这是由角色的信任策略授予的。

```
[profile role-without-mfa]
region = us-west-2
role_arn= arn:aws:iam::128716708097:role/cli-role
source_profile=cli-user

[profile role-with-mfa]
region = us-west-2
role_arn= arn:aws:iam::128716708097:role/cli-role
source_profile = cli-user
mfa_serial = arn:aws:iam::128716708097:mfa/cli-user

[profile cli-user]
region = us-west-2
output = json
```

该 mfa_serial 设置可以采取如图所示的 ARN 或硬件 MFA 令牌的序列号。

第一个配置文件 role-without-mfa 不需要 MFA。但是，由于附加到角色的先前示例信任策略需要 MFA，因此使用此配置文件运行命令的任何尝试都将失败。

```
$ aws iam list-users --profile role-without-mfa

An error occurred (AccessDenied) when calling the AssumeRole operation: Access denied
```

第二个配置文件条目 role-with-mfa 标识要使用的 MFA 设备。当此用户尝试使用此配置文件运行 Amazon CLI 命令时，Amazon CLI 会提示用户输入 MFA 设备提供的一次性密码 (OTP)。如果 MFA 身份验证成功，此命令将执行所请求的操作。OTP 未显示在屏幕上。

```
$ aws iam list-users --profile role-with-mfa
Enter MFA code for arn:aws:iam::123456789012:mfa/cli-user:
{
    "Users": [
        {
            ...
```

跨账户角色和外部 ID

通过将角色配置为跨账户角色，您可以让 用户使用属于不同账户的角色。在创建角色期间，将角色类型设置为其他Amazon账户（ 如创建向 IAM 用户委派权限的角色中所述 ）。（ 可选 ）选择 Require MFA (需要 MFA)。Require MFA (需要 MFA) 将按照 使用多重验证 中所述在信任关系中配置相应条件。

如果使用外部 ID 来加强控制可跨账户使用角色的人员，则还必须将 `external_id` 参数添加到角色配置文件。通常情况下，仅应在其他账户由公司或组织外部的人员控制时才使用该功能。

```
[profile crossaccountrole]
role_arn = arn:aws:iam::234567890123:role/SomeRole
source_profile = default
mfa_serial = arn:aws:iam::123456789012:mfa/saanvi
external_id = 123456
```

指定角色会话名称以便于审核

当许多人共享一个角色时，审核会变得比较困难。您希望将调用的每个操作与调用该操作的个人关联。但是，当个人使用角色时，个人代入角色是一项独立于调用操作的行为，您必须手动将这两者关联起来。

通过在用户代入角色时指定唯一的角色会话名称，您可以简化此过程。只需向指定某一角色的 `role_session_name` 文件中的每个命名配置文件添加 `config` 参数，即可实现这一点。`role_session_name` 值将传递给 `AssumeRole` 操作，并成为角色会话 ARN 的一部分。该值也包含在所有已记录操作的 Amazon CloudTrail 日志中。

例如，您可以创建基于角色的配置文件，如下所示。

```
[profile namedsessionrole]
role_arn = arn:aws:iam::234567890123:role/SomeRole
source_profile = default
role_session_name = Session_Maria_Garcia
```

这会导致角色会话具有以下 ARN。

```
arn:aws:iam::234567890123:assumed-role/SomeRole/Session_Maria_Garcia
```

此外，所有 Amazon CloudTrail 日志都在为每个操作捕获的信息中包含角色会话名称。

通过 Web 身份代入角色

您可以配置一个配置文件，以指示 Amazon CLI 应使用 Web 联合身份验证和 Open ID Connect (OIDC) 代入角色。当您在配置文件中指定此选项时，Amazon CLI 会自动为您发出相应的 Amazon STS `AssumeRoleWithWebIdentity` 调用。

> **Note**
>
> 当您指定使用 IAM 角色的配置文件时，Amazon CLI 会发出相应的调用来检索临时凭证。这些凭证存储在 `~/.aws/cli/cache` 中。指定同一个配置文件的后续 Amazon CLI 命令将使用缓存的临时凭证，直到它们过期。这时，Amazon CLI 将自动刷新这些凭证。

要通过 Web 联合身份验证检索和使用临时凭证，您可以在共享配置文件中指定以下配置值。

role_arn

指定要代入的角色的 ARN。

web_identity_token_file

指定一个文件的路径，该文件包含由身份提供者提供的 OAuth 2.0 访问令牌或 OpenID Connect ID 令牌。Amazon CLI 加载此文件，并将其内容作为 `WebIdentityToken` 操作的 `AssumeRoleWithWebIdentity` 参数传递。

role_session_name

指定应用于此代入角色会话的可选名称。

以下是使用 Web 身份配置文件配置代入角色所需的最少量配置的示例。

```
# In ~/.aws/config

[profile web-identity]
role_arn=arn:aws:iam:123456789012:role/RoleNameToAssume
web_identity_token_file=/path/to/a/token
```

您也可以使用环境变量提供此配置。

AWS_ROLE_ARN

要代入的角色的 ARN。

AWS_WEB_IDENTITY_TOKEN_FILE

Web 身份令牌文件的路径。

AWS_ROLE_SESSION_NAME

应用于此代入角色会话的名称。

> **Note**
>
> 这些环境变量当前仅适用于通过 Web 身份提供程序代入角色。它们不适用于常规的代入角色提供程序配置。

清除缓存的凭证

当您使用一个角色时，Amazon CLI 会在本地缓存临时凭证，直到这些凭证过期。当您下次尝试使用它们时，Amazon CLI 会尝试代表您续订这些凭证。

如果您的角色的临时凭证已吊销，它们不会自动续订，并且使用它们的尝试将失败。但是，您可以删除缓存以强制 Amazon CLI 检索新凭证。

Linux 或 macOS

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

在 Amazon CLI 中使用 IAM 用户凭证进行身份验证

> **Warning**
>
> 为了避免安全风险，在开发专用软件或处理真实数据时，请勿使用 IAM 用户进行身份验证，而是使用与身份提供者的联合身份验证，例如 Amazon IAM Identity Center。

此部分介绍如何使用 IAM 用户配置基本设置。其中包括使用 `config` 和 `credentials` 文件的安全凭证。

主题

- 步骤 1：创建您的 IAM 用户
- 步骤 2：获取您的访问密钥
- 步骤 3：配置 Amazon CLI
  - 使用 aws configure

步骤 1：创建您的 IAM 用户

按照《IAM 用户指南》中的创建 IAM 用户（控制台）过程操作来创建 IAM 用户。

- 对于权限选项，选择直接附加策略以了解如何向该用户分配权限。
- 大多数“入门”开发工具包教程都使用 Amazon S3 服务作为示例。要向应用程序提供对 Amazon S3 的完全访问权限，请选择要附加到此用户的 `AmazonS3FullAccess` 策略。

步骤 2：获取您的访问密钥

1. 登录 Amazon Web Services 管理控制台，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在 IAM 控制台的导航窗格中，选择用户，然后选择您之前创建用户的 **`User name`**。
3. 在用户的页面上，选择安全凭证页面。然后，在访问密钥下，选择创建访问密钥。
4. 对于创建访问密钥步骤 1，选择命令行界面 (CLI)。
5. 对于创建访问密钥步骤 2，输入可选标记并选择下一步。
6. 对于创建访问密钥步骤 3，选择下载.csv 文件以保存包含您的 IAM 用户访问密钥和秘密访问密钥的 `.csv` 文件。稍后您将需要此信息。
7. 选择 Done (完成)。

步骤 3：配置 Amazon CLI

在常规使用中，Amazon CLI 需要以下信息：

- 访问密钥 ID

- 秘密访问密钥
- Amazon 区域
- 输出格式

Amazon CLI 将这些信息存储在 default 文件中名为 credentials 的配置文件（一个设置集合）中。预设情况下，当您运行的 Amazon CLI 命令未明确指定要使用的配置文件时，将使用此配置文件中的信息。有关 credentials 文件的更多信息，请参阅Amazon CLI 中的配置和凭证文件设置。

要配置 Amazon CLI，请使用下列过程之一：

主题

- 使用 aws configure

使用 aws configure

对于一般用途，aws configure 命令是设置 Amazon CLI 安装的最快方法。此配置向导将提示您输入入门所需的每条信息。除非使用 --profile 选项另行指定，否则 Amazon CLI 会将此信息存储在 default 配置文件中。

以下示例使用示例值配置 default 配置文件。将它们替换为您自己的值，如以下部分所述。

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

以下示例使用示例值配置名为 userprod 的配置文件。将它们替换为您自己的值，如以下部分所述。

```
$ aws configure --profile userprod
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

在 Amazon CLI 中将 Amazon EC2 实例元数据用作凭证

从 Amazon Elastic Compute Cloud (Amazon EC2) 实例中运行 Amazon CLI 时，可以简化向命令提供凭证的过程。每个 Amazon EC2 实例都包含 Amazon CLI 能够直接查询临时凭证的元数据。向实例附加 IAM 角色后，Amazon CLI 可以自动并且安全地从实例元数据检索凭证。

要禁用此服务，请使用 AWS_EC2_METADATA_DISABLED 环境变量。

主题

- 先决条件
- 配置 Amazon EC2 元数据的配置文件

先决条件

要将 Amazon EC2 凭证与 Amazon CLI 结合使用，您需要完成以下操作：

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI 和Amazon CLI 身份验证和访问凭证。
- 您了解配置文件和命名配置文件。有关更多信息，请参阅 Amazon CLI 中的配置和凭证文件设置。
- 您已创建一个对所需资源有访问权限的 Amazon Identity and Access Management (IAM) 角色，然后在 Amazon EC2 实例启动时向其附加了该角色。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 策略和《IAM 用户指南》中的向在 Amazon EC2 实例上运行的应用程序授予访问 Amazon 资源的权限。

配置 Amazon EC2 元数据的配置文件

要指定需要使用在托管 Amazon EC2 实例配置文件中提供的凭证，请在配置文件的命名配置文件中使用以下语法。有关更多说明，请参阅以下步骤。

```
[profile profilename]
role_arn = arn:aws:iam::123456789012:role/rolename
credential_source = Ec2InstanceMetadata
region = region
```

1. 在配置文件中创建配置文件。

   ```
   [profile profilename]
   ```

2. 添加有权访问所需资源的 IAM arn 角色。

```
role_arn = arn:aws:iam::123456789012:role/rolename
```

3. 指定 Ec2InstanceMetadata 作为凭证源。

```
credential_source = Ec2InstanceMetadata
```

4. 设置您的区域。

```
region = region
```

示例

以下示例代入 *marketingadminrole* 角色并在名为 *marketingadmin* 的 Amazon EC2 实例配置文件中使用 *us-west-2* 区域。

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
credential_source = Ec2InstanceMetadata
region = us-west-2
```

在 Amazon CLI 中使用外部进程获取凭证

> **Warning**
>
> 本主题讨论从外部进程获取凭证。如果生成凭证的命令可由未经批准的进程或用户访问，则可能存在安全风险。我们建议您使用 Amazon CLI 和 Amazon 提供的支持的安全替代方案，以降低泄露凭证的风险。请务必保管好 config 文件及任何支持文件和工具，以防泄露。
> 确保您的自定义凭证工具不会将任何秘密信息写入 StdErr，因为开发工具包和 Amazon CLI 可以捕获和记录此类信息，可能会将其向未经授权的用户公开。

如果您有 Amazon CLI 不直接支持的生成或查找凭证的方法，则可以通过在 config 文件中配置 credential_process 设置来配置 Amazon CLI 使用它。

例如，您可以在 config 文件中包含类似于以下内容的条目。

```
[profile developer]
credential_process = /opt/bin/awscreds-custom --username helen
```

语法

要以与任何操作系统兼容的方式创建此字符串，请遵循以下规则：

- 如果路径或文件名包含空格，请将完整路径和文件名用双引号 (" ") 括起来。该路径和文件名仅包含以下字符：A-Z a-z 0-9 - _ . 空格
- 如果参数名称或参数值包含空格，则用双引号 (" ") 将该元素括起来。仅括起来名称或值，而不是名称值对。
- 请勿在字符串中包含任何环境变量。例如，您不能包含 $HOME 或 %USERPROFILE%。
- 不要将主文件夹指定为 ~。您必须指定完整路径。

Windows 示例

```
credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter
 with spaces"
```

Linux 或 macOS 示例

```
credential_process = "/Users/Dave/path/to/credentials.sh" parameterWithoutSpaces
 "parameter with spaces"
```

凭证计划的预期输出

Amazon CLI 按照配置文件中指定的方式运行该命令，然后从 STDOUT 读取数据。您指定的命令必须在 STDOUT 上生成符合以下语法的 JSON 输出。

```
{
  "Version": 1,
  "AccessKeyId": "an Amazon access key",
  "SecretAccessKey": "your Amazon secret access key",
  "SessionToken": "the Amazon session token for temporary credentials",
  "Expiration": "ISO8601 timestamp when the credentials expire"
}
```

> **Note**
>
> 截至撰写本文之时，`Version` 密钥必须设置为 1。随时间推移和该结构的发展，该值可能会增加。

`Expiration` 密钥是采用 ISO8601 格式的时间戳。如果工具的输出中不存在 `Expiration` 键，则 CLI 假定凭证是不刷新的长期凭证。否则，将其视为临时凭证，并通过在其过期前重新运行 `credential_process` 命令来自动刷新凭证。

> **Note**
>
> Amazon CLI 不 缓存外部进程凭据，这一点不同于代入角色凭证。如果需要缓存，则必须在外部进程中实现。

外部进程可以返回非零返回代码，以指示在检索凭证时发生错误。

使用 Amazon CLI

除the section called “了解如何查看、监控和管理 SageMaker 端点。”一节中介绍的详细内容外，本节还概述了 Amazon Command Line Interface（Amazon CLI）的一般用法、常见功能和可用选项。

本指南深入探讨了编写 Amazon CLI 命令的基本方面，包括这些命令的基本结构、格式设置和筛选功能。通过了解这些核心元素，您将能够构建精确针对所需资源和操作的命令，而无需浏览基于 Web 的复杂控制台。

此外，还重点介绍了适用于 Amazon CLI 的帮助内容和文档。从内置命令行帮助到全面的 Amazon CLI 参考指南，您可以访问相关信息，以深入了解 Amazon CLI 的特性和功能。

有关 Amazon Web Services 服务的具体示例和使用案例，请参阅Amazon CLI 示例或 Amazon CLI 参考指南。这些文档提供了特定于命令的信息，并演示了如何将 Amazon CLI 用于各种 Amazon Web Services 服务。

> **Note**
>
> 默认情况下，Amazon CLI 通过在 TCP 端口 443 上使用 HTTPS，将请求发送到 Amazon Web Services 服务。要成功使用 Amazon CLI，您必须能够在该端口上建立出站连接。

本指南中的主题

- 在 Amazon CLI 中获取帮助和资源
- Amazon CLI 中的命令结构
- 在 Amazon CLI 中指定参数值
- 在 Amazon CLI 中控制命令输出
- Amazon CLI 中的命令行返回代码
- 在 Amazon CLI 中创建和使用别名
- 排查 Amazon CLI 错误

在 Amazon CLI 中获取帮助和资源

本主题介绍如何访问 Amazon Command Line Interface（Amazon CLI）的帮助内容。

主题

- 内置 Amazon CLI 帮助命令
- Amazon CLI 参考指南
- API 文档
- 错误故障排除
- 其他帮助

内置 Amazon CLI 帮助命令

使用 Amazon Command Line Interface（Amazon CLI）时，您可以获得任何命令的帮助。为此，只需在命令名称末尾键入 `help`。

例如，以下命令显示常规 Amazon CLI 选项和可用顶层命令的帮助。

```
$ aws help
```

以下命令显示可用的 Amazon Elastic Compute Cloud (Amazon EC2) 特定命令。

```
$ aws ec2 help
```

以下示例显示 Amazon EC2 `DescribeInstances` 操作的详细帮助。帮助包括对其输入参数、可用筛选条件以及作为输出包含的内容的描述。它还包含说明如何键入命令的常见变体的示例。

```
$ aws ec2 describe-instances help
```

每个命令的帮助内容分为六个部分：

名称

命令的名称。

```
NAME
       describe-instances -
```

说明

命令调用的 API 操作的描述。

```
DESCRIPTION
       Describes one or more of your instances.
```

```
       If you specify one or more instance IDs, Amazon EC2 returns information
       for those instances. If you do not specify  instance  IDs,  Amazon  EC2
       returns  information  for  all  relevant  instances.  If you specify an
       instance ID that is not valid, an error is returned. If you specify  an
       instance  that  you  do  not  own,  it  is not included in the returned
       results.
...
```

摘要

使用命令及其选项的基本语法。如果某个选项显示在方括号中，则表示该选项是可选的、具有默认值或具有可使用的替代选项。

```
SYNOPSIS
            describe-instances
          [--dry-run | --no-dry-run]
          [--instance-ids <value>]
          [--filters <value>]
          [--cli-input-json <value>]
          [--starting-token <value>]
          [--page-size <value>]
          [--max-items <value>]
          [--generate-cli-skeleton]
```

例如，describe-instances 具有描述当前账户和 Amazon 区域中的所有实例的默认行为。您可以选择指定 instance-ids 列表来描述一个或多个实例；dry-run 是不接受值的可选布尔标志。要使用布尔标志，请指定其中一个显示的值，在本例中为 --dry-run 或 --no-dry-run。同样，--generate-cli-skeleton 不使用值。如果某个选项的使用存在条件，则在 OPTIONS 部分中描述这些条件，或在示例中显示这些条件。

选项

对摘要中显示的每个选项的描述。

```
OPTIONS
       --dry-run | --no-dry-run (boolean)
          Checks whether you have the required  permissions  for  the  action,
          without actually making the request, and provides an error response.
          If you have the required permissions, the error response is  DryRun-
          Operation . Otherwise, it is UnauthorizedOperation .

       --instance-ids (list)
```

```
          One or more instance IDs.

          Default: Describes all your instances.
...
```

示例

一些示例，用于显示命令及其选项的使用。如果您需要的命令或用例没有示例可用，请使用本页面上的反馈链接请求一个示例，或在命令的帮助页上的 Amazon CLI 命令参考中请求一个示例。

```
    EXAMPLES
    To describe an Amazon EC2 instance

    Command:

    aws ec2 describe-instances --instance-ids i-5203422c

    To describe all instances with the instance type m1.small

    Command:

    aws ec2 describe-instances --filters "Name=instance-type,Values=m1.small"

    To describe all instances with an Owner tag

    Command:

    aws ec2 describe-instances --filters "Name=tag-key,Values=Owner"
...
```

输出

来自的响应中包含的每个字段和数据类型的描述Amazon

对于 `describe-instances`，输出是预留对象的列表，每个列表都包含若干字段和对象，这些字段和对象包含与其关联的实例的相关信息。此信息来自 Amazon EC2 使用的预留数据类型的 API 文档。

```
OUTPUT
       Reservations -> (list)
          One or more reservations.

          (structure)
```

```
               Describes a reservation.

               ReservationId -> (string)
                  The ID of the reservation.

               OwnerId -> (string)
                  The ID of the Amazon account that owns the reservation.

               RequesterId -> (string)
                  The ID of the requester that launched the instances  on  your
                  behalf (for example, Amazon Management Console or Auto Scaling).

               Groups -> (list)
                  One or more security groups.

                  (structure)
                      Describes a security group.

                      GroupName -> (string)
                         The name of the security group.

                      GroupId -> (string)
                         The ID of the security group.

               Instances -> (list)
                  One or more instances.

                  (structure)
                      Describes an instance.

                      InstanceId -> (string)
                         The ID of the instance.

                      ImageId -> (string)
                         The ID of the AMI used to launch the instance.

                      State -> (structure)
                         The current state of the instance.

                         Code -> (integer)
                             The  low  byte represents the state. The high byte
                             is an opaque internal value and should be ignored.
...
```

当 Amazon CLI 将输出呈现为 JSON 时，输出将成为预留对象的数组，类似于以下示例。

```
{
    "Reservations": [
        {
            "OwnerId": "012345678901",
            "ReservationId": "r-4c58f8a0",
            "Groups": [],
            "RequesterId": "012345678901",
            "Instances": [
                {
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "PublicDnsName": "ec2-52-74-16-12.us-
west-2.compute.amazonaws.com",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
...
```

每个预留对象都包含一些描述预留的字段和一组实例对象，每个实例对象又带有用来描述它的字段（如 PublicDnsName）和对象（如 State）。

> **Windows 用户**
>
> 您可以通过管道 (|) 将 help 命令的输出发送到 more 命令以便每次查看一页帮助文件。按空格键或 PgDn 键可查看文档的更多内容，按 **q** 可退出。
>
> ```
> C:\> aws ec2 describe-instances help | more
> ```

Amazon CLI 参考指南

帮助文件包含无法通过命令行查看或导航至的链接。您可以使用线上 Amazon CLI 版本 1 参考指南查看这些链接，并与其交互。参考指南还包含所有 Amazon CLI 命令的帮助内容。将显示这些说明以方便在手机、平板电脑或桌面屏幕进行浏览和查看。

API 文档

Amazon CLI 中的所有命令对应于对 Amazon 服务的公用 API 发出的请求。具有公用 API 的每项服务都有一个 API 引用，可从Amazon文档网站上该服务的主页找到。API 参考的内容因 API 的构造方式以及所用协议而有所不同。通常，API 参考包含有关该 API 支持的操作、发送到该服务和从该服务发送的数据以及该服务可能报告的任何错误状况的详细信息。

API 文档的各部分

- Actions（操作）– 有关每个操作及其参数（包括对长度或内容以及默认值的约束）的详细信息。它列出了此操作可能发生的错误。每个操作对应于 Amazon CLI 中的一个子命令。
- Data Types（数据类型）– 有关命令可能需要作为参数或在响应请求时要返回的结构的详细信息。
- Common Parameters （常用参数）– 有关由服务的所有操作共享的参数的详细信息。
- Common Errors（常见错误）– 有关可能由服务的任意操作返回的错误的详细信息。

每个部分的名称和可用性可能根据具体服务而不同。

> **特定于服务的 CLI**
>
> 与以前创建单个 Amazon CLI 以处理所有服务不同，有些服务还有单独的 CLI。这些特定于服务的 CLI 具有单独的文档，该服务的文档页面包含指向该文档的链接。特定于服务的 CLI 的文档不适用于 Amazon CLI。

错误故障排除

如需帮助诊断和修复 Amazon CLI 错误，请参阅 the section called “排查错误”。

其他帮助

有关 Amazon CLI 问题的其他帮助，请访问 GitHub 上的 Amazon CLI 社区。

Amazon CLI 中的命令结构

本主题介绍如何构建 Amazon Command Line Interface（Amazon CLI）命令，以及如何使用 wait 命令。

主题

- 命令结构
- Wait 命令

命令结构

Amazon CLI 在命令行上使用多部分结构，各部分必须按如下顺序指定：

1. 对 aws 计划的基本调用。
2. 顶级命令，这通常对应于 Amazon CLI 支持的Amazon服务。
3. 用于指定要执行的操作的子命令。
4. 常规 Amazon CLI 选项或操作所需的参数。您可以按任意顺序指定这些项，只要它们跟在前三个部分之后。如果多次指定某个排他参数，则仅应用最后一个值。

```
$ aws <command> <subcommand> [options and parameters]
```

参数可采用各种类型的输入值，如数字、字符串、列表、映射和 JSON 结构。支持的内容取决于您指定的命令和子命令。

示例

Amazon S3

以下示例列出您的所有 Amazon S3 存储桶。

```
$ aws s3 ls
2018-12-11 17:08:50 amzn-s3-demo-bucket1
2018-12-14 14:55:44 amzn-s3-demo-bucket2
```

有关 Amazon S3 命令的更多信息，请参阅 Amazon CLI 命令参考 中的 aws s3。

Amazon CloudFormation

以下 create-change-set 命令示例将 cloudformation 堆栈名称更改为 *my-change-set*。

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set-name my-
change-set
```

有关 Amazon CloudFormation 命令的更多信息，请参阅 Amazon CLI 命令参考 中的 aws cloudformation。

Wait 命令

一些 Amazon 服务提供 wait 命令。使用 aws wait 的任何命令通常都会等到命令完成后再进入下一步。这对于多部分命令或脚本编写来说特别有用，因为当命令失败时，您可以使用 wait 命令阻止进入后续步骤。

Amazon CLI 在命令行上对 wait 命令使用多部分结构，各部分必须按如下顺序指定：

1. 对 aws 计划的基本调用。
2. 顶级命令，这通常对应于 Amazon CLI 支持的Amazon服务。
3. wait 命令。
4. 用于指定要执行的操作的子命令。
5. 常规 CLI 选项或操作所需的参数。您可以按任意顺序指定这些项，只要它们跟在前三个部分之后。如果多次指定某个排他参数，则仅应用最后一个值。

```
$ aws <command> wait <subcommand> [options and parameters]
```

参数可采用各种类型的输入值，如数字、字符串、列表、映射和 JSON 结构。支持的内容取决于您指定的命令和子命令。

> **Note**
>
> 并不是所有 AWS 服务都支持 wait 命令。请参阅 Amazon CLI 参考指南，了解您的服务是否支持 wait 命令。

示例

Amazon CloudFormation

在下面的 wait change-set-create-complete 命令示例中，命令将暂停运行，并且仅在可确认 *my-stack* 堆栈中的 *my-change-set* 更改集已准备好运行后才恢复运行。

```
$ aws cloudformation wait change-set-create-complete --stack-name my-stack --change-set-name my-change-set
```

有关 Amazon CloudFormation `wait` 命令的更多信息，请参阅《Amazon CLI 命令参考》中的 `wait`。

Amazon CodeDeploy

在下面的 `wait deployment-successful` 命令示例中，命令将暂停运行，直到 *d-A1B2C3111* 部署成功完成。

```
$ aws deploy wait deployment-successful --deployment-id d-A1B2C3111
```

有关 Amazon CodeDeploy `wait` 命令的更多信息，请参阅《Amazon CLI 命令参考》中的 `wait`。

在 Amazon CLI 中指定参数值

Amazon Command Line Interface（Amazon CLI）中使用的很多参数都是简单的字符串或数值，例如下面的 `aws ec2 create-key-pair` 命令示例中的密钥对名称 `my-key-pair`。

```
$ aws ec2 create-key-pair --key-name my-key-pair
```

终端之间的命令格式可能会有所不同。例如，大多数终端区分大小写，但 Powershell 不区分大小写。这意味着，以下两个命令示例对于区分大小写的终端会产生不同的结果，因为它们将 `MyFile*.txt` 和 `myfile*.txt` 视为不同的参数。

但是，PowerShell 会将 `MyFile*.txt` 和 `myfile*.txt` 视为相同的参数来处理这些请求。以下命令示例使用 `aws s3 cp` 命令演示了这些参数：

```
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --include "MyFile*.txt"
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --include "myfile*.txt"
```

有关 PowerShell 的不区分大小写的更多信息，请参阅 PowerShell 文档中的 about_Case-Sensitivity

有时，您需要在包含特殊字符或空格字符的字符串周围使用引号或文字。有关此格式的规则也可能因终端而异。有关在复杂参数周围使用引号的更多信息，请参阅 在 Amazon CLI 中将引号和文本与字符串结合使用。

这些主题涵盖最常见的终端格式设置规则。如果您在终端识别参数值时遇到问题，请务必查看本节中的主题，并查看终端文档以了解其特定语法规则。

参数主题

- Amazon CLI 中的通用参数类型
- 在 Amazon CLI 中将引号和文本与字符串结合使用
- 在 Amazon CLI 中从文件加载参数
- 在 Amazon CLI 中生成 Amazon CLI 骨架和输入文件
- 在 Amazon CLI 中使用速记语法

Amazon CLI 中的通用参数类型

本节介绍一些通用参数类型以及典型的所需格式。

如果您不知道如何设置特定命令的参数格式，请在命令名称后输入 **help** 来查看帮助。每个子命令的帮助均包括一个选项的名称和描述。该选项的参数类型在括号中列出。有关查看帮助的更多信息，请参阅 the section called “获取帮助”。

参数类型包括：

- String
- Timestamp
- 列出
- 布尔值
- 整数
- 二进制/Blob（二进制大型对象）和流式传输 Blob
- 映射
- 文档

String

字符串参数可以包含 ASCII 字符集中的字母数字字符、符号和空格。包含空格的字符串必须用引号引起来。建议您不要使用标准空格字符以外的符号或空格，并遵循终端的引用规则，以防止出现意外结果。

一些字符串参数可接受来自文件的二进制数据。有关示例，请参阅二进制文件。

Timestamp

时间戳根据 ISO 8601 标准设置格式。这些通常称为“DateTime”或“Date”参数。

```
$ aws ec2 describe-spot-price-history --start-time 2014-10-13T19:00:00Z
```

可接受的格式包括：

- *YYYY-MM-DD*T*hh*:*mm*:*ss.sssTZD (UTC)*，例如，2014-10-01T20:30:00.000Z
- *YYYY-MM-DD*T*hh*:*mm*:*ss.sssTZD######*，例如，2014-10-01T12:30:00.000-08:00
- *YYYY-MM-DD*，例如，2014-10-01
- 以秒为单位的 Unix 时间，如 1412195400。这有时称为 Unix 纪元时间，表示自 1970 年 1 月 1 日午夜 (UTC) 以来经历的秒数。

您可以使用 `cli_timestamp_format` 文件设置来设置时间戳格式。

列出

以空格分隔的一个或多个字符串。如果任何字符串项目包含空格，则必须用引号括起该项目。遵循您终端的引号规则以防止出现意外结果。

```
$ aws ec2 describe-spot-price-history --instance-types m1.xlarge m1.medium
```

布尔值

打开或关闭某一选项的二进制标志。例如，`ec2 describe-spot-price-history` 有一个布尔 `--dry-run` 参数，如果指定该参数，则针对服务验证查询而不实际运行查询。

```
$ aws ec2 describe-spot-price-history --dry-run
```

输出指示命令格式是否正确。此命令还包含一个 `--no-dry-run` 参数版本，可以用来显式指示命令应正常运行。不过不是必须包含此参数，因为这是默认行为。

整数

无符号整数。

```
$ aws ec2 describe-spot-price-history --max-items 5
```

二进制/Blob（二进制大型对象）和流式传输 Blob

在 Amazon CLI 中，您可以将二进制值作为字符串直接在命令行上传递。共有两种类型的 blob：

- Blob
- 流式 blob

Blob

要将值传递给类型为 blob 的参数，必须使用 fileb:// 前缀指定包含二进制数据的本地文件的路径。使用 fileb:// 前缀引用的文件始终被作为原始未编码二进制文件进行处理。指定的路径被解释为相对于当前工作目录。例如，适用于 aws kms encrypt 的 --plaintext 参数是一个 blob。

```
$ aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob | base64 \
    --decode > ExampleEncryptedFile
```

流式 Blob

aws cloudsearchdomain upload-documents 等流式 Blob 不使用前缀。相反，流式传输 blob 参数使用直接文件路径进行格式化。以下示例对于 aws cloudsearchdomain upload-documents 命令使用直接文件路径 document-batch.json：

```
$ aws cloudsearchdomain upload-documents \
    --endpoint-url https://doc-my-domain.us-west-1.cloudsearch.amazonaws.com \
    --content-type application/json \
    --documents document-batch.json
```

映射

使用 JSON 或 CLI 的速记语法指定的一系列密钥值对。以下 JSON 示例使用 map 参数 --key 从名为 my-table 的 Amazon DynamoDB 表中读取项目。此参数在嵌套的 JSON 结构中指定名为 id 且数值为 1 的主键。

要在命令行中使用更高级的 JSON，请考虑使用 jq 等命令行 JSON 处理器来创建 JSON 字符串。有关 jq 的更多信息，请参阅 GitHub 上的 jq 存储库。

```
$ aws dynamodb get-item --table-name my-table --key '{"id": {"N":"1"}}'

{
```

```
    "Item": {
        "name": {
            "S": "John"
        },
        "id": {
            "N": "1"
        }
    }
}
```

文档

> **Note**
>
> 速记语法与文档类型不兼容。

文档类型用于发送数据，无需在字符串中嵌入 JSON。文档类型使服务能够提供任意架构，以便您使用更灵活的数据类型。

这使得无需对值转义，即可发送 JSON 数据。例如，不使用以下转义的 JSON 输入：

```
{"document": "{\"key\":true}"}
```

您可以使用以下文档类型：

```
{"document": {"key": true}}
```

文档类型的有效值

由于文档类型本身多种多样，因此存在多个有效值类型。有效值包括：

字符串

```
--option '"value"'
```

数字

```
--option 123
--option 123.456
```

布尔值

```
--option true
```

Null

```
--option null
```

数组

```
--option '["value1", "value2", "value3"]'
--option '["value", 1, true, null, ["key1", 2.34], {"key2": "value2"}]'
```

对象

```
--option '{"key": "value"}'
--option '{"key1": "value1", "key2": 123, "key3": true, "key4": null, "key5":
 ["value3", "value4"], "key6": {"value5": "value6"}'
```

在 Amazon CLI 中将引号和文本与字符串结合使用

在 Amazon CLI 中使用单引号和双引号主要有两种方式。

- 在包含空格的字符串周围使用引号
- 在字符串内使用引号

在包含空格的字符串周围使用引号

参数名称及其值由命令行中的空格分隔。如果字符串值包含嵌入式空格，则必须用引号将整个字符串括起来，以防止 Amazon CLI 将空格误解为值与下一个参数名称之间的分隔符。您使用的引号类型取决于您运行 Amazon CLI 的操作系统。

Linux and macOS

使用单引号 ' '

```
$ aws ec2 create-key-pair --key-name 'my key pair'
```

有关使用引号的更多信息，请参阅首选 Shell 的用户文档。

PowerShell

单引号（推荐）

单引号 ' ' 称为 verbatim 字符串。字符串将完全按照您键入的方式传递给命令，这意味着 PowerShell 变量将不会传递。

```
PS C:\> aws ec2 create-key-pair --key-name 'my key pair'
```

双引号

双引号 " " 称为 expandable 字符串。变量可以在可展开的字符串中传递。

```
PS C:\> aws ec2 create-key-pair --key-name "my key pair"
```

有关使用引号的更多信息，请参阅 Microsoft PowerShell 文档 中的关于引号规则。

Windows command prompt

使用双引号 " "。

```
C:\> aws ec2 create-key-pair --key-name "my key pair"
```

（可选）您可以用等号 = 而不是空格将参数名称和值分隔开。这通常仅在参数的值以连字符开头时有必要。

```
$ aws ec2 delete-key-pair --key-name=-mykey
```

在字符串内使用引号

字符串可能包含引号，并且您的 Shell 可能需要对引号进行转义才能让其正常发挥作用。常见的参数值类型之一是 JSON 字符串。这很复杂，因为它在 JSON 结构中的每个元素名称和值周围都包含空格和双引号 " "。在命令行中输入 JSON 格式参数的方式因操作系统而异。

要在命令行中使用更高级的 JSON，请考虑使用 jq 等命令行 JSON 处理器来创建 JSON 字符串。有关 jq 的更多信息，请参阅 GitHub 上的 jq 存储库。

Linux and macOS

为了让 Linux 和 macOS 按字面含义解释字符串，请使用单引号 ' ' 将 JSON 数据结构括住，如以下示例所示。您不需要对嵌入在 JSON 字符串中的双引号进行转义，因为会按字面含义对它们进行

处理。由于 JSON 用单引号括住，因此字符串中的任何单引号都需要进行转义，这通常通过在单引号 \' 前面使用反斜杠来实现。

```
$ aws ec2 run-instances \
    --image-id ami-12345678 \
    --block-device-mappings '[{"DeviceName":"/dev/sdb","Ebs":
{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

有关使用引号的更多信息，请参阅首选 Shell 的用户文档。

PowerShell

使用单引号 ' ' 或双引号 " "。

单引号（推荐）

单引号 ' ' 称为 verbatim 字符串。字符串将完全按照您键入的方式传递给命令，这意味着 PowerShell 变量将不会传递。

由于 JSON 数据结构包含双引号，因此我们建议使用单 引号 ' ' 将其括起来。如果使用单 引号，则不需要对嵌入在 JSON 字符串中的双 引号进行转义。但是，您需要在 JSON 结构中使用反撇号 ` 对每个单 引号进行转义。

```
PS C:\> aws ec2 run-instances `
    --image-id ami-12345678 `
    --block-device-mappings '[{"DeviceName":"/dev/sdb","Ebs":
{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

双引号

双引号 " " 称为 expandable 字符串。变量可以在可展开的字符串中传递。

如果您使用双 引号，则不需要对嵌入在 JSON 字符串中的单 引号进行转义。但是，您需要在 JSON 结构中使用反撇号 ` 对每个双 引号进行转义，如以下示例所示。

```
PS C:\> aws ec2 run-instances `
    --image-id ami-12345678 `
    --block-device-mappings "[{`"DeviceName`":`"/dev/sdb`",`"Ebs`":
{`"VolumeSize`":20,`"DeleteOnTermination`":false,`"VolumeType`":`"standard`"}}]"
```

有关使用引号的更多信息，请参阅 Microsoft PowerShell 文档 中的关于引号规则。

Warning

PowerShell 在向 Amazon CLI 发送命令之前，它会确定是使用典型 PowerShell 还是 `CommandLineToArgvW` 引用规则来解释您的命令。PowerShell 使用 `CommandLineToArgvW` 进行处理时，必须用反斜杠 \ 对字符进行转义。
有关 PowerShell 中 `CommandLineToArgvW` 的更多信息，请参阅 Microsoft DevBlogs 上的 What's up with the strange treatment of quotation marks and backslashes by CommandLineToArgvW（CommandLineToArgvW 对引号和反斜杠的奇怪处理是怎么回事）、Microsoft Docs Blog 中的 Everyone quotes command line arguments the wrong way（每个人引用命令行参数的方法都错了）以及 Microsoft Docs 上的 CommandLineToArgvW function（CommandLineToArgvW 函数）。

单引号

单引号 ' ' 称为 `verbatim` 字符串。字符串将完全按照您键入的方式传递给命令，这意味着 PowerShell 变量将不会传递。使用反斜杠 \ 对字符进行转义。

```
PS C:\> aws ec2 run-instances `
    --image-id ami-12345678 `
    --block-device-mappings '[{\"DeviceName\":\"/dev/sdb\",\"Ebs\":
{\"VolumeSize\":20,\"DeleteOnTermination\":false,\"VolumeType\":\"standard\"}}]'
```

双引号

双引号 " " 称为 `expandable` 字符串。变量可以在 `expandable` 字符串中传递。对于双引号字符串，必须对每个引号使用 `` `\ `` 进行两次转义，而不是仅使用反引号。反引号对反斜杠进行转义，然后将反斜杠用作 `CommandLineToArgvW` 流程的转义字符。

```
PS C:\> aws ec2 run-instances `
    --image-id ami-12345678 `
    --block-device-mappings "[{`\"DeviceName`\":`\"/dev/sdb`\",`\"Ebs`\":
{`\"VolumeSize`\":20,`\"DeleteOnTermination`\":false,`\"VolumeType`\":`
\"standard`\"}}]"
```

Blob（推荐）

要绕过 JSON 数据输入的 PowerShell 引用规则，请使用 Blob 将 JSON 数据直接传递到 Amazon CLI。有关 Blob 的更多信息，请参阅Blob。

Windows command prompt

Windows 命令提示符要求使用双引号 " " 括住 JSON 数据结构。此外，为了防止命令处理器误解 JSON 中嵌入的双引号，还必须对 JSON 数据结构本身中的每个双引号 \ 进行转义（在前面加一个反斜杠 " 字符），如以下示例所示。

```
C:\> aws ec2 run-instances ^
    --image-id ami-12345678 ^
    --block-device-mappings "[{\"DeviceName\":\"/dev/sdb\",\"Ebs\":
{\"VolumeSize\":20,\"DeleteOnTermination\":false,\"VolumeType\":\"standard\"}}]"
```

只有最外层双引号不进行转义。

在 Amazon CLI 中从文件加载参数

有些参数需要文件名作为变量，Amazon CLI 将从这些变量中加载数据。其他参数允许您将参数值指定为在命令行上键入的文本或从文件中读取的文本。无论文件是必需的还是可选的，您都必须对文件进行正确编码，以便 Amazon CLI 能够理解它。该文件的编码必须与读取系统的默认区域设置相匹配。您可以通过使用 Python `locale.getpreferredencoding()` 方法来确定这一点。

此方法用于为单个参数加载文件。有关使用单个文件加载多个参数的信息，请参阅the section called “生成 CLI 骨架模板”。

> **Note**
>
> 默认情况下，Windows PowerShell 将文本输出为 UTF-16，这与 JSON 文件和许多 Linux 系统使用的 UTF-8 编码冲突。建议您将 `-Encoding ascii` 与 PowerShell `Out-File` 命令一起使用，以确保 Amazon CLI 可以读取生成的文件。

主题

- 如何从文件加载参数
- 二进制文件
- 远程文件
- 将文件作为速记语法值进行加载

如何从文件加载参数

有时，从文件加载参数值（而不是尝试将其全部键入为命令行参数值）很方便，例如当参数为复杂的 JSON 字符串时。要指定包含该值的文件，请按以下格式指定文件 URL。

```
file://complete/path/to/file
```

- 前两个斜杠“/”字符是规范的一部分。如果所需的路径以“/”开头，结果为三个斜杠字符：`file:///folder/file`。
- URL 提供包含实际参数内容的文件的路径。
- 使用带空格或特殊字符的文件时，请遵循终端的引用和转义规则。

> **Note**
>
> 对于本就要求指定 URL 的参数（例如标识 Amazon CloudFormation 模板 URL 的参数），将自动禁用该行为。您也可以通过禁用 Amazon CLI 配置文件中的 `cli_follow_urlparam` 设置来禁用此行为。

以下示例中的文件路径被解读为相对于当前工作目录。

Linux or macOS

```
// Read from a file in the current directory
$ aws ec2 describe-instances --filters file://filter.json

// Read from a file in /tmp
$ aws ec2 describe-instances --filters file:///tmp/filter.json

// Read from a file with a filename with whitespaces
$ aws ec2 describe-instances --filters 'file://filter content.json'
```

Windows command prompt

```
// Read from a file in C:\temp
C:\> aws ec2 describe-instances --filters file://C:\temp\filter.json

// Read from a file with a filename with whitespaces
```

```
C:\> aws ec2 describe-instances --filters "file://C:\temp\filter content.json"
```

file:// 前缀选项支持包含“~/”、“./”和“../”的 Unix 式扩展。在 Windows 上，“~/”表达式将展开到您的用户目录（存储在 %USERPROFILE% 环境变量中）。例如，在 Windows 10 上，您通常在 %USERPROFILE% 下有一个用户目录。

仍必须对作为另一个 JSON 文档的值嵌入的 JSON 文档进行转义。

```
$ aws sqs create-queue --queue-name my-queue --attributes file://attributes.json
```

attributes.json

```
{
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
west-2:0123456789012:deadletter\", \"maxReceiveCount\":\"5\"}"
}
```

二进制文件

对于将二进制数据用作参数的命令，请使用 fileb:// 前缀指定该数据为二进制内容。接受二进制数据的命令包括：

- **aws ec2 run-instances:** --user-data 参数。
- **aws s3api put-object:** --sse-customer-key 参数。
- **aws kms decrypt:** --ciphertext-blob 参数。

以下示例使用 Linux 命令行工具生成一个二进制 256 位 AES 密钥，然后将该密钥提供给 Amazon S3 以对上载的文件服务器端进行加密。

```
$ dd if=/dev/urandom bs=1 count=32 > sse.key
32+0 records in
32+0 records out
32 bytes (32 B) copied, 0.000164441 s, 195 kB/s
$ aws s3api put-object \
    --bucket amzn-s3-demo-bucket \
    --key test.txt \
    --body test.txt \
    --sse-customer-key fileb://sse.key \
    --sse-customer-algorithm AES256
```

```
{
    "SSECustomerKeyMD5": "iVg8oWa8sy714+FjtesrJg==",
    "SSECustomerAlgorithm": "AES256",
    "ETag": "\"a6118e84b76cf98bf04bbe14b6045c6c\""
}
```

远程文件

Amazon CLI 还支持使用 `http://` 或 `https://` URL 从 Internet 上托管的文件中加载参数。下面的示例引用存储在 Amazon S3 存储桶中的一个文件。这将允许您从任何计算机访问参数文件，但它的确要求容器可公开访问。

```
$ aws ec2 run-instances \
    --image-id ami-12345678 \
    --block-device-mappings http://amzn-s3-demo-bucket.s3.amazonaws.com/filename.json
```

前面的示例假定 `filename.json` 文件包含以下 JSON 数据。

```
[
  {
    "DeviceName": "/dev/sdb",
    "Ebs": {
      "VolumeSize": 20,
      "DeleteOnTermination": false,
      "VolumeType": "standard"
    }
  }
]
```

有关引用包含 JSON 格式参数的文件的更多示例，请参阅将 IAM 托管式策略附加到用户。

将文件作为速记语法值进行加载

如果在值较大或较复杂的情况下使用速记语法，通常更易以值的形式加载文件。要将文件作为速记语法值进行加载，格式将略有变化。使用 `@=` 运算符代替 `=` 运算符，而不使用 `key=value`。`@=` 告知 Amazon CLI，值应作为文件路径而不是字符串进行读取。以下示例显示一个键值对，该键值对为其值加载文件。

Linux or macOS

```
--option key@=file://template.txt
```

Windows

```
--option "key1@=file://template.txt"
```

以下示例演示如何为 `aws rolesanywhere create-trust-anchor` 命令加载证书文件。

```
$ aws rolesanywhere create-trust-anchor --name TrustAnchor \
    --source sourceData={x509CertificateData@=file://root-
ca.crt},sourceType="CERTIFICATE_BUNDLE"  \
    --enabled
```

有关速记语法的更多信息，请参阅the section called “速记语法”。

在 Amazon CLI 中生成 Amazon CLI 骨架和输入文件

大多数 Amazon CLI 命令接受从文件导入参数输入。可以使用 `generate-cli-skeleton` 选项生成这些模板，然后使用 `--cli-input-json` 参数将其导入。

主题

- 关于 Amazon CLI 框架和输入文件
- 生成和导入命令骨架
- 合并输入文件和命令行参数

关于 Amazon CLI 框架和输入文件

大多数 Amazon Command Line Interface（Amazon CLI）命令支持使用 `--cli-input-json` 参数接受文件中的参数输入的功能。

这些相同的命令使用 `--generate-cli-skeleton` 参数通过所有可编辑和填写的参数以 JSON 格式生成文件。然后，您可以运行带 `--cli-input-json` 参数的命令并指向填充的文件。

> **Important**
>
> 自定义 Amazon CLI 命令（例如 `aws s3` 命令）不支持本主题中介绍的 `--generate-cli-skeleton` 或 `--cli-input-json` 参数。要检查特定命令是否支持这些参数，请为要使用的命令运行 `help` 命令，或参阅 Amazon CLI 版本 1 参考指南。

`--generate-cli-skeleton` 生成和显示可自定义并用作命令输入的参数模板。生成的模板包含命令支持的所有参数。

`--generate-cli-skeleton` 参数接受以下值之一：

- `input` – 生成的模板包括格式化为 JSON 的所有输入参数。这是默认值。
- `output` – 生成的模板包括格式化为 JSON 的所有输出参数。

由于 Amazon CLI 本质上是围绕服务 API 的“包装程序”，骨架文件预计您会通过他们底层的 API 参数名称引用所有参数。该参数名称可能与 Amazon CLI 参数名称不同。例如，一个名为 Amazon CLI 的 `user-name` 参数可能映射到名为 Amazon 的 `UserName` 服务 API 参数（注意，更改后的大写字母和缺失的破折号）。我们建议您使用 `--generate-cli-skeleton` 选项，用“正确”的参数名称生成模板，以避免错误。您可以参考服务的《API 参考指南》，查看预期的参数名称。您可以从模板中删除不需要的和不想为其提供值的任何参数。

例如，如果您运行以下命令，它会为 Amazon Elastic Compute Cloud（Amazon EC2）命令 run-instances 生成参数模板。

JSON

以下示例演示如何使用 `input` 参数的默认值（`--generate-cli-skeleton`）生成格式化为 JSON 的模板。

```
$ aws ec2 run-instances --generate-cli-skeleton
```

```
{
    "DryRun": true,
    "ImageId": "",
    "MinCount": 0,
    "MaxCount": 0,
    "KeyName": "",
    "SecurityGroups": [
        ""
    ],
    "SecurityGroupIds": [
        ""
    ],
    "UserData": "",
    "InstanceType": "",
    "Placement": {
        "AvailabilityZone": "",
```

```
        "GroupName": "",
        "Tenancy": ""
    },
    "KernelId": "",
    "RamdiskId": "",
    "BlockDeviceMappings": [
        {
            "VirtualName": "",
            "DeviceName": "",
            "Ebs": {
                "SnapshotId": "",
                "VolumeSize": 0,
                "DeleteOnTermination": true,
                "VolumeType": "",
                "Iops": 0,
                "Encrypted": true
            },
            "NoDevice": ""
        }
    ],
    "Monitoring": {
        "Enabled": true
    },
    "SubnetId": "",
    "DisableApiTermination": true,
    "InstanceInitiatedShutdownBehavior": "",
    "PrivateIpAddress": "",
    "ClientToken": "",
    "AdditionalInfo": "",
    "NetworkInterfaces": [
        {
            "NetworkInterfaceId": "",
            "DeviceIndex": 0,
            "SubnetId": "",
            "Description": "",
            "PrivateIpAddress": "",
            "Groups": [
                ""
            ],
            "DeleteOnTermination": true,
            "PrivateIpAddresses": [
                {
                    "PrivateIpAddress": "",
                    "Primary": true
```

```
                }
            ],
            "SecondaryPrivateIpAddressCount": 0,
            "AssociatePublicIpAddress": true
        }
    ],
    "IamInstanceProfile": {
        "Arn": "",
        "Name": ""
    },
    "EbsOptimized": true
}
```

生成和导入命令骨架

生成并使用参数框架文件

1. 使用 `--generate-cli-skeleton` 参数运行命令以生成 JSON，并将输出定向到某个文件以保存它。

   JSON

   ```
   $ aws ec2 run-instances --generate-cli-skeleton input > ec2runinst.json
   ```

2. 在文本编辑器中打开参数骨架文件，并删除任何不需要的参数。例如，您可以将模板缩减到以下内容。请确认在删除不需要的元素后，文件仍是有效的 JSON。

   JSON

   ```
   {
       "DryRun": true,
       "ImageId": "",
       "KeyName": "",
       "SecurityGroups": [
           ""
       ],
       "InstanceType": "",
       "Monitoring": {
           "Enabled": true
       }
   }
   ```

在此示例中，我们将 DryRun 参数设置为 `true` 以使用 Amazon EC2 空运行功能。通过此功能，您可以安全地测试命令，而无需实际创建或修改任何资源。

3. 使用适合您的场景的值填入剩余的值。在本例中，我们提供要使用的 Amazon 机器映像（AMI）的实例类型、密钥名称、安全组和标识符。此示例假定默认 Amazon Web Services 区域。AMI `ami-dfc39aef` 是 us-west-2 区域中托管的 64 位 Amazon Linux 映像。如果您使用不同的区域，您必须查找正确 AMI ID 来使用。

   JSON

   ```
   {
       "DryRun": true,
       "ImageId": "ami-dfc39aef",
       "KeyName": "mykey",
       "SecurityGroups": [
           "my-sg"
       ],
       "InstanceType": "t2.micro",
       "Monitoring": {
           "Enabled": true
       }
   }
   ```

4. 通过使用 `file://` 前缀将完成的模板文件传递到 `--cli-input-json` 参数，使用填写的参数运行命令。Amazon CLI 将路径解释为相对于当前工作目录。在以下示例中，Amazon CLI 在当前工作目录中查找文件。

   JSON

   ```
   $ aws ec2 run-instances --cli-input-json file://ec2runinst.json
   ```

   ```
   A client error (DryRunOperation) occurred when calling the RunInstances
    operation: Request would have succeeded, but DryRun flag is set.
   ```

   空运行错误表明，JSON 格式正确且参数值有效。如果输出中报告了其他问题，请解决这些问题并重复上一步，直到显示“Request would have succeeded”消息。

5. 现在，您可以将 DryRun 参数设置为 `false` 以禁用空运行。

JSON

```
{
    "DryRun": false,
    "ImageId": "ami-dfc39aef",
    "KeyName": "mykey",
    "SecurityGroups": [
        "my-sg"
    ],
    "InstanceType": "t2.micro",
    "Monitoring": {
        "Enabled": true
    }
}
```

6. 运行此命令，run-instances 实际上会启动 Amazon EC2 实例并显示成功启动所生成的详细信息。输出格式由 --output 参数控制，与输入参数模板的格式分开。

JSON

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json --output json
```

```
{
    "OwnerId": "123456789012",
    "ReservationId": "r-d94a2b1",
    "Groups": [],
    "Instances": [
...
```

合并输入文件和命令行参数

输入文件可用于所有参数，也可与 Amazon CLI 中指定的参数组合使用。您可以将此功能用于在输入文件中频繁重用的设置，并将个人设置保留在命令本身中。

以下 aws ec2 run-instances 示例结合使用了输入文件和参数。我们提供要使用的亚马逊机器映像（AMI）的实例类型、密钥名称、安全组和标识符，并假定默认 Amazon Web Services 区域。AMI ami-dfc39aef 是 us-west-2 区域中托管的 64 位 Amazon Linux 映像。如果您使用不同的区域，您必须查找正确 AMI ID 来使用。

JSON

JSON 文件的内容：

```
{
    "ImageId": "ami-dfc39aef",
    "KeyName": "mykey",
    "SecurityGroups": [
        "my-sg"
    ],
    "InstanceType": "t2.micro",
    "Monitoring": {
        "Enabled": true
    }
}
```

以下示例结合使用输入文件与 `--dry-run` 参数来试运行命令，以确认您是否拥有所需的权限并在文件中填入了有效值。

JSON

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json --dry-run
```

```
A client error (DryRunOperation) occurred when calling the RunInstances operation:
 Request would have succeeded, but DryRun flag is set.
```

之后，以下示例将同一输入文件与 `--no-dry-run` 参数结合使用来执行完整命令。

JSON

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json --no-dry-run --
output json
```

```
{
    "OwnerId": "123456789012",
    "ReservationId": "r-d94a2b1",
    "Groups": [],
    "Instances": [
...
```

在 Amazon CLI 中使用速记语法

Amazon Command Line Interface（Amazon CLI）可以接受 JSON 格式的许多选项。但是，在命令行上输入较大的 JSON 列表或结构会比较繁琐。为了简化此过程，Amazon CLI 还支持一种速记语法，允许采用比完整 JSON 格式更简单的方式表示选项参数。

主题

- 使用键值对组织参数。
- 将文件作为速记语法值进行加载
- 将速记语法与 Amazon CLI 结合使用

使用键值对组织参数。

通过 Amazon CLI 中的速记语法，用户更容易输入平面（非嵌套结构）参数。格式采用以逗号分隔的键值对列表。请务必使用适用于终端的引用以及转义规则，因为速记语法是字符串。

Linux or macOS

```
--option key1=value1,key2=value2,key3=value3
```

等同于以下 JSON 格式的示例。

```
--option '{"key1":"value1","key2":"value2","key3":"value3"}'
```

Windows

```
--option "key1=value1,key2=value2,key3=value3"
```

等同于以下 JSON 格式的示例。

```
--option '{"key1":"value1","key2":"value2","key3":"value3"}'
```

各逗号分隔的键值对之间不能有空格。下面的示例 Amazon DynamoDB `update-table` 命令包含采用速记语法指定的 `--provisioned-throughput` 选项。

```
$ aws dynamodb update-table \
    --provisioned-throughput ReadCapacityUnits=15,WriteCapacityUnits=10 \
```

```
    --table-name MyDDBTable
```

该示例等同于以下 JSON 格式的示例。

```
$ aws dynamodb update-table \
    --provisioned-throughput '{"ReadCapacityUnits":15,"WriteCapacityUnits":10}' \
    --table-name MyDDBTable
```

将文件作为速记语法值进行加载

当值较大或较复杂时，通常更易以值的形式加载。要将文件作为速记语法值进行加载，格式将略有变化。使用 @= 运算符代替 = 运算符，而不使用 key=value。@= 告知 Amazon CLI，值应作为文件路径而不是字符串进行读取。在使用以速记语法加载文件时，常见的 Amazon CLI 文件格式设置规则将适用。以下示例显示一个键值对，该键值对为其值加载文件。

Linux or macOS

```
--option key@=file://template.txt
```

Windows

```
--option "key1@=file://template.txt"
```

以下示例演示如何为 `aws rolesanywhere create-trust-anchor` 命令加载证书文件。

```
$ aws rolesanywhere create-trust-anchor --name TrustAnchor \
    --source sourceData={x509CertificateData@=file://root-
ca.crt},sourceType="CERTIFICATE_BUNDLE"  \
    --enabled
```

将速记语法与 Amazon CLI 结合使用

您可以使用两种方法以列表形式指定输入参数：JSON 或速记。使用 Amazon CLI 速记语法，可更方便地传入含有数字、字符串或非嵌套结构的列表。

下面显示了基本格式，列表中的值用单个空格分隔。

```
--option value1 value2 value3
```

该示例等同于以下 JSON 格式的示例。

```
--option '[value1,value2,value3]'
```

如前所述，您可以用速记语法指定数字列表、字符串列表或非嵌套结构的列表。以下是用于 Amazon Elastic Compute Cloud（Amazon EC2）的 `stop-instances` 命令示例，其中，`--instance-ids` 选项的输入参数（字符串列表）采用速记语法指定。

```
$ aws ec2 stop-instances \
    --instance-ids i-1486157a i-1286157c i-ec3a7e87
```

该示例等同于以下 JSON 格式的示例。

```
$ aws ec2 stop-instances \
    --instance-ids '["i-1486157a","i-1286157c","i-ec3a7e87"]'
```

下面的示例显示 Amazon EC2 `create-tags` 命令，该命令针对 `--tags` 选项使用非嵌套结构的列表。`--resources` 选项指定要添加标签的实例的 ID。

```
$ aws ec2 create-tags \
    --resources i-1286157c \
    --tags Key=My1stTag,Value=Value1 Key=My2ndTag,Value=Value2
 Key=My3rdTag,Value=Value3
```

该示例等同于以下 JSON 格式的示例。JSON 参数分多行编写以便于阅读。

```
$ aws ec2 create-tags \
    --resources i-1286157c \
    --tags '[
        {"Key": "My1stTag", "Value": "Value1"},
        {"Key": "My2ndTag", "Value": "Value2"},
        {"Key": "My3rdTag", "Value": "Value3"}
    ]'
```

在 Amazon CLI 中控制命令输出

本部分介绍控制 Amazon Command Line Interface (Amazon CLI) 的输出的不同方式。在终端中自定义 Amazon CLI 输出可以提高可读性，简化脚本自动化，并为浏览大型数据集提供便利。

Amazon CLI 支持多种输出格式，包括 `json`、`text`、 和 `table`。有些服务在服务器端对数据进行了分页，

最后，Amazon CLI 具有服务器端和客户端筛选功能，您可以单独使用一个功能或同时使用这两个功能来筛选 Amazon CLI 输出。

主题

- 敏感输出
- 服务器端与客户端输出选项
- 在 Amazon CLI 中设置输出格式
- 在 Amazon CLI 中使用分页选项
- 在 Amazon CLI 中筛选输出

敏感输出

Amazon CLI 中的某些操作可能会返回可能被视为敏感内容的信息，包括来自环境变量的信息。在某些情况下，泄露这些信息可能构成安全风险；例如，这些信息可能包含在持续集成和持续部署（CI/CD）日志中。因此，请务必核查何时将此类输出作为日志的一部分，并在不需要时隐藏该输出。

有关保护敏感数据的其他信息，请参阅the section called “数据保护”。

考虑下面的最佳实践：

- 请考虑以编程方式从密钥存储库（如 Amazon Secrets Manager）中检索您的密钥。
- 查看构建日志的内容，确保其中不包含敏感信息。考虑使用诸如 `/dev/null` 管道传输或将输出捕获为 bash 或 PowerShell 变量之类的方法，来抑制命令输出。

  以下是将输出（而不是错误）重定向到 `/dev/null` 的 bash 示例：

  ```
  $ aws s3 ls > /dev/null
  ```

  有关抑制终端输出的详细信息，请参阅所用终端的用户文档。
- 考虑日志的访问权限，并根据您的使用案例适当确定访问范围。

服务器端与客户端输出选项

Amazon CLI 具有服务器端和客户端筛选功能，您可以单独使用或结合使用这两项功能来筛选 Amazon CLI 输出。首先处理服务器端筛选，然后返回输出以进行客户端筛选。服务器端筛选由服务 API 提供支持。客户端筛选使用 `--query` 参数由 Amazon CLI 客户端提供支持。

服务器端输出选项是 Amazon Web Services 服务 API 直接支持的功能。任何经过筛选或分页的数据都不会发送到客户端，这可以缩短 HTTP 响应时间，并为较大的数据集提高带宽。

客户端输出选项是由 Amazon CLI 创建的功能。所有数据都发送到客户端，然后 Amazon CLI 会对显示的内容进行筛选或分页。对于较大的数据集，客户端操作不会加快速度或节省带宽。

当服务器端和客户端选项同时使用时，服务器端操作会首先完成，然后发送到客户端进行客户端操作。这利用了服务器端选项可以加快速度和节省带宽的特点，同时使用其他 Amazon CLI 功能来获得所需的输出。

在 Amazon CLI 中设置输出格式

本主题介绍了 Amazon Command Line Interface (Amazon CLI) 的不同输出格式。Amazon CLI 支持以下输出格式：

- **`json`** – 输出采用 JSON 字符串的格式。
- **`text`** – 输出采用多个制表符分隔字符串值行的格式。这对于将输出传递到文本处理器（如 `grep`、`sed` 或 `awk`）很有用。
- **`table`** – 输出采用表格形式，使用字符 +|- 以形成单元格边框。它通常以“人性化”格式呈现信息，这种格式比其他格式更容易阅读，但从编程方面来讲不是那么有用。

如何选择输出格式

正如配置主题所述，输出格式可通过三种不同方式指定：

- 在 **`config`** 文件的命名配置文件中使用 **`output`** 选项 – 以下示例将默认输出格式设置为 `text`。

```
[default]
output=text
```

- 使用 **`AWS_DEFAULT_OUTPUT`** 环境变量 – 对于此命令行会话中的命令，以下输出将格式设置为 `table`，直到更改此变量或会话结束。使用此环境变量将覆盖在 `config` 文件中设置的任何值。

```
$ export AWS_DEFAULT_OUTPUT="table"
```

- 在命令行上使用 **`--output`** 选项 – 以下示例仅将这一个命令的输出设置为 `json`。对此命令使用此选项将覆盖任何当前设置的环境变量或 `config` 文件中的值。

```
$ aws swf list-domains --registration-status REGISTERED --output json
```

> **Important**
>
> 指定的输出类型更改--query 选项的运行方式：
>
> - 如果您指定 --output text，则在应用 --query 筛选条件之前，输出采用分页方式，并且 Amazon CLI 会一次性地在输出的每个页面 上运行查询。因此，查询在每个页面上包括第一个匹配元素，这可能会导致意外的额外输出。要进一步筛选输出，您可以使用其他命令行工具，例如 head 或 tail。
> - 如果您指定 --output json、，则在应用 --query 筛选条件之前，输出会完全处理为单个本机结构。Amazon CLI 仅针对整个结构运行查询一次，同时生成筛选的结果，然后将结果输出。

JSON 输出格式

JSON 是 Amazon CLI 的默认输出格式。大多数编程语言可以使用内置函数或公开提供的库轻松解码 JSON 字符串。您可以通过强有力的方式将 JSON 输出与 --query 选项结合使用，以筛选和格式化 Amazon CLI JSON 格式的输出。

对于您可能无法使用 --query 执行的更高级的筛选，可以考虑使用 jq，这是一个命令行 JSON 处理器。您可以在以下网址下载它并找到正式的教程：http://stedolan.github.io/jq/。

以下是 JSON 输出的示例。

```
$ aws iam list-users --output json
```

```
{
    "Users": [
        {
            "Path": "/",
            "UserName": "Admin",
            "UserId": "AIDA1111111111EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Admin",
            "CreateDate": "2014-10-16T16:03:09+00:00",
            "PasswordLastUsed": "2016-06-03T18:37:29+00:00"
        },
        {
            "Path": "/backup/",
            "UserName": "backup-user",
            "UserId": "AIDA2222222222EXAMPLE",
```

```
            "Arn": "arn:aws:iam::123456789012:user/backup/backup-user",
            "CreateDate": "2019-09-17T19:30:40+00:00"
        },
        {
            "Path": "/",
            "UserName": "cli-user",
            "UserId": "AIDA3333333333EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/cli-user",
            "CreateDate": "2019-09-17T19:11:39+00:00"
        }
    ]
}
```

文本输出格式

`text` 格式将 Amazon CLI 输出组织为制表符分隔的行。此格式适合在传统 Unix 文本工具（如 `grep`、`sed` 和 `awk`）和由 PowerShell 执行的文本处理中使用。

`text` 输出格式遵循以下所示的基本结构。这些列根据底层 JSON 对象相应的键名称按字母顺序排序。

```
IDENTIFIER  sorted-column1 sorted-column2
IDENTIFIER2 sorted-column1 sorted-column2
```

下面是 `text` 输出的一个示例。每个字段都用标签与其他字段分开，并带有一个额外的标签，其中有一个空字段。

```
$ aws iam list-users --output text
```

```
USERS   arn:aws:iam::123456789012:user/Admin                2014-10-16T16:03:09+00:00
 2016-06-03T18:37:29+00:00   /          AIDA1111111111EXAMPLE   Admin
USERS   arn:aws:iam::123456789012:user/backup/backup-user   2019-09-17T19:30:40+00:00
                            /backup/   AIDA2222222222EXAMPLE   backup-user
USERS   arn:aws:iam::123456789012:user/cli-user             2019-09-17T19:11:39+00:00
                            /          AIDA3333333333EXAMPLE   cli-user
```

第四列是 `PasswordLastUsed` 字段，它对于最后两个条目为空，因为这些用户从未登录过 Amazon Web Services 管理控制台。

> **Important**
>
> 我们强烈建议，如果您指定 *`text`* 输出，则也始终使用 *`--query`* 选项以确保行为一致。

这是因为文本格式按由 Amazon 服务返回的基础 JSON 对象的键名称以字母顺序对输出列进行排序，而类似的资源可能没有相同的键名称。例如，基于 Linux 的 Amazon EC2 实例的 JSON 表示形式中的元素在基于 Windows 的实例的 JSON 表示形式中可能不存在，反之亦然。此外，资源可能在未来的更新中添加或移除键/值元素，从而修改列的顺序。因此，可以使用 `--query` 补充 `text` 输出的功能，以提供对输出格式的完全控制。
在以下示例中，命令指定要显示的元素，并使用列表表示法 `[key1, key2, ...]` 来定义列的顺序。这可让您十分放心：正确的键值始终显示在预期的列中。最后请注意，对于不存在的键，Amazon CLI 将输出 None 作为键值。

```
$ aws iam list-users --output text --query 'Users[*].
[UserName,Arn,CreateDate,PasswordLastUsed,UserId]'
```

```
Admin         arn:aws:iam::123456789012:user/Admin
 2014-10-16T16:03:09+00:00   2016-06-03T18:37:29+00:00   AIDA1111111111EXAMPLE
backup-user   arn:aws:iam::123456789012:user/backup-user
 2019-09-17T19:30:40+00:00   None                        AIDA2222222222EXAMPLE
cli-user      arn:aws:iam::123456789012:user/cli-backup
 2019-09-17T19:11:39+00:00   None                        AIDA3333333333EXAMPLE
```

以下示例显示如何将 `grep` 和 `awk` 与来自 `text` 命令的 `aws ec2 describe-instances` 输出结合使用。第一个命令在 `text` 输出中显示每个实例的可用区、当前状态和实例 ID。第二个命令处理该输出，以仅输出在 `us-west-2a` 可用区中运行的所有实例的实例 ID。

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text
```

```
us-west-2a      running i-4b41a37c
us-west-2a      stopped i-a071c394
us-west-2b      stopped i-97a217a0
us-west-2a      running i-3045b007
us-west-2a      running i-6fc67758
```

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text | grep us-west-2a |
 grep running | awk '{print $3}'
```

```
i-4b41a37c
```

```
i-3045b007
i-6fc67758
```

以下示例更进一步，不仅说明如何筛选输出，还介绍如何使用该输出自动更改每个已停止实例的实例类型。

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].[State.Name,
 InstanceId]' --output text |
> grep stopped |
> awk '{print $2}' |
> while read line;
> do aws ec2 modify-instance-attribute --instance-id $line --instance-type '{"Value":
 "m1.medium"}';
> done
```

text 输出也适用于 PowerShell。因为 text 输出中的列使用制表符分隔，所以可以轻松地使用 PowerShell 的 `t 分隔符将输出拆分为数组。以下命令在第一列 (InstanceId) 与字符串 AvailabilityZone 匹配的情况下显示第三列 (us-west-2a) 的值。

```
PS C:\>aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text |
%{if ($_.split("`t")[0] -match "us-west-2a") { $_.split("`t")[2]; } }
```

```
-4b41a37c
i-a071c394
i-3045b007
i-6fc67758
```

注意，尽管上一个示例的确显示了如何使用 --query 参数解析底层 JSON 对象和提取所需的列，但是 PowerShell 有自己处理 JSON 的功能（如果跨平台兼容性不是问题）。不像大多数命令 shell 需要的那样将输出处理为文本，PowerShell 让您可以使用 ConvertFrom-JSON cmdlet 生成具有层次结构的对象。然后，您可以直接从该对象访问所需的成员。

```
(aws ec2 describe-instances --output json | ConvertFrom-
Json).Reservations.Instances.InstanceId
```

> **Tip**
>
> 如果使用 --query 参数输出文本并将输出筛选到单个字段，则输出是单行制表符分隔值。要将每个值放到单独的行上，可以将输出字段放在括号中，如以下示例所示。

制表符分隔的单行输出：

```
$ aws iam list-groups-for-user --user-name susan  --output text --query
 "Groups[].GroupName"
```

```
HRDepartment    Developers      SpreadsheetUsers  LocalAdmins
```

通过将 [GroupName] 放在括号中，让每个值都在自己的行上：

```
$ aws iam list-groups-for-user --user-name susan  --output text --query
 "Groups[].[GroupName]"
```

```
HRDepartment
Developers
SpreadsheetUsers
LocalAdmins
```

表输出格式

table 格式以表格形式生成复杂 Amazon CLI 输出的易于人阅读的表示。

```
$ aws iam list-users --output table
```

```
-----------------------------------------------------------------------------------------------
|
 ListUsers                                                                    |
+-----------------------------------------------------------------------------------------------
+
||
 Users                                                                    ||
|+-----------------------------------------------------+---------------------------
+---------------------------+----------+-----------------------+-------------+|
||                        Arn                        |       CreateDate          |
 PasswordLastUsed       |   Path   |        UserId         |   UserName  ||
|+-----------------------------------------------------+---------------------------
+---------------------------+----------+-----------------------+-------------+|
||  arn:aws:iam::123456789012:user/Admin              | 2014-10-16T16:03:09+00:00 |
 2016-06-03T18:37:29+00:00 | /        | AIDA1111111111EXAMPLE | Admin       ||
```

```
||  arn:aws:iam::123456789012:user/backup/backup-user | 2019-09-17T19:30:40+00:00 |
                      | /backup/ | AIDA2222222222EXAMPLE | backup-user ||
||  arn:aws:iam::123456789012:user/cli-user           | 2019-09-17T19:11:39+00:00 |
                      | /        | AIDA3333333333EXAMPLE | cli-user    ||
+-------------------------------------------------------------------------------------------
+
```

您可以将 --query 选项与 table 格式结合使用，以显示从原始输出中预先选择的一系列元素。请注意字典和列表表示法之间的输出区别：在第一个示例中，列名按字母顺序排序；在第二个示例中，未命名的列按用户指定的顺序排序。有关 --query 选项的更多信息，请参阅 在 Amazon CLI 中筛选输出。

```
$ aws ec2 describe-volumes --query 'Volumes[*].
{ID:VolumeId,InstanceId:Attachments[0].InstanceId,AZ:AvailabilityZone,Size:Size}' --
output table
```

```
-------------------------------------------------------
|                  DescribeVolumes                    |
+------------+----------------+--------------+-------+
|     AZ     |      ID        | InstanceId   | Size  |
+------------+----------------+--------------+-------+
|  us-west-2a|  vol-e11a5288  |  i-a071c394  |  30   |
|  us-west-2a|  vol-2e410a47  |  i-4b41a37c  |  8    |
+------------+----------------+--------------+-------+
```

```
$ aws ec2 describe-volumes --query 'Volumes[*].
[VolumeId,Attachments[0].InstanceId,AvailabilityZone,Size]' --output table
```

```
-----------------------------------------------------
|                 DescribeVolumes                   |
+--------------+--------------+--------------+-----+
|  vol-e11a5288|  i-a071c394  |  us-west-2a  |  30 |
|  vol-2e410a47|  i-4b41a37c  |  us-west-2a  |  8  |
+--------------+--------------+--------------+-----+
```

在 Amazon CLI 中使用分页选项

本主题介绍分页 Amazon Command Line Interface（Amazon CLI）的输出的不同方式。

服务器端分页

对于返回大型项目列表的大多数命令，Amazon CLI 提供了多个服务器端选项，当 Amazon CLI 调用服务 API 以填充此列表时，您可以使用这些选项控制输出中包括的项目数。Amazon CLI 中的服务器端分页由 Amazon 服务 API 启用，因此，这些选项只有在服务 API 启用它们时才生效。

大多数 Amazon CLI 命令的选项包括：

- 如何使用 --no-paginate 参数
- 如何使用 --page-size 参数
- 如何使用 --max-items 参数
- 如何使用 --starting-token 参数

默认情况下，Amazon CLI 使用由单个服务确定的页面大小并检索所有可用项目。例如，Amazon S3 的原定设置页面大小为 1000。如果您在包含 3500 个对象的 Amazon S3 存储桶上运行 `aws s3api list-objects`，则 Amazon CLI 将自动对 Amazon S3 发出四次调用，以在后台处理服务特定分页逻辑并在最终输出中返回所有 3500 个对象。

有关特定命令是否具有服务器端分页的信息，请参阅《Amazon CLI 参考指南》。

如何使用 --no-paginate 参数

`--no-paginate` 选项在客户端禁用以下分页标记。使用命令时，默认情况下 Amazon CLI 会自动执行多次调用，返回所有可能的结果来创建分页。每页显示一次调用的结果。禁用分页时，Amazon CLI 只执行一次调用，显示命令结果的第一页。

例如，如果您在包含 3500 个对象的 Amazon S3 存储桶上运行 `aws s3api list-objects`，则 Amazon CLI 只会执行对 Amazon S3 的第一个调用，并在最终输出中仅返回前 1000 个对象。

```
$ aws s3api list-objects \
    --bucket amzn-s3-demo-bucket \
    --no-paginate
{
    "Contents": [
...
```

如何使用 --page-size 参数

如果您在对大量资源运行列表命令时发现问题，则表明原定设置页面大小可能过大。这可能会导致对 Amazon 服务的调用超过允许的最大时间并生成“超时”错误。您可以使用 `--page-size` 选项来指定

Amazon CLI 从对 Amazon 服务的每个调用请求数量较少的项目。Amazon CLI 仍将检索完整列表，但会在后台执行大量服务 API 调用，并减少每次调用时检索的项目数。这样，各个调用成功的可能性会更高且不会发生超时。更改页面大小不会影响输出；它只影响生成输出所需进行的 API 调用数。

```
$ aws s3api list-objects \
    --bucket amzn-s3-demo-bucket \
    --page-size 100
{
    "Contents": [
...
```

如何使用 --max-items 参数

要在 Amazon CLI 输出中一次包括更少的项目，请使用 `--max-items` 选项。Amazon CLI 仍会按之前所述使用该服务处理分页，但只打印您指定的一次检索的项目数。

```
$ aws s3api list-objects \
    --bucket amzn-s3-demo-bucket \
    --max-items 100
{
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==",
    "Contents": [
...
```

如何使用 --starting-token 参数

如果项目输出的数量（`--max-items`）少于基础 API 调用所返回的项目总数，则输出将包含您可传递到后续命令的 `NextToken` 以检索下一组项目。以下示例显示如何使用上一示例返回的 `NextToken` 值，并使您能够检索接下来的 100 个项目。

> **Note**
>
> 参数 `--starting-token` 不能为空。如果上一个命令未返回 `NextToken` 值，则没有更多项目可返回，您不需要再次调用该命令。

```
$ aws s3api list-objects \
    --bucket amzn-s3-demo-bucket \
    --max-items 100 \
    --starting-token eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==
{
```

```
    "Contents": [
...
```

每次调用指定的 Amazon 服务时返回项目的顺序可能不同。如果您为 `--page-size` 和 `--max-items` 指定不同的值，您可能会获得意外结果（项目缺失或重复）。为防止出现这种情况，请对 `--page-size` 和 `--max-items` 使用相同的数字，以同步 Amazon CLI 的分页与基础服务的分页。您还可以检索整个列表并在本地执行任何必需的分页操作。

在 Amazon CLI 中筛选输出

Amazon Command Line Interface（Amazon CLI）具有服务器端和客户端筛选功能，您可以单独使用或结合使用这两项功能来筛选 Amazon CLI 输出。首先处理服务器端筛选，然后返回输出以进行客户端筛选。

- 服务器端筛选由 API 提供支持，通常使用 `--filter` 参数来实现筛选。该服务仅返回匹配结果，这可以加快大型数据集的 HTTP 响应时间。
- 客户端筛选使用 `--query` 参数由 Amazon CLI 客户端提供支持。此参数具有服务器端筛选可能没有的功能。

主题

- 服务器端筛选
- 客户端筛选
- 结合服务器端和客户端筛选
- 其他资源

服务器端筛选

Amazon CLI 中的服务器端筛选由 Amazon 服务 API 提供。该 Amazon 服务仅返回 HTTP 响应中与筛选条件匹配的记录，这可以加快大型数据集的 HTTP 响应时间。由于服务器端筛选由服务 API 定义，因此参数名称和函数因服务而异。用于筛选的一些常见参数名称包括：

- `--filter` 例如 ses 和 ce。
- `--filters` 例如 ec2、autoscaling 和 rds。
- 以单词 `filter` 开头的名称，例如 `--filter-expression` 用于 `aws dynamodb scan` 命令。

有关特定命令是否具有服务器端筛选和筛选规则的信息，请参阅 Amazon CLI 参考指南。

客户端筛选

Amazon CLI 使用 `--query` 参数提供内置的基于 JSON 的客户端筛选功能。该 `--query` 参数是一个功能强大的工具，可用来自定义输出的内容和样式。该 `--query` 参数可获取从服务器返回的 HTTP 响应，并在显示结果之前对结果进行筛选。由于在筛选之前将整个 HTTP 响应发送到客户端，因此客户端筛选速度可能比大型数据集的服务器端筛选更慢。

查询使用 JMESPath 语法来创建用于筛选输出的表达式。要了解 JMESPath 语法，请参阅 JMESPath 网站上的教程。

> **Important**
>
> 指定的输出类型更改`--query` 选项的运行方式：
>
> - 如果您指定 `--output text`，则在应用 `--query` 筛选条件之前，输出采用分页方式，并且 Amazon CLI 会一次性地在输出的每个页面 上运行查询。因此，查询在每个页面上包括第一个匹配元素，这可能会导致意外的额外输出。要进一步筛选输出，您可以使用其他命令行工具，例如 `head` 或 `tail`。
> - 如果您指定 `--output json`、，则在应用 `--query` 筛选条件之前，输出会完全处理为单个本机结构。Amazon CLI 仅针对整个结构运行查询一次，同时生成筛选的结果，然后将结果输出。

客户端筛选主题

- 开始之前
- 标识符
- 从列表中选择
- 筛选嵌套数据
- 展平结果
- 筛选特定值
- 传输表达式
- 筛选多个标识符值
- 将标签添加到标识符值
- 函数
- 高级 --query 示例

开始之前

> **Note**
>
> 这些筛选表达式示例是为基本 Linux 式 Shell 编写的。在使用这些示例时，请务必为终端 Shell 使用正确的引用规则。终端解释输入的方式可能会极大地改变发送到 Amazon CLI 的内容。终端读取单引号 '、双引号 " 或反引号 ` 的方式会更改内容的读取方式。
> 有关更多信息，请参阅 the section called “含字符串的引号”。

以下 JSON 输出显示了 `--query` 参数可以生成的示例。此输出描述了附加到单独 Amazon EC2 实例的三个 Amazon EBS 卷。

示例输出

```
$ aws ec2 describe-volumes
{
  "Volumes": [
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-17T00:55:03.000Z",
          "InstanceId": "i-a071c394",
          "VolumeId": "vol-e11a5288",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-e11a5288",
      "State": "in-use",
      "SnapshotId": "snap-f23ec1c8",
      "CreateTime": "2013-09-17T00:55:03.000Z",
      "Size": 30
    },
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-18T20:26:16.000Z",
```

```
          "InstanceId": "i-4b41a37c",
          "VolumeId": "vol-2e410a47",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-2e410a47",
      "State": "in-use",
      "SnapshotId": "snap-708e8348",
      "CreateTime": "2013-09-18T20:26:15.000Z",
      "Size": 8
    },
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2020-11-20T19:54:06.000Z",
          "InstanceId": "i-1jd73kv8",
          "VolumeId": "vol-a1b3c7nd",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-a1b3c7nd",
      "State": "in-use",
      "SnapshotId": "snap-234087fb",
      "CreateTime": "2020-11-20T19:54:05.000Z",
      "Size": 15
    }
  ]
}
```

标识符

标识符是输出值的标签。创建筛选条件时，您可以使用标识符缩小查询结果的范围。在以下输出示例中，所有标识符（如 Volumes、AvailabilityZone 和 AttachTime）都将突出显示。

```
$ aws ec2 describe-volumes
{
```

```
  "Volumes": [
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-17T00:55:03.000Z",
          "InstanceId": "i-a071c394",
          "VolumeId": "vol-e11a5288",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-e11a5288",
      "State": "in-use",
      "SnapshotId": "snap-f23ec1c8",
      "CreateTime": "2013-09-17T00:55:03.000Z",
      "Size": 30
    },
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-18T20:26:16.000Z",
          "InstanceId": "i-4b41a37c",
          "VolumeId": "vol-2e410a47",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-2e410a47",
      "State": "in-use",
      "SnapshotId": "snap-708e8348",
      "CreateTime": "2013-09-18T20:26:15.000Z",
      "Size": 8
    },
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2020-11-20T19:54:06.000Z",
```

```
            "InstanceId": "i-1jd73kv8",
            "VolumeId": "vol-a1b3c7nd",
            "State": "attached",
            "DeleteOnTermination": true,
            "Device": "/dev/sda1"
          }
        ],
        "VolumeType": "standard",
        "VolumeId": "vol-a1b3c7nd",
        "State": "in-use",
        "SnapshotId": "snap-234087fb",
        "CreateTime": "2020-11-20T19:54:05.000Z",
        "Size": 15
      }
    ]
}
```

有关更多信息，请参阅 JMESPath 网站上的标识符。

从列表中选择

列表或数组是后跟方括号“[”的标识符，例如 Volumes 中的 Attachments 和 the section called “开始之前”。

语法

```
<listName>[ ]
```

要筛选数组的所有输出，可以使用通配符表示法。通配符表达式是用于使用 * 表示法返回元素的表达式。

以下示例查询所有 Volumes 内容。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
```

```
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  }
]
```

要按索引查看数组中的特定卷，请调用数组索引。例如，Volumes 数组中的第一个项目的索引为 0，返回 Volumes[0] 查询。有关数组索引的更多信息，请参阅 JMESPath 网站上的索引表达式。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[0]'
{
  "AvailabilityZone": "us-west-2a",
  "Attachments": [
    {
```

```
      "AttachTime": "2013-09-17T00:55:03.000Z",
      "InstanceId": "i-a071c394",
      "VolumeId": "vol-e11a5288",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  "VolumeType": "standard",
  "VolumeId": "vol-e11a5288",
  "State": "in-use",
  "SnapshotId": "snap-f23ec1c8",
  "CreateTime": "2013-09-17T00:55:03.000Z",
  "Size": 30
}
```

要按索引查看特定范围的卷，请使用 `slice` 与以下语法，其中 start 是起始数组索引，stop 是筛选条件停止处理的索引，step 是跳过间隔。

语法

```
<arrayName>[<start>:<stop>:<step>]
```

如果 Slice 表达式中忽略了其中任何一个，它们将使用以下默认值：

- Start – 列表中的第一个索引，0。
- Stop – 列表中的最后一个索引。
- Step – 没有跳过步骤，其中值为 1。

要仅返回前两个卷，请使用起始值 0、停止值 2 和步长值 1，如以下示例所示。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[0:2:1]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
```

```
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-18T20:26:16.000Z",
        "InstanceId": "i-4b41a37c",
        "VolumeId": "vol-2e410a47",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-2e410a47",
    "State": "in-use",
    "SnapshotId": "snap-708e8348",
    "CreateTime": "2013-09-18T20:26:15.000Z",
    "Size": 8
  }
]
```

由于此示例包含默认值，因此您可以将 Slice 从 `Volumes[0:2:1]` 缩短到 `Volumes[:2]`。

以下示例省略了默认值，并返回整个数组中的每两个卷。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[::2]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
```

```
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  }
]
```

Steps 还可以使用负数按数组的相反顺序进行筛选，如以下示例所示。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[::-2]'
[
  {
    "AvailabilityZone": "us-west-2a",
```

```
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  }
]
```

有关更多信息，请参阅 JMESPath 网站上的 Slices。

筛选嵌套数据

要缩小嵌套值 `Volumes[*]` 的筛选范围，您可以通过附加句点和筛选条件来使用子表达式。

语法

```
<expression>.<expression>
```

以下示例显示了所有卷的所有 Attachments 信息。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments'
[
  [
    {
      "AttachTime": "2013-09-17T00:55:03.000Z",
      "InstanceId": "i-a071c394",
      "VolumeId": "vol-e11a5288",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  [
    {
      "AttachTime": "2013-09-18T20:26:16.000Z",
      "InstanceId": "i-4b41a37c",
      "VolumeId": "vol-2e410a47",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  [
    {
      "AttachTime": "2020-11-20T19:54:06.000Z",
      "InstanceId": "i-1jd73kv8",
      "VolumeId": "vol-a1b3c7nd",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ]
]
```

要进一步筛选嵌套值，请为每个嵌套标识符附加表达式。以下示例列出了所有 State 的 Volumes。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[*].State'
[
  [
    "attached"
  ],
  [
    "attached"
  ],
  [
    "attached"
  ]
]
```

展平结果

有关更多信息，请参阅 JMESPath 网站上的 SubExpressions。

您可以通过删除导致 `Volumes[*].Attachments[*].State` 查询的通配符表示法来展平 `Volumes[*].Attachments[].State` 的结果。展平操作通常有助于提高结果的可读性。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[].State'
[
  "attached",
  "attached",
  "attached"
]
```

有关更多信息，请参阅 JMESPath 网站上的展平。

筛选特定值

要筛选列表中的特定值，可以使用筛选条件表达式，如以下语法所示。

语法

```
? <expression> <comparator> <expression>]
```

表达式比较器包括 ==、!=、<、<=、> 和 >=。以下示例为 `VolumeIdsVolumes` 中的所有 `Attached` 筛选 `State`。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[?State==`attached`].VolumeId'
[
  [
    "vol-e11a5288"
  ],
  [
    "vol-2e410a47"
  ],
  [
    "vol-a1b3c7nd"
  ]
]
```

然后可以将其展平，从而生成以下示例。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[?State==`attached`].VolumeId[]'
[
  "vol-e11a5288",
  "vol-2e410a47",
  "vol-a1b3c7nd"
]
```

以下示例筛选了尺寸小于 20 的所有 VolumeIds 的 Volumes。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[?Size < `20`].VolumeId'
[
  "vol-2e410a47",
  "vol-a1b3c7nd"
]
```

有关更多信息，请参阅 JMESPath 网站上的筛选表达式。

传输表达式

您可以将筛选器的结果传输到新列表中，然后通过以下语法使用另一个表达式筛选结果：

语法

```
<expression> | <expression>]
```

以下示例获取 Volumes[*].Attachments[].InstanceId 表达式的筛选结果并在数组中输出第一个结果。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[].InstanceId | [0]'
"i-a071c394"
```

本示例首先通过以下表达式创建数组来实现此目的。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Attachments[].InstanceId'
"i-a071c394",
  "i-4b41a37c",
  "i-1jd73kv8"
```

然后返回该数组中的第一个元素。

```
"i-a071c394"
```

有关更多信息，请参阅 JMESPath 网站上的传输表达式。

筛选多个标识符值

要筛选多个标识符，您可以使用以下语法使用多选列表：

语法

```
<listName>[].[<expression>, <expression>]
```

在以下示例中，VolumeId 和 VolumeType 在 Volumes 列表中进行筛选，生成以下表达式。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[].[VolumeId, VolumeType]'
[
  [
    "vol-e11a5288",
    "standard"
  ],
  [
    "vol-2e410a47",
    "standard"
  ],
```

```
  [
    "vol-a1b3c7nd",
    "standard"
  ]
]
```

要将嵌套数据添加到列表中，请添加另一个多选列表。下面的示例通过在嵌套 InstanceId 列表中筛选 State 和 Attachments 在上一个示例中进行了扩展。这将产生以下表达式。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[].[VolumeId, VolumeType, Attachments[].[InstanceId, State]]'
[
  [
    "vol-e11a5288",
    "standard",
    [
      [
        "i-a071c394",
        "attached"
      ]
    ]
  ],
  [
    "vol-2e410a47",
    "standard",
    [
      [
        "i-4b41a37c",
        "attached"
      ]
    ]
  ],
  [
    "vol-a1b3c7nd",
    "standard",
    [
      [
        "i-1jd73kv8",
        "attached"
      ]
    ]
  ]
]
```

为了更具可读性，请按以下示例所示展开表达式。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[].[VolumeId, VolumeType, Attachments[].[InstanceId, State][]][]'
[
  "vol-e11a5288",
  "standard",
  [
    "i-a071c394",
    "attached"
  ],
  "vol-2e410a47",
  "standard",
  [
    "i-4b41a37c",
    "attached"
  ],
  "vol-a1b3c7nd",
  "standard",
  [
    "i-1jd73kv8",
    "attached"
  ]
]
```

有关更多信息，请参阅 JMESPath 网站上的多选列表。

将标签添加到标识符值

为了使此输出更易于读取，请使用具有以下语法的多选哈希。

语法

```
<listName>[].{<label>: <expression>, <label>: <expression>}
```

您的标识符标签不必与标识符的名称相同。以下示例为 VolumeType 值使用标签 VolumeType。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[].{VolumeType: VolumeType}'
[
  {
    "VolumeType": "standard",
```

```
  },
  {
    "VolumeType": "standard",
  },
  {
    "VolumeType": "standard",
  }
]
```

为简单起见，以下示例保留每个标签的标识符名称，并显示所有卷的 VolumeId、VolumeType、InstanceId 和 State：

```
$ aws ec2 describe-volumes \
    --query 'Volumes[].{VolumeId: VolumeId, VolumeType: VolumeType, InstanceId:
 Attachments[0].InstanceId, State: Attachments[0].State}'
[
  {
    "VolumeId": "vol-e11a5288",
    "VolumeType": "standard",
    "InstanceId": "i-a071c394",
    "State": "attached"
  },
  {
    "VolumeId": "vol-2e410a47",
    "VolumeType": "standard",
    "InstanceId": "i-4b41a37c",
    "State": "attached"
  },
  {
    "VolumeId": "vol-a1b3c7nd",
    "VolumeType": "standard",
    "InstanceId": "i-1jd73kv8",
    "State": "attached"
  }
]
```

有关更多信息，请参阅 JMESPath 网站 上的多选哈希。

函数

JMESPath 语法包含许多可用于查询的函数。有关 JMESPath 函数的信息，请参阅 JMESPath 网站上的内置函数。

为了演示如何将函数合并到查询中，以下示例使用了 sort_by 函数。该 sort_by 函数使用以下语法将表达式作为排序键对数组进行排序：

语法

```
sort_by(<listName>, <sort expression>)[].<expression>
```

以下示例使用先前的多选哈希示例并按照 VolumeId 对输出进行排序。

```
$ aws ec2 describe-volumes \
    --query 'sort_by(Volumes, &VolumeId)[].{VolumeId: VolumeId, VolumeType: VolumeType,
 InstanceId: Attachments[0].InstanceId, State: Attachments[0].State}'
[
  {
    "VolumeId": "vol-2e410a47",
    "VolumeType": "standard",
    "InstanceId": "i-4b41a37c",
    "State": "attached"
  },
  {
    "VolumeId": "vol-a1b3c7nd",
    "VolumeType": "standard",
    "InstanceId": "i-1jd73kv8",
    "State": "attached"
  },
  {
    "VolumeId": "vol-e11a5288",
    "VolumeType": "standard",
    "InstanceId": "i-a071c394",
    "State": "attached"
  }
]
```

有关更多信息，请参阅 JMESPath 网站上的排序依据。

高级 **--query** 示例

从特定项目中提取信息

以下示例使用 --query 参数在列表中查找特定的项目，然后提取该项目的的信息。此示例列出了与指定的服务端点相关联的所有 AvailabilityZones。它从 ServiceDetails 列表中提取具有指定 ServiceName 的项目，然后输出该选定项目的 AvailabilityZones 字段。

```
$ aws --region us-east-1 ec2 describe-vpc-endpoint-services \
    --query 'ServiceDetails[?ServiceName==`com.amazonaws.us-
east-1.ecs`].AvailabilityZones'
[
    [
        "us-east-1a",
        "us-east-1b",
        "us-east-1c",
        "us-east-1d",
        "us-east-1e",
        "us-east-1f"
    ]
]
```

在指定创建日期之后显示快照

以下示例显示如何列出在指定日期之后创建的所有快照，从而在输出中仅包括几个可用字段。

```
$ aws ec2 describe-snapshots --owner self \
    --output json \
    --query 'Snapshots[?StartTime>=`2018-02-07`].
{Id:SnapshotId,VId:VolumeId,Size:VolumeSize}'
[
    {
        "id": "snap-0effb42b7a1b2c3d4",
        "vid": "vol-0be9bb0bf12345678",
        "Size": 8
    }
]
```

显示最新 AMI

以下示例列出了您创建的五个最新 Amazon 机器映像（AMI），从最新到最旧排序。

```
$ aws ec2 describe-images \
    --owners self \
    --query 'reverse(sort_by(Images,&CreationDate))[:5].{id:ImageId,date:CreationDate}'
[
    {
        "id": "ami-0a1b2c3d4e5f60001",
        "date": "2018-11-28T17:16:38.000Z"
    },
```

```
    {
        "id": "ami-0a1b2c3d4e5f60002",
        "date": "2018-09-15T13:51:22.000Z"
    },
    {
        "id": "ami-0a1b2c3d4e5f60003",
        "date": "2018-08-19T10:22:45.000Z"
    },
    {
        "id": "ami-0a1b2c3d4e5f60004",
        "date": "2018-05-03T12:04:02.000Z"
    },
    {
        "id": "ami-0a1b2c3d4e5f60005",
        "date": "2017-12-13T17:16:38.000Z"
    }
]
```

显示运行状况不佳的 Auto Scaling 实例

以下示例仅显示指定 Auto Scaling 组中任何运行状况不佳的实例的 `InstanceId`。

```
$ aws autoscaling describe-auto-scaling-groups \
    --auto-scaling-group-name My-AutoScaling-Group-Name \
    --output text \
    --query 'AutoScalingGroups[*].Instances[?HealthStatus==`Unhealthy`].InstanceId'
```

包括带指定标签的卷

以下示例描述了所有带 `test` 标签的实例。只要附加的卷旁边 `test` 还有另一个标签，那么结果中仍会返回该卷。

下面的表达式在数组中返回带 `test` 标签的所有标签。任何不是 `test` 标签的标签都包含 `null` 值。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Tags[?Value == `test`]'
```

排除具有指定标签的卷

以下示例描述了所有不带 `test` 标签的实例。使用简单 `?Value != `test`` 表达式不适用于排除卷，因为卷可能有多个标签。只要附加的卷旁边 `test` 还有另一个标签，那么结果中仍会返回该卷。

要排除带有 test标签的所有卷，请从下面的表达式开始返回数组中带有该 test 标签的所有标签。任何不是 test 标签的标签都包含 null 值。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[*].Tags[?Value == `test`]'
```

然后使用 test 函数筛选掉所有正数 not_null 结果。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[?!not_null(Tags[?Value == `test`].Value)]'
```

传输结果以展开会导致以下查询的结果。

```
$ aws ec2 describe-volumes \
    --query 'Volumes[?!not_null(Tags[?Value == `test`].Value)] | []'
```

结合服务器端和客户端筛选

您可以同时使用服务器端和客户端筛选。首先完成服务器端筛选，将数据发送到客户端，然后由 --query 参数进行筛选。如果您使用的是大型数据集，首先使用服务器端筛选可以降低每次 Amazon CLI 调用发送给客户端的数据量，同时仍保持客户端筛选提供的强大自定义功能。

以下示例列出了使用服务器端和客户端筛选的 Amazon EC2 卷。该服务在 us-west-2a 可用区中筛选所有附加的卷的列表。--query 参数进一步将输出限制为只有 Size 值大于 50 的卷，并且仅显示具有用户定义名称的指定字段。

```
$ aws ec2 describe-volumes \
    --filters "Name=availability-zone,Values=us-west-2a" "Name=status,Values=attached"
 \
    --query 'Volumes[?Size > `50`].{Id:VolumeId,Size:Size,Type:VolumeType}'
[
    {
        "Id": "vol-0be9bb0bf12345678",
        "Size": 80,
        "VolumeType": "gp2"
    }
]
```

以下示例检索满足多个条件的镜像的列表。然后，它使用 --query 参数按 CreationDate 对输出进行排序，从而仅选择最新的。最终，它显示这一个镜像的 ImageId。

```
$ aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn*gp2" "Name=virtualization-type,Values=hvm"
 "Name=root-device-type,Values=ebs" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text
ami-00ced3122871a4921
```

以下示例通过使用 `length` 计算列表中的数量，以显示超过 1000 IOPS 的可用卷数。

```
$ aws ec2 describe-volumes \
    --filters "Name=status,Values=available" \
    --query 'length(Volumes[?Iops > `1000`])'
3
```

以下示例检索在指定的 Amazon Web Services 区域中使用启动配置（这些配置使用 Amazon CloudFormation 堆栈）的自动扩缩组的名称。

```
$ aws autoscaling describe-auto-scaling-groups --region us-west-2 \
  --filters Name=tag-key,Values=aws:cloudformation:stack-name \
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`].AutoScalingGroupName'
[
    "group-1",
    "group-2",
    "group-3"
]
```

其他资源

JMESPath 终端

JMESPath 终端是一个交互式终端命令，以试验用于客户端筛选的 JMESPath 表达式。使用 `jpterm` 命令，终端会在您键入时显示即时查询结果。您可以直接将 Amazon CLI 输出传输到终端，从而启用高级查询试验。

以下示例将 `aws ec2 describe-volumes` 输出直接传输到 JMESPath 终端。

```
$ aws ec2 describe-volumes | jpterm
```

有关 JMESPath 终端和安装说明的更多信息，请参阅 GitHub 上的 JMESPath 终端。

jq 实用工具

该 jq 实用工具为您提供了一种将客户端输出转换为所需输出格式的方法。有关 jq 和安装说明的更多信息，请参阅 GitHub 上的 jq。

Amazon CLI 中的命令行返回代码

返回代码通常是运行 Amazon Command Line Interface (Amazon CLI) 命令后发送的隐藏代码，该命令可描述命令的状态。您可以使用 echo 命令显示从上一个 Amazon CLI 命令发送的代码，并使用这些代码确定命令是成功还是失败，以及命令可能出错的原因。除了返回代码之外，您还可以运行带有 --debug 开关的命令，查看有关故障的更多详细信息。此开关将生成一个详细报告，描述 Amazon CLI 用于处理命令的步骤以及每个步骤的结果。

要确定 Amazon CLI 命令的返回代码，请在运行 CLI 命令后立即运行以下命令之一。

Linux and macOS

```
$ echo $?
0
```

Windows PowerShell

```
PS> echo $lastexitcode
0
```

Windows Command Prompt

```
C:\> echo %errorlevel%
0
```

以下是运行 Amazon Command Line Interface (Amazon CLI) 命令结束时可能返回的返回代码值。

代码	意义
0	该服务使用 HTTP 响应状态代码 200 进行响应，表示请求发送到的 Amazon CLI 和 Amazon 服务未生成错误。

代码	意义
1	一个或多个 Amazon S3 传输操作失败。仅限 S3 命令。
2	该返回代码的含义取决于命令： • 适用于所有 Amazon CLI 命令 – 无法解析输入的命令。解析失败的原因可能是（但不限于）缺少必需的子命令或参数，或使用了未知的命令或参数。 • 限制为 S3 命令 – 在传输过程中，跳过了标记为要进行传输的一个或多个文件。但是，标记为要进行传输的所有其他文件都已成功传输。传输过程中跳过的文件包括：不存在的文件，作为字符特殊设备、块特殊设备、FIFO 队列或套接字的文件，以及用户没有读取权限的文件。
130	命令已被 SIGINT 中断。这是您通过 `Ctrl+C` 发送的信号，用于取消某个命令。
255	命令失败。Amazon CLI 或将请求发送到的 Amazon 服务生成了错误。

在 Amazon CLI 中创建和使用别名

别名是您可以在 Amazon Command Line Interface（Amazon CLI）中创建的快捷方式，用于缩短您经常使用的命令或脚本。您可以在配置文件夹中的 `alias` 文件中创建别名。

主题

- 先决条件
- 步骤 1：创建别名文件
- 步骤 2：创建别名
- 步骤 3：调用别名
- 别名存储库示例
- 资源

先决条件

要使用别名命令，您需要完成以下操作：

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI和Amazon CLI 身份验证和访问凭证。
- 使用最低 Amazon CLI 版本 1.11.24 或 2.0.0。
- （可选）要使用 Amazon CLI 别名 bash 脚本，必须使用兼容 bash 的终端。

步骤 1：创建别名文件

要创建 `alias` 文件，您可以使用文件导航和文本编辑器，或通过分步过程来使用首选终端。要快速创建别名文件，请使用以下命令块。

Linux and macOS

```
$ mkdir -p ~/.aws/cli
$ echo '[toplevel]' > ~/.aws/cli/alias
```

Windows

```
C:\> md %USERPROFILE%\.aws\cli
C:\> echo [toplevel] > %USERPROFILE%/.aws/cli/alias
```

创建别名文件

1. 在 Amazon CLI 配置文件夹中创建名为 `cli` 的文件夹。默认情况下，配置文件夹为 `~/.aws/`（Linux 或 macOS）和 `%USERPROFILE%\.aws\` (Windows)。您可以通过文件导航或使用以下命令进行创建。

   Linux and macOS

   ```
   $ mkdir -p ~/.aws/cli
   ```

   Windows

   ```
   C:\> md %USERPROFILE%\.aws\cli
   ```

   生成的 `cli` 文件夹默认路径为 `~/.aws/cli/`（Linux 或 macOS）和 `%USERPROFILE%\.aws\cli` (Windows)。

2. 在 cli 文件夹中，创建不带扩展名的名为 alias 的文本文件，然后将 [toplevel] 添加到第一行。您可以通过首选的文本编辑器或使用以下命令创建此文件。

   Linux and macOS

   ```
   $ echo '[toplevel]' > ~/.aws/cli/alias
   ```

   Windows

   ```
   C:\> echo [toplevel] > %USERPROFILE%/.aws/cli/alias
   ```

步骤 2：创建别名

您可以使用基本命令或 bash 脚本创建别名。

创建基本命令别名

您可以在上一步中创建的 alias 文件中使用以下语法来添加命令，从而来创建别名。

语法

```
aliasname = command [--options]
```

aliasname 即您所称的别名。*command* 是您想要调用的命令，它可以包括其他别名。您可以在别名中包含选项或参数，也可以在调用别名时添加选项或参数。

以下示例使用 aws sts get-caller-identity 命令创建别名 aws whoami。由于此别名调用了现有 Amazon CLI 命令，因此您可以编写不带 aws 前缀的命令。

```
whoami = sts get-caller-identity
```

以下示例利用了上一个 whoami 示例并添加了Account 筛选条件和文本 output 选项。

```
whoami2 = sts get-caller-identity --query Account --output text
```

创建子命令别名

> **Note**
>
> 子命令别名功能需要最低 Amazon CLI 版本 1.11.24 或 2.0.0

您可以在上一步中创建的 `alias` 文件中使用以下语法来添加命令，从而为子命令创建别名。

语法

```
[command commandGroup]
aliasname = command [--options]
```

commandGroup 是命令命名空间，例如，命令 `aws ec2 describe-regions` 位于 `ec2` 命令组下。*aliasname* 即您所称的别名。*command* 是您想要调用的命令，它可以包括其他别名。您可以在别名中包含选项或参数，也可以在调用别名时添加选项或参数。

以下示例使用 `aws ec2 describe-regions` 命令创建别名 `aws ec2 regions`。由于此别名调用了 `ec2` 命令命名空间下的现有 Amazon CLI 命令，因此您可以编写不带 `aws ec2` 前缀的命令。

```
[command ec2]
regions = describe-regions --query Regions[].RegionName
```

要使用命令命名空间之外的命令创建别名，请在完整命令前面加上感叹号前缀。以下示例使用 `aws iam list-instance-profiles` 命令创建别名 `aws ec2 instance-profiles`。

```
[command ec2]
instance-profiles = !aws iam list-instance-profiles
```

> **Note**
>
> 别名仅使用现有命令命名空间，您不能创建新的命名空间。例如，您无法使用 `[command johnsmith]` 部分创建别名，因为 `johnsmith` 命令命名空间尚不存在。

创建 bash 脚本别名

> **Warning**
>
> 要使用 Amazon CLI 别名 bash 脚本，必须使用兼容 bash 的终端

您可以使用以下语法为更高级的流程使用 bash 脚本创建别名。

语法

```
aliasname =
    !f() {
        script content
}; f
```

aliasname 即您所称的别名，*script content* 是您调用别名时要运行的脚本。

以下示例使用 opendns 输出您当前的 IP 地址。由于您可以在其他别名中使用别名，因此以下 myip 别名可用于允许或撤消从其他别名访问 IP 地址的权限。

```
myip =
  !f() {
    dig +short myip.opendns.com @resolver1.opendns.com
  }; f
```

以下脚本示例调用了之前的 aws myip 别名，以授权 Amazon EC2 安全组入口的 IP 地址。

```
authorize-my-ip =
  !f() {
    ip=$(aws myip)
    aws ec2 authorize-security-group-ingress --group-id ${1} --cidr $ip/32 --protocol
 tcp --port 22
  }; f
```

当您调用使用 bash 脚本的别名时，变量将始终按照您输入的顺序进行传递。在 bash 脚本中，不考虑变量名称，仅考虑它们出现的顺序。在以下 textalert 别名示例中，--message 选项的变量是第一个，--phone-number 选项是第二个。

```
textalert =
```

```
  !f() {
    aws sns publish --message "${1}" --phone-number ${2}
  }; f
```

步骤 3：调用别名

要运行在 alias 文件中创建的别名，请使用以下语法。您可以在调用别名时添加其他选项。

语法

```
$ aws aliasname
```

以下示例使用 aws whoami 命令别名。

```
$ aws
 whoami
{
    "UserId": "A12BCD34E5FGHI6JKLM",
    "Account": "1234567890987",
    "Arn": "arn:aws:iam::1234567890987:user/userName"
}
```

以下示例使用了带有其他选项的 aws whoami 别名，仅返回 Account 输出中的 text 数字。

```
$ aws whoami --query Account --output
 text
1234567890987
```

以下示例使用 aws ec2 regions 子命令别名。

```
$ aws ec2
 regions
[
    "ap-south-1",
    "eu-north-1",
    "eu-west-3",
    "eu-west-2",
...
```

使用 bash 脚本变量调用别名

调用使用 bash 脚本的别名时，变量将按照输入的顺序进行传递。在 bash 脚本中，不考虑变量的名称，仅考虑它们出现的顺序。例如，在以下 textalert 别名中，选项 --message 的变量是第一个，--phone-number 是第二个。

```
textalert =
  !f() {
    aws sns publish --message "${1}" --phone-number ${2}
  }; f
```

调用 textalert 别名时，您需要按照变量在别名中运行的顺序进行传递。在以下示例中，我们使用变量 $message 和 $phone。$message 变量将作为 ${1} 选项的 --message 传递，$phone 变量将作为 ${2} 选项的 --phone-number 传递。这会成功调用 textalert 别名来发送消息。

```
$ aws textalert $message
 $phone
{
    "MessageId": "1ab2cd3e4-fg56-7h89-i01j-2klmn34567"
}
```

在以下示例中，将别名调用至 $phone 和 $message 时，将切换顺序。$phone 变量将作为 ${1} 选项的 --message 传递，$message 变量将作为 ${2} 选项的 --phone-number 传递。由于变量顺序混乱，因此别名错误地传递了变量。这会导致发生错误，因为 $message 的内容与 --phone-number 选项的电话号码格式要求不匹配。

```
$ aws textalert $phone
 $message
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

Unknown options: text
```

别名存储库示例

GitHub 上的 Amazon CLI 别名存储库包含由 Amazon CLI 开发人员团队和社区创建的 Amazon CLI 别名示例。您可以使用整个 `alias` 文件示例，也可以自己使用单个别名。

> **Warning**
>
> 运行本节中的命令会删除现有 `alias` 文件。为避免覆盖现有别名文件，请更改下载位置。

使用存储库中的别名

1. 安装 Git。有关安装说明，请参阅 Git 文档中的入门 - 安装 Git。
2. 安装 `jp` 命令。`jp` 命令是在 `tostring` 别名中使用的。有关安装说明，请参阅 GitHub 上的 JMESPath (jp) README.md。
3. 安装 `jq` 命令。`jq` 命令是在 `tostring-with-jq` 别名中使用的。有关安装说明，请参阅 GitHub 上的 JSON 处理器 (jq) 。
4. 通过执行以下操作之一下载 `alias` 文件：

   - 运行以下命令，它是从存储库下载的并将 `alias` 文件复制到配置文件夹。

     Linux and macOS

     ```
     $ git clone https://github.com/awslabs/awscli-aliases.git
     $ mkdir -p ~/.aws/cli
     $ cp awscli-aliases/alias ~/.aws/cli/alias
     ```

     Windows

     ```
     C:\> git clone https://github.com/awslabs/awscli-aliases.git
     C:\> md %USERPROFILE%\.aws\cli
     C:\> copy awscli-aliases\alias %USERPROFILE%\.aws\cli
     ```

   - 直接从存储库下载并保存到 Amazon CLI 配置文件夹中的 `cli` 文件夹。默认情况下，配置文件夹为 `~/.aws/`（Linux 或 macOS）和 `%USERPROFILE%\.aws\` (Windows)。
5. 要验证别名是否有效，请运行以下别名。

   ```
   $ aws whoami
   ```

这将显示与 `aws sts get-caller-identity` 命令相同的响应：

```
{
    "Account": "012345678901",
    "UserId": "AIUAINBADX2VEG2TC6HD6",
    "Arn": "arn:aws:iam::012345678901:user/myuser"
}
```

资源

- GitHub 上的 Amazon CLI 别名存储库包含由 Amazon CLI 开发人员团队创建的 Amazon CLI 别名示例以及 Amazon CLI 社区的贡献。
- 来自 Amazon re:Invent 2016：有效的 Amazon CLI 用户的别名特征公告 (YouTube)。
- `aws sts get-caller-identity`
- `aws ec2 describe-instances`
- `aws sns publish`

排查 Amazon CLI 错误

本节介绍常见错误和解决您的问题的故障排除步骤。建议首先进行一般故障排除。

目录

- 首先尝试的一般故障排除
  - 检查您的 Amazon CLI 命令格式
  - 检查您的 Amazon CLI 命令正在使用的 Amazon Web Services 区域
  - 确认您运行的是 Amazon CLI 的最新版本
  - 使用 --debug 选项
  - 启用并审查 Amazon CLI 命令历史记录日志
  - 确认已配置 Amazon CLI
- 找不到命令错误
- “aws --version”命令返回的版本与您安装的版本不同
- 卸载 Amazon CLI 后，“aws --version”命令返回一个版本
- Amazon CLI 处理了参数名称不完整的命令

- 访问被拒绝错误
- 凭证无效和密钥错误
- 签名与错误不匹配
- 找不到 Windows 控制台错误
- SSL 证书错误
- JSON 无效错误
- 其他资源

首先尝试的一般故障排除

如果您使用 Amazon CLI 时收到错误或遇到问题，建议使用以下一般提示以帮助您进行故障排除。

回到顶部

检查您的 Amazon CLI 命令格式

如果您收到一个错误，表明某个命令不存在，或者它无法识别文档指明可用的参数 (`Parameter validation failed`)，则您的命令可能格式不正确。我们建议您检查以下内容：

- 检查命令是否存在拼写和格式错误。
- 确认您的命令中适用于您的终端的所有引号和转义都是正确的。
- 生成 Amazon CLI 骨架以确认命令结构。
- 对于 JSON，请参阅 JSON 值的其他问题排查。如果您在终端处理 JSON 格式时遇到问题，我们建议通过使用 Blob 将 JSON 数据直接传递给 Amazon CLI，以跳过终端的引号规则。

有关应该如何构造特定命令的更多信息，请参阅《Amazon CLI 参考指南》。

回到顶部

检查您的 Amazon CLI 命令正在使用的 Amazon Web Services 区域

> **Note**
>
> 使用 Amazon CLI 时，必须明确指定或通过设置默认区域来指定 Amazon Web Services 区域。有关您可以指定的所有 Amazon Web Services 区域的列表，请参阅《Amazon Web

Services 一般参考》中的Amazon 区域和端点。Amazon CLI 使用的 Amazon Web Services 区域指示符与您在 Amazon Web Services 管理控制台 URL 和服务端点中看到的名称相同。

如果 Amazon Web Services 服务对您指定的 Amazon Web Services 区域不可用，或者您的资源位于其他 Amazon Web Services 区域，则可能会出现错误或意外结果。Amazon Web Services 区域设置的优先顺序如下：

- `--region` 命令行选项。
- `AWS_DEFAULT_REGION` 环境变量。
- `region` 配置文件设置。

确认对您的资源使用了正确的 Amazon Web Services 区域。

回到顶部

确认您运行的是 Amazon CLI 的最新版本

如果您收到一个错误，表明某个命令不存在，或者它无法识别《Amazon CLI 参考指南》中指明可用的参数，请首先确认您的命令格式是否正确。如果格式正确，我们建议您升级到 Amazon CLI 的最新版本。几乎每个工作日都发布 Amazon CLI 的更新版本。这些新版本的 Amazon 中引入了新的 Amazon CLI 服务、功能和参数。获取这些新服务、功能或参数的唯一方式是升级到首次引入该元素后发布的版本。

如何更新您的 Amazon CLI 版本取决于您最初安装它的方式，如安装 Amazon CLI中所述。

如果您使用了某个捆绑安装程序，则可能需要先删除现有安装，然后为您的操作系统下载并安装最新版本。

回到顶部

使用 `--debug` 选项

如果 Amazon CLI 报告一个您不能立即理解的错误，或者产生您不期望的结果，您可以通过使用 `--debug` 选项再次运行此命令，以获得有关该错误的更多详细信息。使用此选项，Amazon CLI 会输出有关它处理命令所执行的每一步的详细信息。输出中的详细信息可以帮助您确定错误发生的时间，并提供错误从何处开始的线索。

您可以将输出发送到文本文件以供日后查看，或者按要求发送给 Amazon Web Services 支持。

当您包含 --debug 选项时，一些详细信息包括：

- 查找凭证
- 解析提供的参数
- 构建发送到 Amazon 服务器的请求
- 发送到的请求的内容Amazon
- 原始响应的内容
- 带格式的输出

以下是使用和不使用 --debug 选项运行命令的示例。

```
$ aws iam list-groups --profile MyTestProfile
{
    "Groups": [
        {
            "Path": "/",
            "GroupName": "MyTestGroup",
            "GroupId": "AGPA0123456789EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/MyTestGroup",
            "CreateDate": "2019-08-12T19:34:04Z"
        }
    ]
}
```

```
$ aws iam list-groups --profile MyTestProfile --debug
2019-08-12 12:36:18,305 - MainThread - awscli.clidriver - DEBUG - CLI version: aws-
cli/1.16.215 Python/3.7.3 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205
2019-08-12 12:36:18,305 - MainThread - awscli.clidriver - DEBUG - Arguments entered to
 CLI: ['iam', 'list-groups', '--debug']
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function add_scalar_parsers at 0x7fdf173161e0>
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function register_uri_param_handler at 0x7fdf17dec400>
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function inject_assume_role_provider_cache at
 0x7fdf17da9378>
2019-08-12 12:36:18,307 - MainThread - botocore.credentials - DEBUG - Skipping
 environment variable credential check because profile name was explicitly set.
2019-08-12 12:36:18,307 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function attach_history_handler at 0x7fdf173ed9d8>
```

```
2019-08-12 12:36:18,308 - MainThread - botocore.loaders - DEBUG - Loading JSON
 file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/
service-2.json
2019-08-12 12:36:18,317 - MainThread - botocore.hooks - DEBUG - Event building-command-
table.iam: calling handler <function add_waiters at 0x7fdf1731a840>
2019-08-12 12:36:18,320 - MainThread - botocore.loaders - DEBUG - Loading JSON
 file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/
waiters-2.json
2019-08-12 12:36:18,321 - MainThread - awscli.clidriver - DEBUG - OrderedDict([('path-
prefix', <awscli.arguments.CLIArgument object at 0x7fdf171ac780>), ('marker',
 <awscli.arguments.CLIArgument object at 0x7fdf171b09e8>), ('max-items',
 <awscli.arguments.CLIArgument object at 0x7fdf171b09b0>)])
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-
argument-table.iam.list-groups: calling handler <function add_streaming_output_arg at
 0x7fdf17316510>
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-
argument-table.iam.list-groups: calling handler <function add_cli_input_json at
 0x7fdf17da9d90>
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-
argument-table.iam.list-groups: calling handler <function unify_paging_params at
 0x7fdf17328048>
2019-08-12 12:36:18,326 - MainThread - botocore.loaders - DEBUG - Loading JSON
 file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/
paginators-1.json
2019-08-12 12:36:18,326 - MainThread - awscli.customizations.paginate - DEBUG -
 Modifying paging parameters for operation: ListGroups
2019-08-12 12:36:18,326 - MainThread - botocore.hooks - DEBUG - Event building-
argument-table.iam.list-groups: calling handler <function add_generate_skeleton at
 0x7fdf1737eae8>
2019-08-12 12:36:18,326 - MainThread - botocore.hooks - DEBUG - Event
 before-building-argument-table-parser.iam.list-groups: calling handler
 <bound method OverrideRequiredArgsArgument.override_required_args of
 <awscli.customizations.cliinputjson.CliInputJSONArgument object at 0x7fdf171b0a58>>
2019-08-12 12:36:18,327 - MainThread - botocore.hooks - DEBUG - Event
 before-building-argument-table-parser.iam.list-groups: calling handler
 <bound method GenerateCliSkeletonArgument.override_required_args of
 <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at
 0x7fdf171c5978>>
2019-08-12 12:36:18,327 - MainThread - botocore.hooks - DEBUG - Event operation-
args-parsed.iam.list-groups: calling handler functools.partial(<function
 check_should_enable_pagination at 0x7fdf17328158>, ['marker', 'max-items'], {'max-
items': <awscli.arguments.CLIArgument object at 0x7fdf171b09b0>}, OrderedDict([('path-
prefix', <awscli.arguments.CLIArgument object at 0x7fdf171ac780>), ('marker',
 <awscli.arguments.CLIArgument object at 0x7fdf171b09e8>), ('max-items',
```

```
 <awscli.customizations.paginate.PageArgument object at 0x7fdf171c58d0>), ('cli-
input-json', <awscli.customizations.cliinputjson.CliInputJSONArgument object at
 0x7fdf171b0a58>), ('starting-token', <awscli.customizations.paginate.PageArgument
 object at 0x7fdf171b0a20>), ('page-size', <awscli.customizations.paginate.PageArgument
 object at 0x7fdf171c5828>), ('generate-cli-skeleton',
 <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at
 0x7fdf171c5978>)]))
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-
arg.iam.list-groups.path-prefix: calling handler <awscli.paramfile.URIArgumentHandler
 object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-
arg.iam.list-groups.marker: calling handler <awscli.paramfile.URIArgumentHandler object
 at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-
arg.iam.list-groups.max-items: calling handler <awscli.paramfile.URIArgumentHandler
 object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG -
 Event load-cli-arg.iam.list-groups.cli-input-json: calling handler
 <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG -
 Event load-cli-arg.iam.list-groups.starting-token: calling handler
 <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-
arg.iam.list-groups.page-size: calling handler <awscli.paramfile.URIArgumentHandler
 object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event
 load-cli-arg.iam.list-groups.generate-cli-skeleton: calling handler
 <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,329 - MainThread - botocore.hooks - DEBUG
 - Event calling-command.iam.list-groups: calling handler
 <bound method CliInputJSONArgument.add_to_call_parameters of
 <awscli.customizations.cliinputjson.CliInputJSONArgument object at 0x7fdf171b0a58>>
2019-08-12 12:36:18,329 - MainThread - botocore.hooks - DEBUG -
 Event calling-command.iam.list-groups: calling handler <bound
 method GenerateCliSkeletonArgument.generate_json_skeleton of
 <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at
 0x7fdf171c5978>>
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for
 credentials via: assume-role
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for
 credentials via: assume-role-with-web-identity
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for
 credentials via: shared-credentials-file
```

```
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - INFO - Found credentials
 in shared credentials file: ~/.aws/credentials
2019-08-12 12:36:18,330 - MainThread - botocore.loaders - DEBUG - Loading JSON file: /
home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/endpoints.json
2019-08-12 12:36:18,334 - MainThread - botocore.hooks - DEBUG - Event choose-service-
name: calling handler <function handle_service_name_alias at 0x7fdf1898eb70>
2019-08-12 12:36:18,337 - MainThread - botocore.hooks - DEBUG - Event creating-client-
class.iam: calling handler <function add_generate_presigned_url at 0x7fdf18a028c8>
2019-08-12 12:36:18,337 - MainThread - botocore.regions - DEBUG - Using partition
 endpoint for iam, us-west-2: aws-global
2019-08-12 12:36:18,337 - MainThread - botocore.args - DEBUG - The s3 config key is not
 a dictionary type, ignoring its value of: None
2019-08-12 12:36:18,340 - MainThread - botocore.endpoint - DEBUG - Setting iam timeout
 as (60, 60)
2019-08-12 12:36:18,341 - MainThread - botocore.loaders - DEBUG - Loading JSON file: /
home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/_retry.json
2019-08-12 12:36:18,341 - MainThread - botocore.client - DEBUG - Registering retry
 handlers for service: iam
2019-08-12 12:36:18,342 - MainThread - botocore.hooks - DEBUG - Event before-
parameter-build.iam.ListGroups: calling handler <function generate_idempotent_uuid at
 0x7fdf189b10d0>
2019-08-12 12:36:18,342 - MainThread - botocore.hooks - DEBUG - Event before-
call.iam.ListGroups: calling handler <function inject_api_version_header_if_needed at
 0x7fdf189b2a60>
2019-08-12 12:36:18,343 - MainThread - botocore.endpoint - DEBUG - Making
 request for OperationModel(name=ListGroups) with params: {'url_path': '/',
 'query_string': '', 'method': 'POST', 'headers': {'Content-Type': 'application/x-
www-form-urlencoded; charset=utf-8', 'User-Agent': 'aws-cli/1.16.215 Python/3.7.3
 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205'}, 'body': {'Action':
 'ListGroups', 'Version': '2010-05-08'}, 'url': 'https://iam.amazonaws.com/',
 'context': {'client_region': 'aws-global', 'client_config': <botocore.config.Config
 object at 0x7fdf16e9a4a8>, 'has_streaming_input': False, 'auth_type': None}}
2019-08-12 12:36:18,343 - MainThread - botocore.hooks - DEBUG - Event request-
created.iam.ListGroups: calling handler <bound method RequestSigner.handler of
 <botocore.signers.RequestSigner object at 0x7fdf16e9a470>>
2019-08-12 12:36:18,343 - MainThread - botocore.hooks - DEBUG - Event choose-
signer.iam.ListGroups: calling handler <function set_operation_specific_signer at
 0x7fdf18996f28>
2019-08-12 12:36:18,343 - MainThread - botocore.auth - DEBUG - Calculating signature
 using v4 auth.
2019-08-12 12:36:18,343 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
POST
/
```

```
content-type:application/x-www-form-urlencoded; charset=utf-8
host:iam.amazonaws.com
x-amz-date:20190812T193618Z

content-type;host;x-amz-date
5f776d91EXAMPLE9b8cb5eb5d6d4a787a33ae41c8cd6eEXAMPLEca69080e1e1f
2019-08-12 12:36:18,344 - MainThread - botocore.auth - DEBUG - StringToSign:
AWS4-HMAC-SHA256
20190812T193618Z
20190812/us-east-1/iam/aws4_request
ab7e367eEXAMPLE2769f178ea509978cf8bfa054874b3EXAMPLE8d043fab6cc9
2019-08-12 12:36:18,344 - MainThread - botocore.auth - DEBUG - Signature:
d85a0EXAMPLEb40164f2f539cdc76d4f294fe822EXAMPLE18ad1ddf58a1a3ce7
2019-08-12 12:36:18,344 - MainThread - botocore.endpoint - DEBUG - Sending
 http request: <AWSPreparedRequest stream_output=False, method=POST,
 url=https://iam.amazonaws.com/, headers={'Content-Type': b'application/
x-www-form-urlencoded; charset=utf-8', 'User-Agent': b'aws-cli/1.16.215
 Python/3.7.3 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205',
 'X-Amz-Date': b'20190812T193618Z', 'Authorization': b'AWS4-HMAC-SHA256
 Credential=AKIA01234567890EXAMPLE-east-1/iam/aws4_request, SignedHeaders=content-
type;host;x-amz-date, Signature=d85a07692aceb401EXAMPLEa1b18ad1ddf58a1a3ce7EXAMPLE',
 'Content-Length': '36'}>
2019-08-12 12:36:18,344 - MainThread - urllib3.util.retry - DEBUG - Converted retries
 value: False -> Retry(total=False, connect=None, read=None, redirect=0, status=None)
2019-08-12 12:36:18,344 - MainThread - urllib3.connectionpool - DEBUG - Starting new
 HTTPS connection (1): iam.amazonaws.com:443
2019-08-12 12:36:18,664 - MainThread - urllib3.connectionpool - DEBUG - https://
iam.amazonaws.com:443 "POST / HTTP/1.1" 200 570
2019-08-12 12:36:18,664 - MainThread - botocore.parsers - DEBUG - Response headers:
 {'x-amzn-RequestId': '74c11606-bd38-11e9-9c82-559da0adb349', 'Content-Type': 'text/
xml', 'Content-Length': '570', 'Date': 'Mon, 12 Aug 2019 19:36:18 GMT'}
2019-08-12 12:36:18,664 - MainThread - botocore.parsers - DEBUG - Response body:
b'<ListGroupsResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">\n
 <ListGroupsResult>\n    <IsTruncated>false</IsTruncated>\n    <Groups>\n
  <member>\n        <Path>/</Path>\n        <GroupName>MyTestGroup</GroupName>
\n        <Arn>arn:aws:iam::123456789012:group/MyTestGroup</Arn>\n
 <GroupId>AGPA1234567890EXAMPLE</GroupId>\n        <CreateDate>2019-08-12T19:34:04Z</
CreateDate>\n      </member>\n    </Groups>\n  </ListGroupsResult>\n
 <ResponseMetadata>\n    <RequestId>74c11606-bd38-11e9-9c82-559da0adb349</RequestId>\n
 </ResponseMetadata>\n</ListGroupsResponse>\n'
2019-08-12 12:36:18,665 - MainThread - botocore.hooks - DEBUG - Event needs-
retry.iam.ListGroups: calling handler <botocore.retryhandler.RetryHandler object at
 0x7fdf16e9a780>
2019-08-12 12:36:18,665 - MainThread - botocore.retryhandler - DEBUG - No retry needed.
```

```
2019-08-12 12:36:18,665 - MainThread - botocore.hooks - DEBUG - Event after-
call.iam.ListGroups: calling handler <function json_decode_policies at 0x7fdf189b1d90>
{
    "Groups": [
        {
            "Path": "/",
            "GroupName": "MyTestGroup",
            "GroupId": "AGPA123456789012EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/MyTestGroup",
            "CreateDate": "2019-08-12T19:34:04Z"
        }
    ]
}
```

回到顶部

启用并审查 Amazon CLI 命令历史记录日志

您可以使用 Amazon CLI 文件设置启用 cli_history 命令历史记录日志。启用此设置后，Amazon CLI 记录 aws 命令的历史记录。

您可以使用 aws history list 命令列出您的历史记录，然后使用 aws history show 命令中生成的 command_id 获取详细信息。有关更多信息，请参阅《Amazon CLI 参考指南》中的 aws history。

当您包含 --debug 选项时，一些详细信息包括：

- 针对 botocore 的 API 调用
- 状态代码
- HTTP 响应
- 标头
- 返回代码

您可以使用此信息来确认参数数据和 API 调用是否按预期方式运行，然后可以推断出您的命令在哪个步骤失败。

回到顶部

确认已配置 Amazon CLI

如果您的 `config` 和 `credentials` 文件或您的 IAM 用户或角色未得以正确配置，可能发生各种错误。有关解决 `config` 和 `credentials` 文件或您的 IAM 用户或角色的错误的更多信息，请参阅the section called “访问被拒绝错误”和the section called “凭证无效和密钥错误”。

回到顶部

找不到命令错误

此错误意味着操作系统找不到 Amazon CLI 命令。安装可能不完整或需要更新。

可能的原因：您正在尝试使用的 Amazon CLI 功能比您安装的版本更高，或者格式不正确

错误示例文本：

```
$ aws s3 copy
aws: [ERROR]: argument operation: Found invalid choice 'copy'

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help
```

如果命令格式不正确，或者您使用的是发布此功能之前的早期版本，可能会发生各种错误。有关解决围绕这两个问题的错误的更多信息，请参阅the section called “检查您的 Amazon CLI 命令格式”和the section called “确认您运行的是 Amazon CLI 的最新版本”。

回到顶部

可能的原因：安装后需要重新启动终端

错误示例文本：

```
$ aws --version
command not found: aws
```

如果第一次安装或更新 Amazon CLI 后找不到 `aws` 命令，您可能需要重新启动终端以便它识别任何 PATH 更新。

回到顶部

可能的原因：Amazon CLI 未完全安装

错误示例文本：

```
$ aws --version
command not found: aws
```

如果第一次安装或更新 Amazon CLI 后找不到 aws 命令，它可能尚未完全安装。按照 安装 Amazon CLI 中适用于您的平台的步骤尝试重新安装。

回到顶部

可能的原因：Amazon CLI 没有权限 (Linux)

如果第一次在 Linux 安装或更新 aws 后找不到 Amazon CLI 命令，可能是没有安装文件夹的 execute 权限。使用 PATH 对您的 Amazon CLI 安装运行以下命令，以将 chmod 权限提供给 Amazon CLI：

```
$ sudo chmod -R 755 /usr/local/aws-cli/
```

回到顶部

可能的原因：安装期间未更新操作系统 **PATH**。

错误示例文本：

```
$ aws --version
command not found: aws
```

您可能需要将 aws 可执行文件添加到操作系统的 PATH 环境变量中。要将 Amazon CLI 添加到 PATH，请按照适用于您的操作系统的以下说明操作。

Linux and macOS

1. 在您的用户目录中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 echo $SHELL。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash：.bash_profile、.profile 或 .bash_login
   - Zsh – .zshrc

- Tcsh：.tcshrc、.cshrc 或 .login

2. 向配置文件脚本中添加导出命令。以下命令将您的本地 bin 添加到当前 PATH 变量。

```
export PATH=/usr/local/bin:$PATH
```

3. 将更新的配置文件重新加载到当前会话中。

```
$ source ~/.bash_profile
```

Windows

1. 在 Windows 命令提示符下，使用带 /R *path* 参数的 where 命令来查找 aws 文件位置。结果将返回所有包含 aws 的文件夹。

```
C:\> where /R c:\ aws
c:\Program Files\Amazon\AWSCLIV2\aws.exe
...
```

默认情况下，Amazon CLI 版本 2 位于：

```
c:\Program Files\Amazon\AWSCLIV2\aws.exe
```

2. 按 Windows 键并输入 **environment variables**。
3. 从建议列表中，选择 Edit environment variables for your account（编辑您账户的环境变量）。
4. 选择 PATH，然后选择 Edit（编辑）。
5. 将在第一步中找到的路径添加到 Variable value（变量值）字段中，例如 ***C:\Program Files\Amazon\AWSCLIV2\aws.exe***。
6. 选择 OK（确定）两次以应用新设置。
7. 关闭任何运行的命令提示符并重新打开命令提示符窗口。

回到顶部

“aws --version”命令返回的版本与您安装的版本不同

您的终端可能为 Amazon CLI 返回的 PATH 与您预期的不同。

可能的原因：安装后需要重新启动终端

如果 aws 命令显示错误的版本，您可能需要重新启动终端以便它识别任何 PATH 更新。需要关闭所有打开的终端，而不仅仅是活动的终端。

回到顶部

可能的原因：安装后需要重新启动系统

如果 aws 命令显示了错误的版本，并且重新启动终端不起作用，您可能需要重新启动系统，系统才能识别您的 PATH 更新。

回到顶部

可能的原因：您有多个 Amazon CLI 版本

如果您更新了 Amazon CLI 并且使用了与预先存在的安装不同的安装方法，可能会导致安装多个版本。例如，如果您在 Linux 或 macOS 上使用 pip 进行了当前安装，但试图使用 .pkg 安装文件进行更新，这可能会导致一些冲突，尤其是当 PATH 指向旧版本时。

要解决这个问题，请卸载 Amazon CLI 的所有版本，然后执行净安装。

卸载所有版本后，请按照适用于您的操作系统的说明安装所需版本：Amazon CLI 版本 1 或 Amazon CLI 版本 2。

> **Note**
>
> 如果您在预安装了 Amazon CLI 版本 1 的情况下安装了 Amazon CLI 版本 2 之后发生这种情况，请遵循从 Amazon CLI 版本 1 迁移时的安装说明中的迁移说明进行操作。

回到顶部

卸载 Amazon CLI 后，“**aws --version**”命令返回一个版本

当系统中的某个位置仍安装了 Amazon CLI 时，通常会发生这种情况。

可能的原因：卸载后需要重新启动终端

如果 aws --version 命令仍然有效，您可能需要重新启动终端以便它识别任何终端更新。

回到顶部

可能的原因：系统上有多个 Amazon CLI 版本，或者您没有使用与最初安装 Amazon CLI 所用的相同卸载方法

如果使用与安装 Amazon CLI 所用的不同方法卸载它，或者您安装了多个版本，则可能无法正确卸载 Amazon CLI。例如，如果您使用 pip 执行了当前安装，则必须使用 pip 卸载它。要解决此问题，请使用与安装 Amazon CLI 所用的相同方法执行卸载。

1. 按照适用于您的操作系统的说明和原始安装方法卸载 Amazon CLI 版本 1 和 Amazon CLI 版本 2。
2. 关闭所有打开的终端。
3. 打开首选终端，输入以下命令并确认没有返回任何版本。

```
$ aws --version
command not found: aws
```

如果输出中仍列出一个版本，很可能 Amazon CLI 是使用不同的方法安装的，或者有多个版本。如果您不知道安装 Amazon CLI 时采用的是哪种方法，请按照适用于您的操作系统的 Amazon CLI 版本 1 和 Amazon CLI 版本 2 的每种卸载方法的说明进行操作，直至不会收到任何版本输出。

> **Note**
>
> 如果您是使用软件包管理器（pip、apt、brew 等）安装 Amazon CLI 的，则必须使用同一个软件包管理器来卸载它。请务必按照软件包管理器提供的有关如何卸载软件包的所有版本的说明进行操作。

回到顶部

Amazon CLI 处理了参数名称不完整的命令

可能的原因：您使用了普遍接受的 Amazon CLI 参数缩写

由于 Amazon CLI 是使用 Python 构建的，因此 Amazon CLI 使用 Python `argparse` 库，包括 `allow_abbrev` 参数。Amazon CLI 可识别并处理参数的缩写。

以下 `create-change-set` 命令示例更改 CloudFormation 堆栈名称。参数 `--change-set-n` 被识别为 `--change-set-name` 的缩写，然后 Amazon CLI 处理命令。

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set-n my-
change-set
```

当您的缩写可能代表多个命令时，参数将不会被识别为缩写。

以下 create-change-set 命令示例更改 CloudFormation 堆栈名称。参数 --change-set- 无法识别为缩写，因为它可以是多个参数（例如 --change-set-name 和 --change-set-type）的缩写。因此，Amazon CLI 不会处理该命令。

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set- my-
change-set
```

> **Warning**
>
> 不要刻意使用参数缩写。缩写不可靠且不向后兼容。如果在命令中添加了任何会使缩写产生混淆的新参数，则会中断您的命令。
> 此外，如果参数是单值参数，则会导致命令出现意外行为。如果传递了单值参数的多个实例，则只有最后一个实例会运行。在以下示例中，参数 --filters 是单值参数。指定了参数 --filters 和 --filter。--filter 参数是 --filters 的缩写。这会导致应用 --filters 的两个实例，只有最后一个 --filter 参数会应用。
>
> ```
> $ aws ec2 describe-vpc-peering-connections \
>     --filters Name=tag:TagName,Values=VpcPeeringConnection \
>     --filter Name=status-code,Values=active
> ```
>
> 在运行命令之前，请确认您使用的是有效参数，以防止出现意外行为。

回到顶部

访问被拒绝错误

可能的原因：Amazon CLI 程序文件没有“运行”权限

在 Linux 或 macOS 上，确保 aws 程序具有发出调用的用户的运行权限。通常，权限设置为 755。

要添加用户的运行权限，请运行以下命令，并将 *~/.local/bin/aws* 替换为您计算机上指向此程序的路径。

```
$ chmod +x ~/.local/bin/aws
```

回到顶部

可能的原因：您的 IAM 身份没有执行此操作的权限

错误示例文本：

```
$ aws s3 ls
An error occurred (AcessDenied) when calling the ListBuckets operation: Access
 denied.
```

当您运行 Amazon CLI 命令时，使用将您与 IAM 账户或角色关联的凭证代表您执行 Amazon 操作。附加的策略必须向您授予权限，才能调用与通过 Amazon CLI 运行的命令相对应的 API 操作。

大多数命令会通过一个与命令名称匹配的名称来调用单个操作。但是，像 `aws s3 sync` 这样的自定义命令会调用多个 API。您可以查看命令通过使用 `--debug` 选项调用哪些 API。

如果您确定用户或角色具有策略所分配的适当权限，确保您的 Amazon CLI 命令使用的是您预期的凭证。请参阅下一节中有关凭证的内容以验证 Amazon CLI 使用的是您预期的凭证。

有关分配 IAM 权限的信息，请参阅《IAM 用户指南》中的访问管理概述：权限和策略。

回到顶部

凭证无效和密钥错误

错误示例文本：

```
$ aws s3 ls
An error occurred (InvalidAccessKeyId) when calling the ListBuckets operation: The AWS
 Access Key Id
you provided does not exist in our records.
```

```
$ aws s3 ls
An error occurred (InvalidClientTokenId) when calling the ListBuckets operation: The
 security token
included in the request is invalid.
```

可能的原因：Amazon CLI 从意外位置读取了不正确的凭证

Amazon CLI 读取凭证的位置可能与您的预期不同，或者密钥对信息不正确。您可以运行 `aws configure list` 以确认使用哪些凭证。

以下示例说明如何检查用于默认配置文件的凭证。

```
$ aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************ABCD  shared-credentials-file
secret_key     ****************ABCD  shared-credentials-file
    region                us-west-2             env    AWS_DEFAULT_REGION
```

以下示例说明如何检查命名配置文件的凭证。

```
$ aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************ABCD  shared-credentials-file
secret_key     ****************ABCD  shared-credentials-file
    region                us-west-2             env    AWS_DEFAULT_REGION
```

要确认密钥对详细信息，请检查 `config` 和 `credentials` 文件。有关 `config` 和 `credentials` 文件的更多信息，请参阅the section called “配置设置”。有关凭证和身份验证（包括凭证优先顺序）的更多信息，请参阅身份验证和访问凭证。

回到顶部

可能的原因：您计算机的时钟不同步

如果您使用的凭证是有效的，则可能是您的时钟不同步。在 Linux 或 macOS 上，运行 `date` 以检查时间。

```
$ date
```

如果您的系统时钟在几分钟内不正确，则使用 `ntpd` 进行同步。

```
$ sudo service ntpd stop
```

```
$ sudo ntpdate time.nist.gov
$ sudo service ntpd start
$ ntpstat
```

在 Windows 上，使用控制面板中的日期和时间选项来配置系统时钟。

回到顶部

签名与错误不匹配

错误示例文本：

```
$ aws s3 ls
An error occurred (SignatureDoesNotMatch) when calling the ListBuckets operation: The
 request signature we
calculated does not match the signature you provided. Check your key and signing
 method.
```

当 Amazon CLI 运行命令时，它会向 Amazon 服务器发送加密请求以执行适当的 Amazon 服务操作。您的凭证（访问密钥和私有密钥）参与了加密过程，使 Amazon 能够对发出请求的人员进行身份验证。有多种因素可能会干扰此过程的正常执行，如下所示。

可能的原因：您的时钟与 Amazon 服务器不同步

为了帮助防范重播攻击，在加密/解密过程中可能会使用当前时间。如果客户端和服务器的时间不一致超出允许的时间量，该过程可能会失败，并且请求会被拒绝。当您在其时钟与主机时钟不同步的虚拟机中运行命令时，也可能发生此错误。一个可能的原因是，当虚拟机休眠时，唤醒后需要一些时间才能将时钟与主机同步。

在 Linux 或 macOS 上，运行 `date` 以检查时间。

```
$ date
```

如果您的系统时钟在几分钟内不正确，则使用 `ntpd` 进行同步。

```
$ sudo service ntpd stop
$ sudo ntpdate time.nist.gov
$ sudo service ntpd start
$ ntpstat
```

在 Windows 上，使用控制面板中的日期和时间选项来配置系统时钟。

回到顶部

可能的原因：操作系统错误地处理了包含某些特殊字符的 Amazon 密钥

如果 Amazon 密钥包含某些特殊字符（例如 -、+、/ 或 %），则某些操作系统变体会不正确地处理该字符串，并导致对该密钥字符串的解释不正确。

如果使用其他工具或脚本（例如，在创建新实例期间在其上构建凭证文件的工具）处理密钥，则这些工具和脚本可能有自己对特殊字符的处理，这会导致将这些字符转换为 Amazon 不再识别的内容。

我们建议重新生成私有密钥，以使获得的私有密钥不包含会导致问题的特殊字符。

回到顶部

找不到 Windows 控制台错误

错误示例文本：

```
$ aws s3 ls
No Windows console found. Are you running cmd.exe?
```

当您使用 Amazon CLI 命令时，您收到“找不到 Windows 控制台。您是否正在运行 cmd.exe？” 错误消息。如果您安装的 Python `prompt_toolkit` 已经过时，那么对于 Amazon CLI 版本 1，这通常是一个错误。要解决此问题，请安装 Python 网站上最新版本的 `prompt_toolkit`。

回到顶部

SSL 证书错误

可能的原因：Amazon CLI 不信任您的代理的证书

错误示例文本：

```
$ aws s3 ls
[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed
```

当您使用 Amazon CLI 命令时，您收到 `[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed` 错误消息。这是因为 Amazon CLI 不信任代理的证书，原因包括

代理的证书是自签名的，而您的公司却设置为证书颁发机构 (CA)。这会致使 Amazon CLI 无法在本地 CA 注册机构中找到您的公司 CA 根证书。

要解决这个问题，请指示 Amazon CLI 使用 ca_bundle 配置文件设置、--ca-bundle 命令行选项或 AWS_CA_BUNDLE 环境变量在何处查找公司 .pem 文件。

回到顶部

可能的原因：您的配置未指向正确的 CA 根证书位置

错误示例文本：

```
$ aws s3 ls
SSL validation failed for regionname [Errno 2] No such file or directory
```

这是由于您的证书颁发机构 (CA)捆绑包文件位置在 Amazon CLI 中配置不正确所致。要解决此问题，请确认您的公司 .pem 文件所在的位置，并使用 ca_bundle 配置文件设置、--ca-bundle 命令行选项或 AWS_CA_BUNDLE 环境变量更新 Amazon CLI 配置。

回到顶部

可能的原因：您的配置使用了错误的 Amazon Web Services 区域

错误文本示例：

```
$ aws s3 ls
[SSL: CERTIFICATE_ VERIFY_FAILED] certificate verify failed
```

如果 Amazon Web Services 服务对您指定的 Amazon Web Services 区域不可用，或者您的资源位于其他 Amazon Web Services 区域，则可能会出现错误或意外结果。有关问题排查步骤，请参阅the section called “检查您的 Amazon CLI 命令正在使用的 Amazon Web Services 区域”。

回到顶部

可能的原因：您的 TLS 版本需要更新

错误文本示例：

```
$ aws s3 ls
[SSL: UNSAFE_LEGACY_RENEGOTIATION_DISABLED] unsafe legacy renegotiation disabled
```

Amazon Web Services 服务使用的 TLS 版本与您设备的 TLS 版本不兼容。要解决此问题，请更新到受支持的 TLS 版本。有关更多信息，请参阅 the section called “强制实施最低 TLS 版本”。

回到顶部

JSON 无效错误

错误文本示例：

```
$ aws dynamodb update-table \
    --provisioned-throughput '{"ReadCapacityUnits":15,WriteCapacityUnits":10}' \
    --table-name MyDDBTable
Error parsing parameter '--provisioned-throughput': Invalid JSON: Expecting property
 name enclosed in
double quotes: line 1 column 25 (char 24)
JSON received: {"ReadCapacityUnits":15,WriteCapacityUnits":10}
```

当您使用 Amazon CLI 命令时，您收到“Invalid JSON”错误消息。这通常是在您输入具有预期 JSON 格式的命令而 Amazon CLI 无法正确读取您的 JSON 时出现的错误。

可能的原因：您没有输入有效的 JSON 供 Amazon CLI 使用

确认您为命令输入了有效的 JSON。我们建议您对格式设置遇到问题的 JSON 使用 JSON 验证器。

要在命令行中使用更高级的 JSON，请考虑使用 jq 等命令行 JSON 处理器来创建 JSON 字符串。有关 jq 的更多信息，请参阅 GitHub 上的 jq 存储库。

回到顶部

可能的原因：您的终端的引号规则阻止将有效的 JSON 发送到 Amazon CLI

在 Amazon CLI 从命令接收任何内容之前，终端使用它自己的引号和转义规则来处理该命令。由于终端的格式设置规则，您的某些 JSON 内容可能会在命令传递给 Amazon CLI 之前剥离掉。在构建命令时，请务必使用终端的引号规则。

要进行故障排除，请使用 echo 命令来查看 Shell 如何处理您的参数：

```
$ echo {"ReadCapacityUnits":15,"WriteCapacityUnits":10}
ReadCapacityUnits:15 WriteCapacityUnits:10
```

```
$ echo '{"ReadCapacityUnits":15,"WriteCapacityUnits":10}'
{"ReadCapacityUnits":15,"WriteCapacityUnits":10}
```

修改您的命令，直到返回有效的 JSON。

要进行更深入的故障排除，请使用 --debug 参数来查看调试日志，因为它们将确切地显示传递给 Amazon CLI 的内容：

```
$ aws dynamodb update-table \
    --provisioned-throughput '{"ReadCapacityUnits":15,WriteCapacityUnits":10}' \
    --table-name MyDDBTable \
    --debug
2022-07-19 22:25:07,741 - MainThread - awscli.clidriver - DEBUG - CLI version: aws-
cli/1.18.147
Python/2.7.18 Linux/5.4.196-119.356.amzn2int.x86_64 botocore/1.18.6
2022-07-19 22:25:07,741 - MainThread - awscli.clidriver - DEBUG - Arguments entered
 to CLI:
['dynamodb', 'update-table', '--provisioned-throughput',
 '{"ReadCapacityUnits":15,WriteCapacityUnits":10}',
 '--table-name', 'MyDDBTable', '--debug']
```

使用终端的引号规则来修复 JSON 输入在发送到 Amazon CLI 时出现的任何问题。有关引号规则的更多信息，请参阅the section called “含字符串的引号”。

> **Note**
>
> 如果您在将有效的 JSON 发送到 Amazon CLI 时遇到问题，我们建议您使用 BLOB 将 JSON 数据直接发送到 Amazon CLI，从而绕过终端对 JSON 数据输入的引号规则。有关 Blob 的更多信息，请参阅Blob。

回到顶部

其他资源

有关 Amazon CLI 问题的其他帮助，请访问 GitHub 上的 Amazon CLI 社区，或者访问 Amazon Web Services re:Post 社区。

回到顶部

Amazon CLI 的示例

本章提供了一系列示例，向您展示如何将 Amazon Command Line Interface (Amazon CLI) 与 Amazon Web Services 服务结合使用。

在本指南中，Amazon CLI 具有以下类型的示例：

- 引导式命令示例 -《Amazon CLI 用户指南》中的引导式命令示例，演示如何将 Amazon CLI 与某些 Amazon Web Services 服务结合使用。这些示例通常比 Amazon CLI 参考指南中的示例更为详细。
- Amazon CLI 命令示例 - 开源命令示例，也可在 Amazon CLI 参考指南中找到。命令示例托管在 GitHub 上的 Amazon CLI 存储库中。
- Amazon CLI 使用 Bash 脚本代码示例 - 开源 bash 脚本示例。Bash 脚本示例托管在 GitHub 上的 Amazon 代码示例存储库中。

> **反馈示例**
>
> 找不到所需的内容？ 使用本页底部的提供反馈链接或 Amazon CLI 参考指南中的相关命令页面请求命令示例。
> 想要提交示例？ 在 GitHub 上的 Amazon 代码示例存储库中提交 Amazon CLI 命令示例。有关提交示例的更多信息，请参阅 GitHub 页面上的 Amazon CLI 代码示例提交快速步骤。

Amazon CLI 的引导式命令示例

Amazon Command Line Interface（Amazon CLI）是一种开源工具，使您能够使用命令行 Shell 中的命令与众多 Amazon Web Services 服务进行交互。本节提供了指导示例，说明如何利用 Amazon CLI 来访问其中一些 Amazon Web Services 服务。这包括一些自定义 Amazon CLI 命令，例如高级 `aws s3` 命令。这些命令示例演示了某些 Amazon Web Services 服务命令使用的常见操作，并提供了更多有用资源。

无论您是经验丰富的 Amazon 用户还是初次接触 Amazon CLI，这些指导示例都可以作为简化 Amazon 操作的资源。

有关可用于每种 Amazon Web Services 服务的命令的完整参考，请参阅 Amazon CLI 参考指南。此外，您还可以利用内置命令行帮助来浏览 Amazon CLI 中的一系列 Amazon Web Services 服务、命令、选项和功能。

有关本节中未提供的更多命令示例，请参阅 Amazon CLI 命令示例一节。这些是开源命令示例，也可在 Amazon CLI 参考指南中找到。命令示例托管在 GitHub 上的 Amazon CLI 存储库中。

有关开源 bash 脚本示例，请参阅the section called “Bash 脚本示例”。Bash 脚本示例托管在 GitHub 上的 Amazon 代码示例存储库中。

服务

- 在 Amazon CLI 中使用 Amazon DynamoDB
- 在 Amazon CLI 中使用 Amazon EC2
- 在 Amazon CLI 中使用 Amazon Glacier
- 在 Amazon CLI 中使用 IAM
- 在 Amazon CLI 中使用 Amazon S3
- 在 Amazon CLI 中访问 Amazon SNS

在 Amazon CLI 中使用 Amazon DynamoDB

Amazon Command Line Interface (Amazon CLI) 为所有Amazon数据库服务（包括 Amazon DynamoDB）提供支持。您可以使用 Amazon CLI 进行临时操作，如创建表。您还可以使用它在实用工具脚本中嵌入 DynamoDB 操作。

有关将 Amazon CLI 与 DynamoDB 结合使用的更多信息，请参阅《Amazon CLI 命令参考》中的 dynamodb。

要列出适用于 DynamoDB 的 Amazon CLI 命令，请使用以下命令。

```
$ aws dynamodb help
```

主题

- 先决条件
- 创建和使用 DynamoDB 表
- 使用 DynamoDB Local
- 资源

先决条件

要运行 dynamodb 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI和Amazon CLI 身份验证和访问凭证。

创建和使用 DynamoDB 表

命令行格式为 DynamoDB 命令名称后接该命令的参数。Amazon CLI 对于参数值支持 CLI 速记语法和完全 JSON。

下面的示例创建了一个名为 `MusicCollection` 的表。

```
$ aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-definitions AttributeName=Artist,AttributeType=S
 AttributeName=SongTitle,AttributeType=S \
    --key-schema AttributeName=Artist,KeyType=HASH
 AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

您可以使用类似下面的命令向表中添加新行，如以下示例所示。这些示例使用速记语法和 JSON 的组合。

```
$ aws dynamodb put-item \
    --table-name MusicCollection \
    --item '{
        "Artist": {"S": "No One You Know"},
        "SongTitle": {"S": "Call Me Today"} ,
        "AlbumTitle": {"S": "Somewhat Famous"}
      }' \
    --return-consumed-capacity TOTAL
{
    "ConsumedCapacity": {
        "CapacityUnits": 1.0,
        "TableName": "MusicCollection"
    }
}
```

```
$ aws dynamodb put-item \
    --table-name MusicCollection \
    --item '{
        "Artist": {"S": "Acme Band"},
        "SongTitle": {"S": "Happy Day"} ,
```

```
        "AlbumTitle": {"S": "Songs About Life"}
      }' \
    --return-consumed-capacity TOTAL
{
    "ConsumedCapacity": {
        "CapacityUnits": 1.0,
        "TableName": "MusicCollection"
    }
}
```

可能难以在单行命令中编写有效的 JSON。为了使之更简单，Amazon CLI 可以读取 JSON 文件。例如，请考虑以下 JSON 代码段，它存储在一个名为 `expression-attributes.json` 的文件中。

```
{
  ":v1": {"S": "No One You Know"},
  ":v2": {"S": "Call Me Today"}
}
```

您可以使用此文件通过 `query` 发出 Amazon CLI 请求。在下面的示例中，`expression-attributes.json` 文件的内容用作 `--expression-attribute-values` 参数的值。

```
$ aws dynamodb query --table-name MusicCollection \
    --key-condition-expression "Artist = :v1 AND SongTitle = :v2" \
    --expression-attribute-values file://expression-attributes.json
{
    "Count": 1,
    "Items": [
        {
            "AlbumTitle": {
                "S": "Somewhat Famous"
            },
            "SongTitle": {
                "S": "Call Me Today"
            },
            "Artist": {
                "S": "No One You Know"
            }
        }
    ],
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
```

使用 DynamoDB Local

除了 DynamoDB 之外，您还可以将 Amazon CLI 与 DynamoDB Local 结合使用。DynamoDB Local 是模拟 DynamoDB 服务的小客户端数据库和服务器。通过 DynamoDB Local 可编写使用 DynamoDB API 的应用程序，而无需操作 DynamoDB Web 服务中的任何表或数据。而所有 API 操作均重新路由到本地数据库。这样可节省预配置吞吐量、数据存储和数据传输费用。

有关 DynamoDB Local 以及如何将其与 Amazon CLI 结合使用的更多信息，请参阅 Amazon DynamoDB 开发人员指南中的以下部分：

- DynamoDB Local
- 将 Amazon CLI 与 DynamoDB Local 结合使用

资源

《Amazon CLI 参考：》

- `aws dynamodb`
- `aws dynamodb create-table`
- `aws dynamodb put-item`
- `aws dynamodb query`

《服务参考：》

- Amazon DynamoDB 开发人员指南中的 DynamoDB Local
- Amazon DynamoDB 开发人员指南中的将 Amazon CLI 与 DynamoDB Local 结合使用

在 Amazon CLI 中使用 Amazon EC2

Amazon Elastic Compute Cloud 简介
Amazon EC2 简介 – 通过 Amazon 实现弹性云服务器和托管

Amazon Elastic Compute Cloud（Amazon EC2）提供高度可扩展的灵活虚拟计算环境。Amazon EC2 允许您预置和管理虚拟服务器（称为 Amazon EC2 实例），以满足各种计算需求。

Amazon EC2 实例是虚拟机，可以使用不同的 CPU、内存、存储和联网能力配置对其进行自定义。根据您的应用程序要求，您可以从多种实例类型中进行选择，从轻量级、经济高效的选项到功能强大的高性能实例，不一而足。这种灵活性使您能够满足自己的计算需求，从而优化性能和成本效益。

此外，Amazon EC2 还提供了一套功能，使您能够有效地管理计算资源。其中包括快速启动新实例、创建用于快速部署的自定义机器映像（AMI），以及根据需要向上或向下扩展计算容量的功能。

您可以使用 Amazon Command Line Interface（Amazon CLI）访问 Amazon EC2 功能。要列出适用于 Amazon EC2 的 Amazon CLI 命令，请使用以下命令。

```
aws ec2 help
```

在运行任何命令之前，请设置默认证书。有关更多信息，请参阅 配置 Amazon CLI 设置。

本主题展示了执行 Amazon EC2 常见任务的 Amazon CLI 命令的短格式示例。

有关 Amazon CLI 命令的长格式示例，请参阅 GitHub 上的 Amazon CLI 代码示例存储库。

主题

- 在 Amazon CLI 中创建、显示和删除 Amazon EC2 密钥对
- 在 Amazon CLI 中创建、配置和删除 Amazon EC2 安全组
- 在 Amazon CLI 中启动、列出和删除 Amazon EC2 实例
- 在 Amazon CLI 中使用 bash 脚本更改 Amazon EC2 实例类型

在 Amazon CLI 中创建、显示和删除 Amazon EC2 密钥对

您可以使用 Amazon Command Line Interface (Amazon CLI) 创建、显示和删除 Amazon Elastic Compute Cloud (Amazon EC2) 密钥对。您可以使用密钥对连接 Amazon EC2 实例。在创建实例时您可以向 Amazon EC2 提供密钥对，然后在连接到实例时使用该密钥对进行身份验证。

> **Note**
>
> 有关其他命令示例，请参阅 Amazon CLI 参考指南。

主题

- 先决条件
- 创建密钥对

- 显示您的密钥对
- 删除您的密钥对
- 参考信息

先决条件

要运行 ec2 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI和Amazon CLI 身份验证和访问凭证。
- 设置您的 IAM 权限以允许访问 Amazon EC2。有关 Amazon EC2 的 IAM 权限的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 策略。

创建密钥对

要创建密钥对，请使用 aws ec2 create-key-pair 命令以及 --query 选项和 --output text 选项，以通过管道将私有密钥直接传输到文件。

```
$ aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text
 > MyKeyPair.pem
```

对于 Windows PowerShell，> file 重定向会默认采用 UTF-8 编码，这种编码不能用于某些 SSH 客户端。因此，您必须通过管道将输出传输到 out-file 命令和显式将编码设置为 ascii 来转换输出。

```
PS C:\>aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text
 | out-file -encoding ascii -filepath MyKeyPair.pem
```

生成的 MyKeyPair.pem 文件类似于以下内容。

```
-----BEGIN RSA PRIVATE KEY-----
EXAMPLEKEYKCAQEAy7WZhaDsrA1W3mRlQtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
vBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7iO5dSrvC7dQkW2duV5QuUdE0QW
Z/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449ilx9X1F
G50TCFeOzfl8dqqCP6GzbPaIjiU19xX/azOR9V+tpUOzEL+wmXnZt3/nHPQ5xvD2OJH67km6SuPW
oPzev/D8V+x4+bHthfSjR9Y7DvQFjfBVwHXigBdtZcU2/wei8D/HYwIDAQABAoIBAGZ1kaEvnrqu
/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtcHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/ufGxbL1
mb5qwMGUnEpJaZD6QSSs3kICLwWUYUiGfc0uiSbmJoap/GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2
bahyWyJNfjLe4M86yd2YK3V2CmK+X/BOsShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/tJWSD9
81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA3OzdXzMQexXVJ1TLZVEH0E7bhlY9d8O1ozR
```

```
oQs/FiZNAx2iijCWyv0lpjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfql+lIp1
YkriL0DbLXlvRAH+yHPRit2hHOjtUNZh4Axv+cpg09qbUI3+43eEy24B7G/Uh+GTfbjsXsOxQx/x
p9otyVwc7hsQ5TA5PZb+mvkJ5OBEKzet9XcKwONBYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
ayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWVOEihvm+xTtmaZlSp//lkq75XDwnU
WA8gkn6O3QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
gYBjbO+OZk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
oMo7yykABY7Ozd5wQewBQ4AdSlWSX4nGDtsiFxWiI5sKuAAeOCbTosy1s8w8fxoJ5Tz1sdoxNeGs
Arq6Wv/G16zQuAE9zK9vvwKBgF+09VI/1wJBirsDGz9whVWfFPrTkJNvJZzYt69qezxlsjgFKshy
WBhd4xHZtmCqpBPlAymEjr/TOlbxyARmXMnIOWIAnNXMGB4KGSyl1mzSVAoQ+fqR+cJ3d0dyPl1j
jjb0Ed/NY8frlNDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0iOegLDa
NWUH38v/nDCgEpIXD5Hn3qAEcju1IjmbwlvtW+nY2jVhv7UGd8MjwUTNGItdb6nsYqM2asrnF3qS
VRkAKKKYeGjkpUfVTrW0YFjXkfcrR/V+QFL5OndHAKJXjW7a4ejJLncTzmZSpYzwApc=
-----END RSA PRIVATE KEY-----
```

您的私有密钥不存储在 Amazon 中，并且只有 在创建后才能检索。以后您无法恢复它。而如果您丢失了私有密钥，则必须创建新的密钥对。

如果您要从 Linux 电脑连接到您的实例，建议您使用以下命令设置您的私有密钥文件的权限，以确保只有您可以读取该文件。

```
$ chmod 400 MyKeyPair.pem
```

显示您的密钥对

指纹是从密钥对生成的，您可以使用指纹验证您本地电脑上的私有密钥是否与 Amazon 中存储的公有密钥匹配。

指纹是取自私有密钥的 DER 编码副本的 SHA1 哈希。在创建密钥对时将捕获此值，此值与公有密钥一起存储在 Amazon 中。您可以在 Amazon EC2 控制台中或通过运行 Amazon CLI 命令 aws ec2 describe-key-pairs 查看指纹。

以下示例显示 MyKeyPair 的指纹。

```
$ aws ec2 describe-key-pairs --key-name MyKeyPair
{
    "KeyPairs": [
        {
            "KeyName": "MyKeyPair",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f"
        }
    ]
```

```
}
```

有关密钥和指纹的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 密钥对。

删除您的密钥对

要删除密钥对，请运行 `aws ec2 delete-key-pair` 命令，将 *MyKeyPair* 替换为要删除的密钥对的名称。

```
$ aws ec2 delete-key-pair --key-name MyKeyPair
```

参考信息

Amazon CLI 参考：

- `aws ec2`
- `aws ec2 create-key-pair`
- `aws ec2 delete-key-pair`
- `aws ec2 describe-key-pairs`

其他参考资料：

- Amazon Elastic Compute Cloud 文档
- 要查看和贡献 Amazon 开发工具包和 Amazon CLI 代码示例，请参阅 GitHub 上的 Amazon 代码示例存储库。

在 Amazon CLI 中创建、配置和删除 Amazon EC2 安全组

您可以为您的 Amazon Elastic Compute Cloud（Amazon EC2）实例创建安全组，安全组本质上用作一个防火墙，具有可确定哪些网络流量可以进入和离开的规则。

使用 Amazon Command Line Interface (Amazon CLI) 创建一个安全组，向现有安全组添加规则，以及删除安全组。

> **Note**
>
> 有关其他命令示例，请参阅 Amazon CLI 参考指南。

主题

- 先决条件
- 创建安全组
- 为您的安全组添加规则
- 删除您的安全组
- 参考信息

先决条件

要运行 ec2 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI和Amazon CLI 身份验证和访问凭证。
- 设置您的 IAM 权限以允许访问 Amazon EC2。有关 Amazon EC2 的 IAM 权限的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 策略。

创建安全组

您可以创建与虚拟私有云（VPC）关联的安全组。

以下 aws ec2 create-security-group 示例说明如何为指定的 VPC 创建安全组。

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group" --
vpc-id vpc-1a2b3c4d
{
    "GroupId": "sg-903004f8"
}
```

要查看安全组的初始信息，请运行 aws ec2 describe-security-groups 命令。您不能仅通过其 vpc-id 而非其名称引用 EC2-VPC 安全组。

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
```

```
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": []
                }
            ],
            "Description": "My security group"
            "IpPermissions": [],
            "GroupName": "my-sg",
            "VpcId": "vpc-1a2b3c4d",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

为您的安全组添加规则

运行 Amazon EC2 实例时，您必须启用安全组中的规则，以便为连接到镜像的方法允许入站网络流量。

例如，如果您要启动 Windows 实例，您通常会添加规则以允许 TCP 端口 3389 (RDP) 上的入站流量，从而支持远程桌面协议 (RDP)。如果您要启动 Linux 实例，则通常会添加规则以允许 TCP 端口 22 上的入站流量，从而支持 SSH 连接。

使用 aws ec2 authorize-security-group-ingress 命令向安全组添加规则。此命令的必需参数是您电脑的公有 IP 地址，或您的电脑连接到的网络（以地址范围形式）（采用 CIDR 表示法）。

以下示例说明如何使用 IP 地址将适用于 RDP 的规则（TCP 端口 3389）添加到 ID 为 sg-903004f8 的 EC2-VPC 安全组。

首先，确定您的 IP 地址。

```
$ curl https://checkip.amazonaws.com
x.x.x.x
```

然后，您可以运行 aws ec2 authorize-security-group-ingress 命令以将 IP 地址添加到安全组。

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port
 3389 --cidr x.x.x.x/x
```

以下命令添加另一个规则以对同一安全组中的实例启用 SSH。

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port
 22 --cidr x.x.x.x/x
```

要查看对安全组所做的更改，请运行 aws ec2 describe-security-groups 命令。

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": []
                }
            ],
            "Description": "My security group"
            "IpPermissions": [
                {
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "IpRanges": [
                        {
                            "CidrIp": "x.x.x.x/x"
                        }
                    ]
                    "UserIdGroupPairs": [],
                    "FromPort": 22
                }
            ],
            "GroupName": "my-sg",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8"
```

```
        }
    ]
}
```

删除您的安全组

要删除安全组，请运行 aws ec2 delete-security-group 命令。

> **Note**
>
> 如果安全组当前已附加到环境，则无法删除。

以下命令示例删除 EC2-VPC 安全组。

```
$ aws ec2 delete-security-group --group-id sg-903004f8
```

参考信息

Amazon CLI 参考：

- aws ec2
- aws ec2 authorize-security-group-ingress
- aws ec2 create-security-group
- aws ec2 delete-security-group
- aws ec2 describe-security-groups

其他参考资料：

- Amazon Elastic Compute Cloud 文档
- 要查看和贡献 Amazon 开发工具包和 Amazon CLI 代码示例，请参阅 GitHub 上的 Amazon 代码示例存储库。

在 Amazon CLI 中启动、列出和删除 Amazon EC2 实例

您可以使用 Amazon Command Line Interface（Amazon CLI）启动、列出和删除 Amazon Elastic Compute Cloud（Amazon EC2）实例。如果您启动不在 Amazon 免费套餐范围内的实例，那么在启动实例后您将需要付费，费用按实例的运行时间计算，即使实例处于闲置状态也需付费。

> **Note**
>
> 有关其他命令示例，请参阅《Amazon CLI 参考指南》。

主题

- 先决条件
- 启动实例
- 向实例添加块储存设备
- 向您的实例添加标签
- 连接到您的实例
- 列出您的实例
- 删除实例
- 引用

先决条件

要运行本主题中的 ec2 命令，您需要先完成以下操作：

- 安装和配置 Amazon CLI。有关更多信息，请参阅 安装 Amazon CLI 和 Amazon CLI 身份验证和访问凭证。
- 设置您的 IAM 权限以允许访问 Amazon EC2。有关 Amazon EC2 的 IAM 权限的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 的 IAM 策略。
- 创建密钥对和安全组。
- 选择 Amazon Machine Image（AMI）并记下 AMI ID。有关更多信息，请参阅《Amazon EC2 用户指南》中的查找适合的 AMI。

启动实例

要使用所选的 AMI 启动单个 Amazon EC2 实例，请使用 `aws ec2 run-instances` 命令。您可以将实例启动到虚拟私有云（VPC）中。

最初，您的实例显示为 `pending` 状态，但在几分钟后将更改为 `running` 状态。

以下示例在 VPC 的指定子网中启动 `t2.micro` 实例。将##参数值替换为您自己的值。

```
$ aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t2.micro --
key-name MyKeyPair --security-group-ids sg-903004f8 --subnet-id subnet-6e7f829e
{
    "OwnerId": "123456789012",
    "ReservationId": "r-5875ca20",
    "Groups": [
        {
            "GroupName": "my-sg",
            "GroupId": "sg-903004f8"
        }
    ],
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": null,
            "Platform": "windows",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "EbsOptimized": false,
            "LaunchTime": "2013-07-19T02:42:39.000Z",
            "PrivateIpAddress": "10.0.1.114",
            "ProductCodes": [],
            "VpcId": "vpc-1a2b3c4d",
            "InstanceId": "i-5203422c",
            "ImageId": "ami-173d747e",
            "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
            "KeyName": "MyKeyPair",
            "SecurityGroups": [
                {
                    "GroupName": "my-sg",
                    "GroupId": "sg-903004f8"
                }
            ],
            "ClientToken": null,
            "SubnetId": "subnet-6e7f829e",
            "InstanceType": "t2.micro",
            "NetworkInterfaces": [
                {
                    "Status": "in-use",
```

```
                "SourceDestCheck": true,
                "VpcId": "vpc-1a2b3c4d",
                "Description": "Primary network interface",
                "NetworkInterfaceId": "eni-a7edb1c9",
                "PrivateIpAddresses": [
                    {
                        "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
                        "Primary": true,
                        "PrivateIpAddress": "10.0.1.114"
                    }
                ],
                "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
                "Attachment": {
                    "Status": "attached",
                    "DeviceIndex": 0,
                    "DeleteOnTermination": true,
                    "AttachmentId": "eni-attach-52193138",
                    "AttachTime": "2013-07-19T02:42:39.000Z"
                },
                "Groups": [
                    {
                        "GroupName": "my-sg",
                        "GroupId": "sg-903004f8"
                    }
                ],
                "SubnetId": "subnet-6e7f829e",
                "OwnerId": "123456789012",
                "PrivateIpAddress": "10.0.1.114"
            }
        ],
        "SourceDestCheck": true,
        "Placement": {
            "Tenancy": "default",
            "GroupName": null,
            "AvailabilityZone": "us-west-2b"
        },
        "Hypervisor": "xen",
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/sda1",
                "Ebs": {
                    "Status": "attached",
                    "DeleteOnTermination": true,
                    "VolumeId": "vol-877166c8",
```

```
                    "AttachTime": "2013-07-19T02:42:39.000Z"
                }
            }
        ],
        "Architecture": "x86_64",
        "StateReason": {
            "Message": "pending",
            "Code": "pending"
        },
        "RootDeviceName": "/dev/sda1",
        "VirtualizationType": "hvm",
        "RootDeviceType": "ebs",
        "Tags": [
            {
                "Value": "MyInstance",
                "Key": "Name"
            }
        ],
        "AmiLaunchIndex": 0
    }
]
}
```

向实例添加块储存设备

每个启动的实例都具有关联的根设备卷。您可以使用块储存设备映射来指定实例启动时要连接的其他 Amazon Elastic Block Store (Amazon EBS) 卷或实例存储卷。

要向实例添加块储存设备，请在使用 `--block-device-mappings` 时指定 `run-instances` 选项。

以下示例参数配置大小为 20 GB 的标准 Amazon EBS 卷，并使用标识符 `/dev/sdf` 将其映射到您的实例。

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"VolumeSize\":20,
\"DeleteOnTermination\":false}}]"
```

以下示例基于现有快照添加映射到 `/dev/sdf` Amazon EBS 卷。快照表示加载到卷的镜像。指定快照时，无需指定卷大小；它的大小足够容纳您的镜像。但是，如果您确定指定大小，则大小必须大于或等于快照的大小。

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"SnapshotId\":\"snap-
a1b2c3d4\"}}]"
```

以下示例向实例添加两个卷。可用于您的实例的卷的数目取决于其实例类型。

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"VirtualName\":\"ephemeral0\"},
{\"DeviceName\":\"/dev/sdg\",\"VirtualName\":\"ephemeral1\"}]"
```

以下示例创建映射 (/dev/sdj)，但未为实例预配置卷。

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdj\",\"NoDevice\":\"\"}]"
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的块设备映射。

向您的实例添加标签

标签是为 Amazon 资源分配的标记。它允许您向您可用于各种目的的资源添加元数据。有关更多信息，请参阅《Amazon EC2 用户指南》中的标记您的资源。

以下示例显示如何使用 aws ec2 create-tags 命令，将带有密钥名称“Name”和值“MyInstance”的标签添加到指定的实例。

```
$ aws ec2 create-tags --resources i-5203422c --tags Key=Name,Value=MyInstance
```

连接到您的实例

当您的实例运行时，您可以连接到该实例，然后像使用您面前的电脑一样使用该实例。有关更多信息，请参阅《Amazon EC2 用户指南》中的连接到您的 Amazon EC2 实例。

列出您的实例

您可以使用 Amazon CLI 列出您的实例并查看有关这些实例的信息。您可以列出所有实例，或根据您感兴趣的实例对结果进行筛选。

以下示例演示了如何使用 aws ec2 describe-instances 命令。

以下命令列出您的所有实例。

```
$ aws ec2 describe-instances
```

以下命令将列表筛选到只仅限您的 t2.micro 实例，并仅为每个匹配项输出 InstanceId 值。

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query
 "Reservations[].Instances[].InstanceId"
[
    "i-05e998023d9c69f9a"
]
```

以下命令列出具有标签 Name=MyInstance 的任何实例。

```
$ aws ec2 describe-instances --filters "Name=tag:Name,Values=MyInstance"
```

以下命令列出使用以下任何 AMI 启动的实例：ami-x0123456、ami-y0123456 和 ami-z0123456。

```
$ aws ec2 describe-instances --filters "Name=image-id,Values=ami-x0123456,ami-
y0123456,ami-z0123456"
```

删除实例

当您不再需要某个 Amazon EC2 实例时，可以使用 Amazon CLI 来终止（删除）该实例。

> **Important**
>
> 终止实例是永久且不可逆转的。
> 终止一个实例后，您将无法再连接到该实例，而且也无法对其进行恢复操作。被配置为在终止后删除的所有连接的 Amazon EBS 卷也将被永久删除并且无法恢复。实例存储卷中存储的所有数据将永久丢失。有关更多信息，请参阅实例终止的工作原理。
> 在终止实例之前，请务必确保已将需要在终止后保留的所有数据备份到持久存储中。

一旦实例的状态变为 shutting-down 或 terminated，您即停止为该实例付费。如果您希望稍后重新连接到实例，请使用 stop-instances，而不是 terminate-instances。有关更多信息，请参阅《Amazon EC2 用户指南》中的终止实例。

以下示例说明如何使用 aws ec2 terminate-instances 命令删除实例。

```
$ aws ec2 terminate-instances --instance-ids i-5203422c
{
    "TerminatingInstances": [
        {
```

```
            "InstanceId": "i-5203422c",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

引用

Amazon CLI 参考：

- aws ec2
- aws ec2 create-tags
- aws ec2 describe-instances
- aws ec2 run-instances
- aws ec2 terminate-instances

其他参考资料：

- Amazon Elastic Compute Cloud 文档
- 要查看和贡献 Amazon 开发工具包和 Amazon CLI 代码示例，请参阅 GitHub 上的 Amazon 代码示例存储库。

在 Amazon CLI 中使用 bash 脚本更改 Amazon EC2 实例类型

Amazon EC2 的这一 bash 脚本示例使用 Amazon Command Line Interface (Amazon CLI) 更改 Amazon EC2 实例的实例类型。如果实例正在运行，它会停止实例，更改实例类型，然后根据请求重启实例。Shell 脚本是专用于在命令行界面中运行的程序。

> **Note**
>
> 有关其他命令示例，请参阅 Amazon CLI 参考指南。

主题

- 开始之前
- 关于此示例
- 参数
- 文件
- 参考信息

开始之前

您需要先满足以下条件，才能运行下文中的任何示例代码。

- 安装和配置 Amazon CLI。有关更多信息，请参阅安装 Amazon CLI和Amazon CLI 身份验证和访问凭证。
- 您使用的配置文件必须具有允许示例代码执行 Amazon 操作的权限。
- 在账户中有正在运行的 Amazon EC2 实例，并且您有权停止和修改该实例。如果您运行测试脚本，它会为您启动一个实例，测试更改类型，然后终止该实例。
- 作为 Amazon 的最佳实践，请授予此代码最低的权限，或者仅授予它执行任务所需的权限。有关更多信息，请参阅 Amazon Identity and Access Management (IAM) 用户指南中的授予最低权限。
- 此代码尚未在所有 Amazon 区域中进行测试。有些 Amazon 服务仅在特定区域中提供。有关更多信息，请参阅《Amazon 一般参考指南》中的服务端点和配额。
- 运行此代码可能会导致您的 Amazon 账户产生相关费用。您有责任确保在使用完由此脚本创建的任何资源后删除这些资源。

关于此示例

此示例是作为 shell 脚本文件 `change_ec2_instance_type.sh` 中的一个函数编写的，您可以通过其他脚本或命令行 `source` 该函数。每个脚本文件都包含了介绍每个函数的注释。一旦该函数进入内存，您就可以通过命令行调用它。例如，以下命令会将指定实例的类型更改为 `t2.nano`：

```
$ source ./change_ec2_instance_type.sh
$ ./change_ec2_instance_type -i *instance-id* -t new-type
```

有关完整示例和可下载的脚本文件，请参阅 GitHub 上的 Amazon 代码示例存储库 中的更改 Amazon EC2 实例类型。

参数

-i - (字符串) 指定要修改的实例 ID。

-t - (字符串) 指定要切换到的 Amazon EC2 实例类型。

-r - (开关) 默认情况下，此参数未设置。如果设置了 `-r`，则在类型切换后将会重启实例。

-f - (开关) 默认情况下，脚本在进行切换之前会提示用户确认关闭实例。如果设置了 `-f`，则函数在关闭实例以进行类型切换之前不会提示用户

-v - (开关) 默认情况下，脚本会以静默方式运行，仅在出现错误时才显示输出。如果设置了 `-v`，则函数会在整个运行过程中显示状态。

文件

`change_ec2_instance_type.sh`

该主脚本文件包含执行以下任务的 `change_ec2_instance_type()` 函数：

- 验证指定的 Amazon EC2 实例是否存在。
- 除非选择了 `-f`，否则函数会在停止实例之前向用户发出警告。
- 更改实例类型
- 如果您设置了 `-r`，请重启实例并确认实例正在运行

在 GitHub 上查看 change_ec2_instance_type.sh 的代码。

`test_change_ec2_instance_type.sh`

脚本文件 `test_change_ec2_instance_type.sh` 会测试 `change_ec2_instance_type` 函数的各种代码路径。如果测试脚本中的所有步骤都正确完成，测试脚本将会删除它创建的所有资源。

您可以使用以下参数运行该测试脚本：

- -v - (开关) 每个测试在运行时都会显示通过/失败状态。默认情况下，测试以静默方式运行，输出仅包含最终的总体通过/失败状态。
- -i - (开关) 脚本会在每个测试后暂停，以便您能查看每个步骤的中间结果。让您能够使用 Amazon EC2 控制台检查实例的当前状态。您在系统提示符处按 Enter 后，脚本才会继续执行下一步。

在 GitHub 上查看 test_change_ec2_instance_type.sh 的代码。

awsdocs_general.sh

脚本文件 awsdocs_general.sh 中包含了在 Amazon CLI 的高级示例中使用的通用函数。

在 GitHub 上查看 awsdocs_general.sh 的代码。

参考信息

Amazon CLI 参考：

- aws ec2
- aws ec2 describe-instances
- aws ec2 modify-instance-attribute
- aws ec2 start-instances
- aws ec2 stop-instances
- aws ec2 wait instance-running
- aws ec2 wait instance-stopped

其他参考资料：

- Amazon Elastic Compute Cloud 文档
- 要查看和贡献 Amazon 开发工具包和 Amazon CLI 代码示例，请参阅 GitHub 上的 Amazon 代码示例存储库。

在 Amazon CLI 中使用 Amazon Glacier

Amazon Glacier 简介
Amazon Glacier 简介

本主题显示执行 Amazon Glacier 常见任务的 Amazon CLI 命令的一些示例。这些示例演示如何使用 Amazon CLI 将大型文件上传到 Amazon Glacier，方法是将它拆分为较小的部分并从命令行上传它们。

您可以使用 Amazon Command Line Interface（Amazon CLI）访问 Amazon Glacier 功能。要列出 Amazon Glacier 的 Amazon CLI 命令，请使用以下命令。

```
aws glacier help
```

> **Note**
>
> 有关命令参考和其他示例，请参阅《Amazon CLI 命令参考》中的 `aws glacier`。

主题

- 先决条件
- 创建 Amazon Glacier 保管库
- 准备要上传的文件
- 启动文件分段上传和上传
- 完成上传
- 资源

先决条件

要运行 `glacier` 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅 安装 Amazon CLI 和 Amazon CLI 身份验证和访问凭证。
- 本教程使用几个命令行工具，这些工具通常预装在 Unix 类操作系统上，包括 Linux 和 macOS。Windows 用户可以通过安装 Cygwin 并从 Cygwin 终端运行命令来使用相同的工具。如有可执行相同功能的 Windows 本机命令和实用工具，我们会注明。

创建 Amazon Glacier 保管库

使用 `create-vault` 命令创建文件库。

```
$ aws glacier create-vault --account-id - --vault-name myvault
{
    "location": "/123456789012/vaults/myvault"
```

```
}
```

> **Note**
>
> 所有 Amazon Glacier 命令都需要一个账户 ID 参数。使用连字符 (`--account-id -`) 以使用当前账户。

准备要上传的文件

创建一个用于测试上传的文件。以下命令将创建一个正好包含 3 MiB 随机数据的名为 *largefile* 的文件。

Linux 或 macOS

```
$ dd if=/dev/urandom of=largefile bs=3145728 count=1
1+0 records in
1+0 records out
3145728 bytes (3.1 MB) copied, 0.205813 s, 15.3 MB/s
```

dd 是一个实用工具，该实用工具将大量字节从输入文件复制到输出文件。上一个示例使用系统设备文件 `/dev/urandom` 作为随机数据的源。`fsutil` 在 Windows 中执行相似的功能。

Windows

```
C:\> fsutil file createnew largefile 3145728
File C:\temp\largefile is created
```

接下来，使用文件拆分器将文件拆分为 1 MiB（1,048,576 字节）的块。

```
$ split -b 1048576 --verbose largefile chunk
creating file `chunkaa'
creating file `chunkab'
creating file `chunkac'
```

启动文件分段上传和上传

使用 `initiate-multipart-upload` 命令在 Amazon Glacier 中创建分段上传。

```
$ aws glacier initiate-multipart-upload --account-id - --archive-description "multipart
 upload test" --part-size 1048576 --vault-name myvault
{
    "uploadId": "19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ",
    "location": "/123456789012/vaults/myvault/multipart-
uploads/19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
}
```

Amazon Glacier 需要每个部分的大小以字节为单位（本例中以 1 MiB 为单位）、您的保管库名称和一个账户 ID，用来配置分段上传。操作完成时，Amazon CLI 会输出一个上传 ID。将上传 ID 保存到 shell 变量以待将来使用。

Linux 或 macOS

```
$ UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
```

Windows

```
C:\> set UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ"
```

接下来，使用 upload-multipart-part 命令上传三个部分的每个部分。

```
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkaa --range 'bytes
 0-1048575/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkab --range 'bytes
 1048576-2097151/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkac --range 'bytes
 2097152-3145727/*' --account-id - --vault-name myvault
{
    "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
```

> **Note**
>
> 上一个示例使用美元符号 ($) 在 Linux 上引用 UPLOADID shell 变量的内容。在 Windows 命令行上，在变量名称（例如，%UPLOADID%）的任一侧使用百分号 (%)。

在上传各个部分时，您必须指定其字节范围，以便 Amazon Glacier 可以按正确的顺序重组它。由于每个部分的大小为 1,048,576 字节，因此第一个部分占用 0-1048575 字节，第二个部分占用 1048576-2097151 字节，第三个部分占用 2097152-3145727 字节。

完成上传

Amazon Glacier 需要原始文件的树形哈希，以确认所有上传的部分都已完整地到达Amazon。

要计算树形哈希，必须将文件拆分为 1 MiB 的部分并计算每个部分的二进制 SHA-256 哈希。然后，将哈希列表拆分成对，合并每对中的两个二进制哈希，并采用结果的哈希。重复此步骤，直到只剩下一个哈希。如果任一级别出现奇数数量的哈希，请将其提升到下一级别而无需修改。

在使用命令行实用工具时，正确计算树形哈希的关键是以二进制的形式存储每个哈希，并且仅在最后一步将其转换为十六进制。对树中的任何哈希的十六进制版本进行合并或哈希处理将导致错误结果。

> **Note**
>
> Windows 用户可使用 type 命令来代替 cat。OpenSSL 适用于 Windows，可在 OpenSSL.org 找到。

计算树形哈希

1. 将原始文件拆分为 1 MiB 的部分（如果您还没有这样做）。

```
$ split --bytes=1048576 --verbose largefile chunk
creating file `chunkaa'
creating file `chunkab'
creating file `chunkac'
```

2. 计算并存储每个区块的二进制 SHA-256 哈希。

```
$ openssl dgst -sha256 -binary chunkaa > hash1
```

```
$ openssl dgst -sha256 -binary chunkab > hash2
$ openssl dgst -sha256 -binary chunkac > hash3
```

3. 合并前两个哈希，并采用结果的二进制哈希。

```
$ cat hash1 hash2 > hash12
$ openssl dgst -sha256 -binary hash12 > hash12hash
```

4. 将区块 aa 和 ab 的父哈希与区块 ac 的哈希合并并对结果进行哈希处理，此时将输出十六进制。将结果存储在 shell 变量中。

```
$ cat hash12hash hash3 > hash123
$ openssl dgst -sha256 hash123
SHA256(hash123)= 9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
$ TREEHASH=9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
```

最后，使用 complete-multipart-upload 命令完成上传。此命令采用原始文件的大小（以字节为单位）、最终树形哈希值（十六进制形式）以及您的账户 ID 和文件库名称。

```
$ aws glacier complete-multipart-upload --checksum $TREEHASH --archive-size 3145728 --
upload-id $UPLOADID --account-id - --vault-name myvault
{
    "archiveId": "d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrAlLGAAONJAzo5QdP-
N83MKqd96Unspoa5H5lItWX-sK8-QS0ZhwsyGiu9-R-
kwWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg",
    "checksum": "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
    "location": "/123456789012/vaults/myvault/archives/
d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrAlLGAAONJAzo5QdP-N83MKqd96Unspoa5H5lItWX-sK8-
QS0ZhwsyGiu9-R-kwWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg"
}
```

您也可以使用 describe-vault 命令查看文件库的状态。

```
$ aws glacier describe-vault --account-id - --vault-name myvault
{
    "SizeInBytes": 3178496,
    "VaultARN": "arn:aws:glacier:us-west-2:123456789012:vaults/myvault",
    "LastInventoryDate": "2018-12-07T00:26:19.028Z",
    "NumberOfArchives": 1,
    "CreationDate": "2018-12-06T21:23:45.708Z",
    "VaultName": "myvault"
```

```
}
```

> **Note**
>
> 基本上每天都会更新一次文件库的状态。有关更多信息，请参阅 使用文件库。

现在可以安全删除您创建的块和哈希文件。

```
$ rm chunk* hash*
```

有关分段上传的更多信息，请参阅《Amazon Glacier 开发人员指南》中的分段上传大型档案和计算校验和。

资源

Amazon CLI《参考：》

- aws glacier
- aws glacier complete-multipart-upload
- aws glacier create-vault
- aws glacier describe-vault
- aws glacier initiate-multipart-upload

《服务参考：》

- 《Amazon Glacier 开发人员指南》
- 《Amazon Glacier 开发人员指南》中的分段上传大型档案
- 《Amazon Glacier 开发人员指南》中的计算校验和
- 《Amazon Glacier 开发人员指南》中的处理保管库

在 Amazon CLI 中使用 IAM

Amazon Identity and Access Management 简介
Amazon Identity and Access Management 简介

您可以使用 Amazon Command Line Interface (Amazon CLI) 访问 Amazon Identity and Access Management (IAM) 的功能。要列出 IAM 的 Amazon CLI 命令，请使用以下命令。

```
aws iam help
```

本主题显示执行 IAM 常见任务的 Amazon CLI 命令。

在运行任何命令之前，请设置默认证书。有关更多信息，请参阅 配置 Amazon CLI 设置。

有关 IAM 服务的更多信息，请参阅 Amazon Identity and Access Management 用户指南。

主题

- 创建 IAM 用户和组
- 将 IAM 托管式策略附加到用户
- 为 IAM 用户设置初始密码
- 为 IAM 用户创建访问密钥

创建 IAM 用户和组

创建组并向其中添加新用户

1. 使用 create-group 命令创建组。

```
$ aws iam create-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2018-12-14T03:03:52.834Z",
        "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    }
}
```

2. 使用 create-user 命令创建用户。

```
$ aws iam create-user --user-name MyUser
{
    "User": {
        "UserName": "MyUser",
```

```
        "Path": "/",
        "CreateDate": "2018-12-14T03:13:02.581Z",
        "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/MyUser"
    }
}
```

3. 使用 add-user-to-group 命令将用户添加到组。

```
$ aws iam add-user-to-group --user-name MyUser --group-name MyIamGroup
```

4. 要验证 MyIamGroup 组包含 MyUser，请使用 get-group 命令。

```
$ aws iam get-group --group-name MyIamGroup
{
    "Group": {
        "GroupName": "MyIamGroup",
        "CreateDate": "2018-12-14T03:03:52Z",
        "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
        "Path": "/"
    },
    "Users": [
        {
            "UserName": "MyUser",
            "Path": "/",
            "CreateDate": "2018-12-14T03:13:02Z",
            "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/MyUser"
        }
    ],
    "IsTruncated": "false"
}
```

将 IAM 托管式策略附加到用户

此示例中的策略为用户提供“高级用户访问”。

将 IAM 托管式策略附加到用户

1. 确定要附加的策略的 Amazon 资源名称 (ARN)。以下命令使用 list-policies 查找具有名称 PowerUserAccess 的策略的 ARN。然后，它会将该 ARN 存储在环境变量中。

```
$ export POLICYARN=$(aws iam list-policies --query 'Policies[?
PolicyName==`PowerUserAccess`].{ARN:Arn}' --output text)       ~
$ echo $POLICYARN
arn:aws:iam::aws:policy/PowerUserAccess
```

2. 要附加策略，请使用 attach-user-policyattach-user-policy 命令，并引用存放策略 ARN 的环境变量。

```
$ aws iam attach-user-policy --user-name MyUser --policy-arn $POLICYARN
```

3. 通过运行 list-attached-user-policies 命令验证策略已附加到此用户。

```
$ aws iam list-attached-user-policies --user-name MyUser
{
    "AttachedPolicies": [
        {
            "PolicyName": "PowerUserAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
        }
    ]
}
```

有关更多信息，请参阅 访问管理资源。该主题提供权限和策略概览的链接以及用于访问 Amazon S3、Amazon EC2 和其他服务的示例策略的链接。

为 IAM 用户设置初始密码

以下命令使用 create-login-profile 为指定用户设置初始密码。当用户首次登录时，用户需要将密码更改为只有用户知道的内容。

```
$ aws iam create-login-profile --user-name MyUser --password My!User1Login8P@ssword --
password-reset-required
{
    "LoginProfile": {
        "UserName": "MyUser",
        "CreateDate": "2018-12-14T17:27:18Z",
        "PasswordResetRequired": true
    }
}
```

您可以使用 update-login-profile 命令更改用户的密码。

```
$ aws iam update-login-profile --user-name MyUser --password My!User1ADifferentP@ssword
```

为 IAM 用户创建访问密钥

您可以使用 create-access-key 命令为用户创建访问密钥。访问密钥是一组安全凭证，由访问密钥 ID 和私有密钥组成。

用户一次只能创建两个访问密钥。如果您尝试创建第三组，则命令返回 LimitExceeded 错误。

```
$ aws iam create-access-key --user-name MyUser
{
    "AccessKey": {
        "UserName": "MyUser",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2018-12-14T17:34:16Z"
    }
}
```

使用 delete-access-key 命令为用户删除访问密钥。使用访问密钥 ID 指定要删除的访问密钥。

```
$ aws iam delete-access-key --user-name MyUser --access-key-id AKIAIOSFODNN7EXAMPLE
```

在 Amazon CLI 中使用 Amazon S3

Amazon Simple Storage Service (Amazon S3) 简介
Amazon Simple Storage Service (Amazon S3) 简介 - Amazon 上的云存储

您可以使用 Amazon Command Line Interface (Amazon CLI) 访问 Amazon Simple Storage Service (Amazon S3) 的功能。Amazon S3 是一种高度可扩展的持久对象存储服务。Amazon S3 旨在提供几乎无限的存储容量，使其成为满足各种数据存储和管理需求的理想解决方案。

借助 Amazon S3，您能够以对象的形式存储和检索从小文件到大型数据集的任意数量的数据。每个对象都存储在名为存储桶的容器中，可以通过 Amazon Web Services 管理控制台或 Amazon SDK、工具和 Amazon CLI 以编程方式访问和管理存储桶。

除基本存储之外，Amazon S3 还提供一系列功能，包括生命周期管理、版本控制、可扩展性和安全性。这些功能可与其他 Amazon Web Services 服务集成，使您能够构建可根据需要扩展的云端解决方案。

Amazon CLI 提供两个层级的命令来访问 Amazon S3：

- s3 – 自定义专门针对 Amazon CLI 执行的高级命令，以简化常见任务，比如创建、操作、删除和同步对象及存储桶。
- s3api – 提供对所有 Amazon S3 API 操作的直接访问，使您能够执行高级操作。

本指南中的主题：

- 在 Amazon CLI 中使用高级（s3）命令
- 在 Amazon CLI 中使用 API 级 (s3api) 命令
- Amazon CLI 中的 Amazon S3 存储桶生命周期脚本示例

在 Amazon CLI 中使用高级（s3）命令

本主题介绍一些命令，可用于在 Amazon CLI 中通过 `aws s3` 命令管理 Amazon S3 存储桶和对象。有关本主题未涵盖的命令和其他命令示例，请参阅《Amazon CLI 参考》中的 `aws s3` 命令。

高级别 `aws s3` 命令简化了 Amazon S3 对象管理。使用这些命令，您能够在 Amazon S3 自身中管理其内容以及使用本地目录管理这些内容。

主题

- 先决条件
- 开始之前
- 创建存储桶
- 列出存储桶和对象
- 删除存储桶
- 删除对象
- 移动对象
- 复制对象
- 同步对象
- s3 命令的常用选项

- 资源

先决条件

要运行 s3 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅 安装 Amazon CLI 和 Amazon CLI 身份验证和访问凭证。
- 您使用的配置文件必须具有允许示例代码执行Amazon操作的权限。
- 需了解如下 Amazon S3 术语：
  - 存储桶 – 顶级 Amazon S3 文件夹。
  - 前缀 – 存储桶中的 Amazon S3 文件夹。
  - 对象 – 托管在 Amazon S3 存储桶中的任何项。

开始之前

本节介绍在使用 aws s3 命令之前需要注意的一些事项。

大型对象上载

当您使用 aws s3 命令将大型对象上传到 Amazon S3 存储桶时，Amazon CLI 会自动执行分段上传。使用这些 aws s3 命令时，您无法恢复失败的上传操作。

如果分段上传由于超时而失败，或者您在 Amazon CLI 中手动取消该操作，则 Amazon CLI 会停止上传并清除已创建的任何文件。此过程可能耗时数分钟。

如果使用 kill 命令或者由于系统故障而取消了分段上传或清理过程，则创建的文件将保留在 Amazon S3 存储桶中。要清除分段上传，请使用 s3api abort-multipart-upload 命令。

分段复制中的文件属性和标签

当您使用 aws s3 命名空间中的 Amazon CLI 版本 1 版本命令将文件从一个 Amazon S3 存储桶位置复制到另一个 Amazon S3 存储桶位置，并且该操作使用分段复制时，源对象中的文件属性不会被复制到目标对象。

创建存储桶

使用 s3 mb 命令创建存储桶。存储桶名称必须是全局 唯一的（在所有 Amazon S3 存储桶中都是唯一的），并且应符合 DNS 标准。

存储桶名称可以包含小写字母、数字、连字符和点号。存储桶名称只能以字母或数字开头和结尾，连字符或点号后不能跟点号。

语法

```
$ aws s3 mb <target> [--options]
```

s3 mb 示例

以下示例将创建 s3://amzn-s3-demo-bucket 存储桶。

```
$ aws s3 mb s3://amzn-s3-demo-bucket
```

列出存储桶和对象

要列出存储桶、文件夹或对象，请使用 s3 ls 命令。使用不带目标或选项的命令时，将会列出所有存储桶。

语法

```
$ aws s3 ls <target> [--options]
```

有关与此命令一起使用的一些常见选项以及相关示例，请参阅 s3 命令的常用选项。有关可用选项的完整列表，请参阅《Amazon CLI 命令参考》中的 s3 ls。

s3 ls 示例

以下示例列出您的所有 Amazon S3 存储桶。

```
$ aws s3 ls
2018-12-11 17:08:50 amzn-s3-demo-bucket1
2018-12-14 14:55:44 amzn-s3-demo-bucket2
```

下面的命令列出一个存储桶中的所有对象和前缀。在本示例输出中，前缀 example/ 有一个名为 MyFile1.txt 的文件。

```
$ aws s3 ls s3://amzn-s3-demo-bucket
                           PRE example/
2018-12-04 19:05:48          3 MyFile1.txt
```

您可以通过在命令中包含特定前缀将输出筛选到此前缀。下面的命令列出 *bucket-name/example/* 中的对象（即 *bucket-name* 中按前缀 *example/* 筛选出的对象）。

```
$ aws s3 ls s3://amzn-s3-demo-bucket/example/
2018-12-06 18:59:32          3 MyFile1.txt
```

要仅显示特定区域内的存储桶和对象，请使用 `--region` 选项

```
$ aws s3 ls --region us-east-2
2018-12-06 18:59:32          3 MyFile1.txt
```

如果您的存储桶和对象列表比较大，可以使用 `--max-items` 或 `--page-size` 选项对结果进行分页。`--max-items` 选项可限制调用中返回的存储桶和对象的总数，`--page-size` 选项可限制页面上列出的存储桶和对象的数量。

```
$ aws s3 ls --max-items 100 --page-size 10
```

有关分页的更多信息，请参阅the section called “--page-size”和the section called “--max-items”。

删除存储桶

要删除存储桶，请使用 `s3 rb` 命令。

语法

```
$ aws s3 rb <target> [--options]
```

s3 rb 示例

以下示例删除 `s3://amzn-s3-demo-bucket` 存储桶。

```
$ aws s3 rb s3://amzn-s3-demo-bucket
```

默认情况下，存储桶必须为空，此操作才能成功。要删除不为空的存储桶，您必须包含 `--force` 选项。如果您使用的是受版本控制的存储桶，即其中包含以前删除“但仍保留”的对象，则此命令 不 允许您删除该存储桶。您必须先删除所有内容。

以下示例命令删除存储桶中的所有对象和前缀，然后删除存储桶本身。

```
$ aws s3 rb s3://amzn-s3-demo-bucket --force
```

删除对象

要删除存储桶或本地目录中的对象，请使用 s3 rm 命令。

语法

```
$ aws s3 rm  <target> [--options]
```

有关与此命令一起使用的一些常见选项以及相关示例，请参阅 s3 命令的常用选项。有关选项的完整列表，请参阅《Amazon CLI 命令参考》中的 s3 rm。

s3 rm 示例

以下示例将从 filename.txt 删除 s3://amzn-s3-demo-bucket/example。

```
$ aws s3 rm s3://amzn-s3-demo-bucket/example/filename.txt
```

以下示例使用 s3://amzn-s3-demo-bucket/example 选项从 --recursive 删除所有对象。

```
$ aws s3 rm s3://amzn-s3-demo-bucket/example --recursive
```

移动对象

使用 s3 mv 命令可从存储桶或本地目录中移动对象。s3 mv 命令将源对象或文件复制到指定目标，然后删除源对象或文件。

语法

```
$ aws s3 mv <source> <target> [--options]
```

有关与此命令一起使用的一些常见选项以及相关示例，请参阅 s3 命令的常用选项。有关可用选项的完整列表，请参阅《Amazon CLI 命令参考》中的 s3 mv。

> **Warning**
>
> 如果您在 Amazon S3 源或目标 URI 中使用任何类型的接入点 ARN 或接入点别名，则必须格外注意源和目标 Amazon S3 URI 解析到不同的底层存储桶。如果源存储桶和目标存储桶相

同，则可以将源文件或对象移到其自身上，这可能会导致源文件或对象被意外删除。要验证源存储桶和目标存储桶是否不同，请使用 `--validate-same-s3-paths` 参数或将环境变量 `AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS` 设置为 `true`。

s3 mv 示例

以下示例将所有对象从 `s3://amzn-s3-demo-bucket/example` 移动到 `s3://amzn-s3-demo-bucket/`。

```
$ aws s3 mv s3://amzn-s3-demo-bucket/example s3://amzn-s3-demo-bucket/
```

以下示例使用 `s3 mv` 命令，将本地文件从当前工作目录移动到 Amazon S3 存储桶。

```
$ aws s3 mv filename.txt s3://amzn-s3-demo-bucket
```

以下示例将文件从 Amazon S3 存储桶移动到当前工作目录，其中 `./` 指定当前的工作目录。

```
$ aws s3 mv s3://amzn-s3-demo-bucket/filename.txt ./
```

复制对象

使用 `s3 cp` 命令可从存储桶或本地目录复制对象。

语法

```
$ aws s3 cp <source> <target> [--options]
```

您可以使用短划线参数，将文件流式传输到标准输入（`stdin`）或标准输出（`stdout`）。

Warning

如果您使用的是 PowerShell，则 Shell 可能会更改 CRLF 编码，或者将 CRLF 添加到管道输入或输出或重定向的输出中。

`s3 cp` 命令使用以下语法，将文件流从 `stdin` 上传到指定的存储桶。

语法

```
$ aws s3 cp - <target> [--options]
```

s3 cp 命令使用以下语法下载 stdout 的 Amazon S3 文件流。

语法

```
$ aws s3 cp <target> [--options] -
```

有关与此命令一起使用的一些常见选项以及相关示例，请参阅 s3 命令的常用选项。有关选项的完整列表，请参阅《Amazon CLI 命令参考》中的 s3 cp。

s3 cp 示例

以下示例将所有对象从 s3://amzn-s3-demo-bucket/example 复制到 s3://amzn-s3-demo-bucket/。

```
$ aws s3 cp s3://amzn-s3-demo-bucket/example s3://amzn-s3-demo-bucket/
```

以下示例使用 s3 cp 命令，将本地文件从当前工作目录复制到 Amazon S3 存储桶。

```
$ aws s3 cp filename.txt s3://amzn-s3-demo-bucket
```

以下示例将文件从 Amazon S3 存储桶复制到当前工作目录，其中 ./ 指定当前的工作目录。

```
$ aws s3 cp s3://amzn-s3-demo-bucket/filename.txt ./
```

以下示例使用 echo 将文本“hello world”流式传输到 s3://bucket-name/filename.txt 文件。

```
$ echo "hello world" | aws s3 cp - s3://amzn-s3-demo-bucket/filename.txt
```

以下示例将 s3://amzn-s3-demo-bucket/filename.txt 文件流式传输到 stdout，并将内容输出到控制台。

```
$ aws s3 cp s3://amzn-s3-demo-bucket/filename.txt -
hello world
```

以下示例将 s3://bucket-name/pre 的内容流式传输到 stdout，使用 bzip2 命令压缩文件，并将名为 key.bz2 的新压缩文件上传到 s3://bucket-name。

```
$ aws s3 cp s3://amzn-s3-demo-bucket/pre - | bzip2 --best | aws s3 cp - s3://amzn-s3-
demo-bucket/key.bz2
```

同步对象

s3 sync 命令同步一个存储桶与一个目录中的内容，或者同步两个存储桶中的内容。通常，s3 sync 在源和目标之间复制缺失或过时的文件或对象。不过，您还可以提供 --delete 选项来从目标中删除源中不存在的文件或对象。

语法

```
$ aws s3 sync <source> <target> [--options]
```

有关与此命令一起使用的一些常见选项以及相关示例，请参阅 s3 命令的常用选项。有关选项的完整列表，请参阅《Amazon CLI 命令参考》中的 s3 sync。

s3 sync 示例

下面的示例将名为 amzn-s3-demo-bucket 的存储桶中名为 path 的 Amazon S3 前缀中的内容与当前工作目录同步。

s3 sync 将更新与目标位置中同名文件的大小或修改时间不同的任何文件。输出显示在同步期间执行的特定操作。请注意，该操作将与 MySubdirectory 递归地同步子目录 s3://amzn-s3-demo-bucket/path/MySubdirectory 及其内容。

```
$ aws s3 sync . s3://amzn-s3-demo-bucket/path
upload: MySubdirectory\MyFile3.txt to s3://amzn-s3-demo-bucket/path/MySubdirectory/
MyFile3.txt
upload: MyFile2.txt to s3://amzn-s3-demo-bucket/path/MyFile2.txt
upload: MyFile1.txt to s3://amzn-s3-demo-bucket/path/MyFile1.txt
```

下面的示例对上一示例进行了扩展，显示了如何使用 --delete 选项。

```
// Delete local file
$ rm ./MyFile1.txt

// Attempt sync without --delete option - nothing happens
$ aws s3 sync . s3://amzn-s3-demo-bucket/path

// Sync with deletion - object is deleted from bucket
$ aws s3 sync . s3://amzn-s3-demo-bucket/path --delete
```

```
delete: s3://amzn-s3-demo-bucket/path/MyFile1.txt

// Delete object from bucket
$ aws s3 rm s3://amzn-s3-demo-bucket/path/MySubdirectory/MyFile3.txt
delete: s3://amzn-s3-demo-bucket/path/MySubdirectory/MyFile3.txt

// Sync with deletion - local file is deleted
$ aws s3 sync s3://amzn-s3-demo-bucket/path . --delete
delete: MySubdirectory\MyFile3.txt

// Sync with Infrequent Access storage class
$ aws s3 sync . s3://amzn-s3-demo-bucket/path --storage-class STANDARD_IA
```

使用 --delete 选项时，--exclude 和 --include 选项可以筛选要在 s3 sync 操作过程中删除的文件或对象。在这种情况下，参数字符串必须指定要在目标目录或存储桶上下文中包含或排除在删除操作中的文件。下面是一个示例。

```
Assume local directory and s3://amzn-s3-demo-bucket/path currently in sync and each
 contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt
'''

// Sync with delete, excluding files that match a pattern. MyFile88.txt is deleted,
 while remote MyFile1.txt is not.
$ aws s3 sync . s3://amzn-s3-demo-bucket/path --delete --exclude "path/MyFile?.txt"
delete: s3://amzn-s3-demo-bucket/path/MyFile88.txt
'''

// Sync with delete, excluding MyFile2.rtf - local file is NOT deleted
$ aws s3 sync s3://amzn-s3-demo-bucket/path . --delete --exclude "./MyFile2.rtf"
download: s3://amzn-s3-demo-bucket/path/MyFile1.txt to MyFile1.txt
'''

// Sync with delete, local copy of MyFile2.rtf is deleted
$ aws s3 sync s3://amzn-s3-demo-bucket/path . --delete
delete: MyFile2.rtf
```

s3 命令的常用选项

以下选项经常用于本主题中所述的命令。有关可在命令中使用的选项的完整列表，请参阅《Amazon CLI 参考指南》中的特定命令。

acl

s3 sync 和 s3 cp 可以使用 --acl 选项。这样您能够为复制到 Amazon S3 的文件设置访问权限。--acl 选项接受 private、public-read 和 public-read-write 值。有关更多信息，请参阅《Amazon S3 用户指南》中的标准 ACL。

```
$ aws s3 sync . s3://amzn-s3-demo-bucket/path --acl public-read
```

exclude

使用 s3 cp、s3 mv、s3 sync 或 s3 rm 命令时，可以使用 --exclude 或 --include 选项筛选结果。--exclude 选项将规则设置为仅从命令中排除对象，并且系统将按照指定的顺序应用这些选项。如下例所示。

```
Local directory contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt

// Exclude all .txt files, resulting in only MyFile2.rtf being copied
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --exclude "*.txt"

// Exclude all .txt files but include all files with the "MyFile*.txt" format,
 resulting in, MyFile1.txt, MyFile2.rtf, MyFile88.txt being copied
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --exclude "*.txt" --include
 "MyFile*.txt"

// Exclude all .txt files, but include all files with the "MyFile*.txt" format,
 but exclude all files with the "MyFile?.txt" format resulting in, MyFile2.rtf and
 MyFile88.txt being copied
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --exclude "*.txt" --include
 "MyFile*.txt" --exclude "MyFile?.txt"
```

情况如：

使用 s3 cp、s3 mv、s3 sync 或 s3 rm 命令时，可以使用 --exclude 或 --include 选项筛选结果。--include 选项将规则设置为仅包括为命令指定的对象，并且系统将按照指定的顺序应用这些选项。如下例所示。

```
Local directory contains 3 files:
MyFile1.txt
MyFile2.rtf
```

```
MyFile88.txt

// Include all .txt files, resulting in MyFile1.txt and MyFile88.txt being copied
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --include "*.txt"

// Include all .txt files but exclude all files with the "MyFile*.txt" format,
 resulting in no files being copied
$ aws s3 cp . s3://amzn-s3-demo-bucket/path --include "*.txt" --exclude
 "MyFile*.txt"

// Include all .txt files, but exclude all files with the "MyFile*.txt" format, but
 include all files with the "MyFile?.txt" format resulting in MyFile1.txt being
 copied

$ aws s3 cp . s3://amzn-s3-demo-bucket/path --include "*.txt" --exclude
 "MyFile*.txt" --include "MyFile?.txt"
```

授予

s3 cp、s3 mv 和 s3 sync 命令包括一个 --grants 选项，可用来向指定的用户或组授予对对象的权限。使用以下语法对权限列表设置 --grants 选项。将 Permission、Grantee_Type 和 Grantee_ID 替换为您自己的值。

语法

```
--grants Permission=Grantee_Type=Grantee_ID
         [Permission=Grantee_Type=Grantee_ID ...]
```

每个值都包含以下元素：

- *##* – 指定授予的权限。可以设置为 read、readacl、writeacl 或 full。
- *Grantee_Type* – 指定如何标识被授权者。可以设置为 uri、emailaddress 或 id。
- *Grantee_ID* – 根据 *Grantee_Type*.指定被授权者。
  - uri – 组 URI。有关更多信息，请参阅 谁是授权者？
  - emailaddress – 账户的电子邮件地址。
  - id – 账户的规范 ID。

有关 Amazon S3 访问控制的更多信息，请参阅 访问控制。

下面的示例将一个对象复制到一个存储桶中。它授予所有人对对象的 read 权限，向 full 的关联账户授予 read 权限（readacl、writeacl 和 user@example.com）。

```
$ aws s3 cp file.txt s3://amzn-s3-demo-bucket/ --grants read=uri=http://
acs.amazonaws.com/groups/global/AllUsers full=emailaddress=user@example.com
```

还可以为上传到 Amazon S3 的对象指定非默认存储类（REDUCED_REDUNDANCY 或 STANDARD_IA）。为此，请使用 `--storage-class` 选项。

```
$ aws s3 cp file.txt s3://amzn-s3-demo-bucket/ --storage-class REDUCED_REDUNDANCY
```

recursive

使用此选项时，系统针对所指定目录或前缀下的所有文件或对象执行该命令。以下示例删除 `s3://amzn-s3-demo-bucket/path` 及其所有内容。

```
$ aws s3 rm s3://amzn-s3-demo-bucket/path --recursive
```

资源

Amazon CLI《参考：》

- aws s3
- aws s3 cp
- aws s3 mb
- aws s3 mv
- aws s3 ls
- aws s3 rb
- aws s3 rm
- aws s3 sync

《服务参考：》

- 《Amazon S3 开发人员指南》中的使用 Amazon S3 存储桶
- 《Amazon S3 用户指南》中的使用 Amazon S3 对象
- 《Amazon S3 用户指南》中的使用前缀和分隔符按层次结构列出密钥
- 《Amazon S3 用户指南》中的使用 适用于 .NET 的 Amazon SDK（低级别）中止到 S3 存储桶的分段上传

在 Amazon CLI 中使用 API 级 (s3api) 命令

API 级命令（包含在 `s3api` 命令集中）提供对 Amazon Simple Storage Service (Amazon S3) API 的直接访问，并且可以执行高级别 `s3` 命令中未公开的某些操作。这些命令等同于对服务功能提供 API 级访问的其他 Amazon 服务的命令。有关 `s3` 命令的更多信息，请参阅 在 Amazon CLI 中使用高级（s3）命令。

本主题提供了若干示例，以演示如何使用映射到 Amazon S3 API 的低级别命令。此外，您可以在《Amazon CLI 参考指南》的 `s3api` 部分中找到每个 S3 API 命令的示例。

主题

- 先决条件
- 应用自定义 ACL
- 配置日志记录策略
- 资源

先决条件

要运行 `s3api` 命令，您需要：

- 安装和配置 Amazon CLI。有关更多信息，请参阅 安装 Amazon CLI 和 Amazon CLI 身份验证和访问凭证。
- 您使用的配置文件必须具有允许示例代码执行Amazon操作的权限。
- 需了解如下 Amazon S3 术语：
  - 存储桶 – 顶级 Amazon S3 文件夹。
  - 前缀 – 存储桶中的 Amazon S3 文件夹。
  - 对象 – 托管在 Amazon S3 存储桶中的任何项。

应用自定义 ACL

对于高级别命令，您可以使用 `--acl` 选项对 Amazon S3 对象应用预定义的访问控制列表 (ACL)。但不能使用该命令设置存储桶范围的 ACL。不过，您可以通过使用 `put-bucket-acl` API 级命令来执行此操作。

下面的示例说明如何向两个 Amazon 用户（user1@example.com 和 user2@example.com）授予完全控制权限，并向所有人授予读取权限。“everyone”的标识符来自作为参数传递的特殊 URI。

```
$ aws s3api put-bucket-acl --bucket amzn-s3-demo-bucket --grant-full-control
 'emailaddress="user1@example.com",emailaddress="user2@example.com"' --grant-read
 'uri="http://acs.amazonaws.com/groups/global/AllUsers"'
```

有关如何构建 ACL 的详细信息，请参阅 Amazon Simple Storage Service API 参考中的 PUT Bucket acl。CLI 中的 s3api ACL 命令（如 put-bucket-acl）使用相同的 简化参数表示法。

配置日志记录策略

API 命令 put-bucket-logging 配置存储桶日志记录策略。

在下面的示例中，已向Amazon用户 user@example.com 授予对日志文件的完全控制权限，而向所有用户授予了读取访问权限。请注意，还需要使用 put-bucket-acl 命令向 Amazon S3 的日志传输系统（由 URI 指定）授予必要的权限，以读取和向存储桶写入日志。

```
$ aws s3api put-bucket-acl --bucket amzn-s3-demo-bucket --grant-read-acp 'URI="http://
acs.amazonaws.com/groups/s3/LogDelivery"' --grant-write 'URI="http://acs.amazonaws.com/
groups/s3/LogDelivery"'
$ aws s3api put-bucket-logging --bucket amzn-s3-demo-bucket --bucket-logging-status
 file://logging.json
```

上一个命令中的 logging.json 文件具有以下内容。

```
{
  "LoggingEnabled": {
    "TargetBucket": "amzn-s3-demo-bucket",
    "TargetPrefix": "amzn-s3-demo-bucketLogs/",
    "TargetGrants": [
      {
        "Grantee": {
          "Type": "AmazonCustomerByEmail",
          "EmailAddress": "user@example.com"
        },
        "Permission": "FULL_CONTROL"
      },
      {
        "Grantee": {
          "Type": "Group",
          "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
        },
        "Permission": "READ"
```

```
      }
    ]
  }
}
```

资源

《Amazon CLI 参考：》

- `aws s3api`
- `aws s3api put-bucket-acl`
- `aws s3api put-bucket-logging`

《服务参考：》

- 《Amazon S3 开发人员指南》中的使用 Amazon S3 存储桶
- 《Amazon S3 用户指南》中的使用 Amazon S3 对象
- 《Amazon S3 用户指南》中的使用前缀和分隔符按层次结构列出密钥
- 《Amazon S3 用户指南》中的使用 适用于 .NET 的 Amazon SDK（低级别）中止到 S3 存储桶的分段上传

Amazon CLI 中的 Amazon S3 存储桶生命周期脚本示例

本主题介绍使用 Amazon Command Line Interface (Amazon CLI) 的 Amazon S3 存储桶生命周期操作的 bash 脚本示例。此脚本示例使用 `aws s3api` 命令集。Shell 脚本是专用于在命令行界面中运行的程序。

主题

- 在您开始之前
- 关于此示例
- 文件
- 引用

在您开始之前

您需要先满足以下条件，才能运行下文中的任何示例代码。

- 安装和配置 Amazon CLI。有关更多信息，请参阅 安装 Amazon CLI 和 Amazon CLI 身份验证和访问凭证。
- 您使用的配置文件必须具有允许示例代码执行Amazon操作的权限。

- 作为Amazon的最佳实践，请授予此代码最低的权限，或者仅授予它执行任务所需的权限。有关更多信息，请参阅 IAM 用户指南 中的 授予最低权限。
- 此代码尚未在所有Amazon区域中进行测试。有些 Amazon 服务仅在特定区域中提供。有关更多信息，请参阅《Amazon 一般参考指南》中的服务端点和配额。
- 运行此代码可能会导致您的 Amazon 账户产生相关费用。您有责任确保在使用完由此脚本创建的任何资源后删除这些资源。

Amazon S3 服务使用以下术语：

- 存储桶 – 顶级 Amazon S3 文件夹。
- 前缀 – 存储桶中的 Amazon S3 文件夹。
- 对象 – Amazon S3 存储桶中托管的任何项。

关于此示例

此示例说明如何使用 shell 脚本文件中的一组函数与一些基本的 Amazon S3 操作进行交互。这些函数包含在名为 `bucket-operations.sh` 的 shell 脚本文件中。您可以在另一个文件中调用这些函数。每个脚本文件都包含了介绍每个函数的注释。

要查看每个步骤的中间结果，请使用 `-i` 参数运行脚本。您可以使用 Amazon S3 控制台查看存储桶或其内容的当前状态。仅当您在系统提示符处按 Enter 后，脚本才会继续执行下一步。

有关完整示例和可下载的脚本文件，请参阅 GitHub 上的 Amazon 代码示例存储库 中的 Amazon S3 存储桶生命周期操作。

文件

此示例包含以下文件：

bucket-operations.sh

此主脚本文件可以从另一个文件中调取。它包含了执行以下任务的函数：

- 创建存储桶并验证它是否存在

- 将文件从本地电脑复制到存储桶
- 将文件从一个存储桶位置复制到另一个存储桶位置
- 列出存储桶中的内容
- 从存储桶中删除文件
- 删除存储桶

在 GitHub 上查看 bucket-operations.sh 的代码。

test-bucket-operations.sh

shell 脚本文件 test-bucket-operations.sh 演示了如何调用这些函数，即调取 bucket-operations.sh 文件然后调用其中的每个函数。调用函数后，该测试脚本会删除它创建的所有资源。

在 GitHub 上查看 test-bucket-operations.sh 的代码。

awsdocs-general.sh

脚本文件 awsdocs-general.sh 中包含了在 Amazon CLI 的高级代码示例中使用的通用函数。

在 GitHub 上查看 awsdocs-general.sh 的代码。

引用

Amazon CLI 参考：

- aws s3api
- aws s3api create-bucket
- aws s3api copy-object
- aws s3api delete-bucket
- aws s3api delete-object
- aws s3api head-bucket
- aws s3api list-objects
- aws s3api put-object

其他参考资料：

- 《Amazon S3 开发人员指南》中的使用 Amazon S3 存储桶
- 《Amazon S3 用户指南》中的使用 Amazon S3 对象
- 要查看和贡献 Amazon 开发工具包和 Amazon CLI 代码示例，请参阅 GitHub 上的 Amazon 代码示例存储库。

在 Amazon CLI 中访问 Amazon SNS

您可以使用 Amazon Command Line Interface (Amazon CLI)访问 Amazon Simple Notification Service (Amazon SNS) 的功能。要列出适用于 Amazon SNS 的 Amazon CLI 命令，请使用以下命令。

```
aws sns help
```

在运行任何命令之前，请设置默认证书。有关更多信息，请参阅 配置 Amazon CLI 设置。

本主题显示执行 Amazon SNS 常见任务的 Amazon CLI 命令。

主题

- 创建主题
- 订阅主题
- 向主题发布
- 取消订阅主题
- 删除主题

创建主题

要创建主题，请使用 `sns create-topic` 命令并指定要分配给该主题的名称。

```
$ aws sns create-topic --name my-topic
{
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

记下响应的 `TopicArn`，您随后将用它来发布消息。

订阅主题

要订阅主题，请使用 sns subscribe 命令。

以下示例为 email 指定 notification-endpoint 协议和电子邮件地址。

```
$ aws sns subscribe --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --
protocol email --notification-endpoint saanvi@example.com
{
    "SubscriptionArn": "pending confirmation"
}
```

Amazon 通过向您在 subscribe 命令中指定的地址发送电子邮件，立即发送确认电子邮件。电子邮件具有以下文本。

```
You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:123456789012:my-topic
To confirm this subscription, click or visit the following link (If this was in error
 no action is necessary):
Confirm subscription
```

收件人单击确认订阅链接后，收件人的浏览器显示通知消息，信息类似于以下内容。

```
Subscription confirmed!

You have subscribed saanvi@example.com to the topic:my-topic.

Your subscription's id is:
arn:aws:sns:us-west-2:123456789012:my-topic:1328f057-de93-4c15-512e-8bb22EXAMPLE

If it was not your intention to subscribe, click here to unsubscribe.
```

向主题发布

要将消息发送给某一主题的所有订阅者，请使用 sns publish 命令。

以下示例发送消息“Hello World!” 特定主体的所有订阅者。

```
$ aws sns publish --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --
message "Hello World!"
```

```
{
    "MessageId": "4e41661d-5eec-5ddf-8dab-2c867EXAMPLE"
}
```

在本示例中，Amazon 将包含文本“Hello World!”的电子邮件发送 到 saanvi@example.com。

取消订阅主题

要取消订阅某个主题并停止接收向该主题发布的消息，请使用 sns unsubscribe 命令并指定您要取消订阅的主题的 ARN。

```
$ aws sns unsubscribe --subscription-arn arn:aws:sns:us-west-2:123456789012:my-topic:1328f057-de93-4c15-512e-8bb22EXAMPLE
```

要验证您已成功取消订阅，请使用 sns list-subscriptions 命令以确认该 ARN 不再显示在列表中。

```
$ aws sns list-subscriptions
```

删除主题

要删除主题，请运行 sns delete-topic 命令。

```
$ aws sns delete-topic --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic
```

要验证 Amazon 是否已成功删除主题，请使用 sns list-topics 命令以确认该主题不再显示在列表中。

```
$ aws sns list-topics
```

Amazon CLI 命令示例

本主题中的代码示例向您展示了如何将Amazon Command Line Interface 与 Amazon 结合使用。

基本功能是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您展示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 服务结合来完成特定任务的代码示例。

某些服务包含其他示例类别，这些类别说明如何利用特定于服务的库或函数。

Services

- 使用 Amazon CLI 的 ACM 示例
- 使用 Amazon CLI 的 API Gateway 示例
- 使用 Amazon CLI 的 API Gateway HTTP 和 WebSocket API 示例
- 使用 Amazon CLI 的 API Gateway Management API 示例
- 使用 Amazon CLI 的 App Mesh 示例
- 使用 Amazon CLI 的 App Runner 示例
- Amazon AppConfig使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Application Auto Scaling 示例
- 使用 Amazon CLI 的 Application Discovery Service 示例
- 使用 Amazon CLI 的 Application Signals 示例
- 使用 Amazon CLI 的 AppRegistry 示例
- 使用 Amazon CLI 的 Athena 的示例
- 使用 Amazon CLI 的 Auto Scaling 示例
- 使用 Amazon CLI 的 Auto Scaling Plans 示例
- Amazon Backup使用 的 示例Amazon CLI
- Amazon Batch使用 的 示例Amazon CLI
- Amazon Budgets使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon Chime 示例
- 使用 Amazon CLI 的 Cloud Control API 示例
- Amazon Cloud Map使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon Cloud9 示例
- 使用 Amazon CLI 的 Amazon CloudFormation 示例
- 使用 Amazon CLI 的 CloudFront 示例
- 使用 Amazon CLI 的 Amazon CloudSearch 示例
- 使用 Amazon CLI 的 CloudTrail 示例

- 使用 Amazon CLI 的 CloudWatch 示例
- 使用 Amazon CLI 的 CloudWatch Logs 示例
- 使用 Amazon CLI 的 CloudWatch 网络监控示例
- 使用 Amazon CLI 的 CloudWatch Observability Access Monitor 示例
- 使用 Amazon CLI 的 CloudWatch Observability Admin 示例
- 使用 Amazon CLI 的 CloudWatch Synthetics 示例
- 使用 Amazon CLI 的 CodeArtifact 示例
- 使用 Amazon CLI 的 CodeBuild 示例
- 使用 Amazon CLI 的 CodeCommit 示例
- 使用 Amazon CLI 的 CodeDeploy 示例
- 使用 Amazon CLI 的 CodeGuru Reviewer 示例
- 使用 Amazon CLI 的 CodePipeline 示例
- 使用 Amazon CLI 的 AWS CodeStar Notifications 示例
- 使用 Amazon CLI 的 CodeConnections 示例
- 使用 Amazon CLI 的 Amazon Cognito Identity 示例
- 使用 Amazon CLI 的 Amazon Cognito 身份提供者示例
- 使用 Amazon CLI 的 Amazon Comprehend 示例
- 使用 Amazon CLI 的 Amazon Comprehend Medical 示例
- Amazon Config使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon Connect 示例
- Amazon 成本和使用情况报告使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Cost Explorer Service 示例
- 使用 Amazon CLI 的 Firehose 示例
- 使用 Amazon CLI 的 Amazon Data Lifecycle Manager 示例
- Amazon Data Pipeline使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 DataSync 示例
- 使用 Amazon CLI 的 DAX 示例
- 使用 Amazon CLI 的 Detective 示例
- 使用 Amazon CLI 的 Device Farm 示例

- Amazon Direct Connect使用 的 示例Amazon CLI
- Amazon Directory Service使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon Directory Service 数据示例
- Amazon DMS使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon DocumentDB 示例
- 使用 Amazon CLI 的 DynamoDB 示例
- 使用 Amazon CLI 的 DynamoDB Streams 示例
- 使用 Amazon CLI 的 Amazon EC2 示例
- 使用 Amazon CLI 的 Amazon EC2 Instance Connect 示例
- 使用 Amazon CLI 的 Amazon ECR 示例
- 使用 Amazon CLI 的 Amazon ECR Public 示例
- 使用 Amazon CLI 的 Amazon ECS 示例
- 使用 Amazon CLI 的 Amazon EFS 示例
- 使用 Amazon CLI 的 Amazon EKS 示例
- 使用 Amazon CLI 的 Elastic Beanstalk 示例
- ELB – 使用 Amazon CLI 的版本 1 示例
- ELB – 使用 Amazon CLI 的版本 2 示例
- 使用 Amazon CLI 的 ElastiCache 示例
- 使用 Amazon CLI 的 MediaStore 示例
- 使用 Amazon CLI 的 Amazon EMR 示例
- 使用 Amazon CLI 的 Amazon EMR on EKS 示例
- 使用 Amazon CLI 的 EventBridge 示例
- 使用 Amazon CLI 的 EventBridge Pipes 示例
- 使用 Amazon CLI 的 Firewall Manager 示例
- 使用 Amazon CLI 的 Amazon FIS 示例
- 使用 Amazon CLI 的 Amazon GameLift Servers 示例
- 使用Amazon CLI 的 Amazon Glacier 示例
- 使用 Amazon CLI 的 Global Accelerator 示例
- Amazon Glue使用 的 示例Amazon CLI

- 使用 Amazon CLI 的 GuardDuty 示例
- Amazon Health使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 HealthImaging 示例
- 使用 Amazon CLI 的 HealthLake 示例
- 使用 Amazon CLI 的 HealthOmics 示例
- 使用 Amazon CLI 的 IAM 示例
- 使用 Amazon CLI 的 IAM Access Analyzer 示例
- 使用 Amazon CLI 的 Image Builder 示例
- 使用 Amazon CLI 的 Incident Manager 示例
- 使用 Amazon CLI 的 Incident Manager 联系人示例
- 使用 Amazon CLI 的 Amazon Inspector 示例
- Amazon IoT使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon IoT Analytics 示例
- 使用 Amazon CLI 的 Device Advisor 示例
- 使用 Amazon CLI 的 Amazon IoT data示例
- 使用 Amazon CLI 的 Amazon IoT Events示例
- 使用 Amazon CLI 的 Amazon IoT Events-Data示例
- Amazon IoT Greengrass使用 的 示例Amazon CLI
- Amazon IoT Greengrass V2使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon IoT Jobs SDK 发布 示例
- 使用 Amazon CLI 的 Amazon IoT SiteWise 示例
- Amazon IoT Things Graph使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon IoT Wireless 示例
- 使用 Amazon CLI 的 Amazon IVS 示例
- 使用 Amazon CLI 的 Amazon IVS 聊天功能
- 使用 Amazon CLI 的 Amazon IVS 实时直播功能示例
- 使用 Amazon CLI 的 Amazon Kendra 示例
- 使用 Amazon CLI 的 Kinesis 示例
- 使用 Amazon CLI 的 Amazon KMS 示例

- 使用 Amazon CLI 的 Lake Formation 示例
- 使用 Amazon CLI 的 Lambda 示例
- 使用 Amazon CLI 的 License Manager 示例
- Lightsail使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Macie 示例
- 使用 Amazon CLI 的 Amazon Managed Grafana 示例
- 使用 Amazon CLI 的 MediaConnect 示例
- 使用 Amazon CLI 的 MediaConvert 示例
- 使用 Amazon CLI 的 MediaLive 示例
- 使用 Amazon CLI 的 MediaPackage 示例
- 使用 Amazon CLI 的 MediaPackage VOD 示例
- 使用 Amazon CLI 的 MediaStore Data Plane 示例
- 使用 Amazon CLI 的 MediaTailor 示例
- 使用 Amazon CLI 的 MemoryDB 示例
- 使用 Amazon CLI 的 Amazon MSK 示例
- 使用 Amazon CLI 的 Network Flow Monitor 示例
- 使用 Amazon CLI 的 Network Manager 示例
- 使用 Amazon CLI 的 OpenSearch Service 示例
- 使用Amazon CLI的 Organizations 示例
- Amazon Outposts使用 的 示例Amazon CLI
- Amazon Payment Cryptography使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon Payment Cryptography 数据面板
- 使用 Amazon CLI 的 Amazon Pinpoint 示例
- 使用 Amazon CLI 的 Amazon Polly 示例
- Amazon 价目表使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon 私有 CA 示例
- Amazon Proton使用 的 示例Amazon CLI
- 使用Amazon CLI的 Amazon RDS 示例
- 使用 Amazon CLI 的 Amazon RDS 数据服务示例

- 使用 Amazon CLI 的 Amazon RDS 性能详情示例
- 使用 Amazon CLI 的 Amazon Redshift 示例
- 使用 Amazon CLI 的 Amazon Rekognition 示例
- Amazon RAM使用 的 示例Amazon CLI
- 使用 Amazon CLI 的资源管理器示例
- 使用 Amazon CLI 的资源组示例
- 使用 Amazon CLI 的资源组标记 API
- 使用 Amazon CLI 的 Route 53 示例
- 使用Amazon CLI的 Route 53 域注册示例
- 使用 Amazon CLI 的 Route 53 配置文件示例
- 使用 Amazon CLI 的 Route 53 Resolver 示例
- 使用Amazon CLI的 Amazon S3 示例
- 使用 Amazon CLI 的 Amazon S3 控件示例
- 使用Amazon CLI的 Secrets Manager 示例
- 使用 Amazon CLI 的 Security Hub CSPM 示例
- 使用 Amazon CLI 的 Security Lake 示例
- Amazon Serverless Application Repository使用 的 示例Amazon CLI
- 使用 Amazon CLI 的服务目录示例
- 使用 Amazon CLI 的服务配额示例
- 使用Amazon CLI的 Amazon SES 示例
- 使用 Amazon CLI 的 Shield 示例
- 使用 Amazon CLI 的 Signer 示例
- 使用 Amazon CLI 的 Snowball Edge 示例
- 使用Amazon CLI的 Amazon SNS 示例
- 使用Amazon CLI的 Amazon SQS 示例
- 使用 Amazon CLI 的 Storage Gateway 示例
- Amazon STS使用 的 示例Amazon CLI
- Amazon Web Services 支持使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 Amazon SWF 示例
- 使用 Amazon CLI 的 Systems Manager 示例

- 使用 Amazon CLI 的 Amazon Textract 示例
- 使用 Amazon CLI 的 Amazon Transcribe 示例
- 使用 Amazon CLI 的 Amazon Translate 示例
- 使用 Amazon CLI 的 Trusted Advisor 示例
- 使用 Amazon CLI 的 Verified Permissions 示例
- 使用 Amazon CLI 的 VPC Lattice 示例
- Amazon WAF Classic使用 的 示例Amazon CLI
- Amazon WAF Classic Regional使用 的 示例Amazon CLI
- Amazon WAFV2使用 的 示例Amazon CLI
- 使用 Amazon CLI 的 WorkDocs 示例
- 使用 Amazon CLI 的 Amazon WorkMail 示例
- 使用 Amazon CLI 的 Amazon WorkMail Message Flow 示例
- 使用 Amazon CLI 的 WorkSpaces 示例
- 使用 Amazon CLI 的 X-Ray 示例

使用 Amazon CLI 的 ACM 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 ACM 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-certificate`

以下代码示例演示了如何使用 `add-tags-to-certificate`。

Amazon CLI

向现有 ACM 证书添加标签

以下 add-tags-to-certificate 命令向指定证书添加两个标签。使用空格分隔多个标签：

```
aws acm add-tags-to-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
tags Key=Admin,Value=Alice Key=Purpose,Value=Website
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToCertificate。

delete-certificate

以下代码示例演示了如何使用 delete-certificate。

Amazon CLI

从账户中删除 ACM 证书

以下 delete-certificate 命令删除具有指定 ARN 的证书：

```
aws acm delete-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCertificate。

describe-certificate

以下代码示例演示了如何使用 describe-certificate。

Amazon CLI

检索 ACM 证书中包含的字段

以下 describe-certificate 命令检索具有指定 ARN 的证书的所有字段：

```
aws acm describe-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

输出类似于以下内容：

```
{
  "Certificate": {
    "CertificateArn":
 "arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012",
    "CreatedAt": 1446835267.0,
    "DomainName": "www.example.com",
    "DomainValidationOptions": [
      {
        "DomainName": "www.example.com",
        "ValidationDomain": "www.example.com",
        "ValidationEmails": [
          "hostmaster@example.com",
          "admin@example.com",
          "owner@example.com.whoisprivacyservice.org",
          "tech@example.com.whoisprivacyservice.org",
          "admin@example.com.whoisprivacyservice.org",
          "postmaster@example.com",
          "webmaster@example.com",
          "administrator@example.com"
        ]
      },
      {
        "DomainName": "www.example.net",
        "ValidationDomain": "www.example.net",
        "ValidationEmails": [
          "postmaster@example.net",
          "admin@example.net",
          "owner@example.net.whoisprivacyservice.org",
          "tech@example.net.whoisprivacyservice.org",
          "admin@example.net.whoisprivacyservice.org",
          "hostmaster@example.net",
          "administrator@example.net",
          "webmaster@example.net"
        ]
      }
    ],
    "InUseBy": [],
    "IssuedAt": 1446835815.0,
    "Issuer": "Amazon",
    "KeyAlgorithm": "RSA-2048",
    "NotAfter": 1478433600.0,
    "NotBefore": 1446768000.0,
```

```
    "Serial": "0f:ac:b0:a3:8d:ea:65:52:2d:7d:01:3a:39:36:db:d6",
    "SignatureAlgorithm": "SHA256WITHRSA",
    "Status": "ISSUED",
    "Subject": "CN=www.example.com",
    "SubjectAlternativeNames": [
      "www.example.com",
      "www.example.net"
    ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificate。

export-certificate

以下代码示例演示了如何使用 export-certificate。

Amazon CLI

导出私有 CA 签发的私有证书。

以下 export-certificate 命令将私有证书、证书链和私有密钥导出到显示屏：

```
aws acm export-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
passphrase file://path-to-passphrase-file
```

要将证书、证书链和私钥导出到本地文件，请使用以下命令：

```
aws acm export-certificate --certificate-
arn arn:aws:acm:region:sccount:certificate/12345678-1234-1234-1234-123456789012 --
passphrase file://path-to-passphrase-file > c:\temp\export.txt
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportCertificate。

get-certificate

以下代码示例演示了如何使用 get-certificate。

Amazon CLI

检索 ACM 证书

以下 get-certificate 命令检索指定 ARN 的证书和证书链：

```
aws acm get-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

输出类似于以下内容：

```
{
  "Certificate": "-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",

  "CertificateChain": "-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
"-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
```

```
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
"-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCertificate。

`import-certificate`

以下代码示例演示了如何使用 `import-certificate`。

Amazon CLI

将证书导入 ACM。

以下 `import-certificate` 命令将证书导入 ACM。将文件名替换为您自己的文件名：

```
aws acm import-certificate --certificate file://Certificate.pem --certificate-
chain file://CertificateChain.pem --private-key file://PrivateKey.pem
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportCertificate。

list-certificates

以下代码示例演示了如何使用 list-certificates。

Amazon CLI

列出 Amazon 账户的 ACM 证书

以下 list-certificates 命令列出了您账户中证书的 ARN：

```
aws acm list-certificates
```

上述命令生成类似于以下内容的输出：

```
{
    "CertificateSummaryList": [
        {
            "CertificateArn":
 "arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012",
            "DomainName": "www.example.com"
        },
        {
            "CertificateArn": "arn:aws:acm:region:account:certificate/aaaaaaaa-bbbb-
cccc-dddd-eeeeeeeeeeee",
            "DomainName": "www.example.net"
        }
    ]
}
```

您可以决定每次调用 list-certificates 时要显示多少证书。例如，如果您有四个证书，且希望每次显示的证书不超过两个，请将 max-items 参数设置为 2，如下例所示：

```
aws acm list-certificates --max-items 2
```

将显示两个证书 ARN 和 NextToken 值：

```
"CertificateSummaryList": [
  {
    "CertificateArn": "arn:aws:acm:region:account: \
           certificate/12345678-1234-1234-1234-123456789012",
    "DomainName": "www.example.com"
  },
  {
    "CertificateArn": "arn:aws:acm:region:account: \
            certificate/aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "DomainName": "www.example.net"
  }
  ],
    "NextToken": "9f4d9f69-275a-41fe-b58e-2b837bd9ba48"
```

要在您的账户中显示接下来的两个证书，请在下次调用中设置此 NextToken 值：

```
aws acm list-certificates --max-items 2 --next-token 9f4d9f69-275a-41fe-
b58e-2b837bd9ba48
```

您可以使用 certificate-statuses 参数筛选输出。以下命令显示处于 PENDING_VALIDATION 状态的证书：

```
aws acm list-certificates --certificate-statuses PENDING_VALIDATION
```

您也可以使用 includes 参数筛选输出。以下命令显示根据以下属性筛选的证书。要显示的证书：

```
- Specify that the RSA algorithm and a 2048 bit key are used to generate key pairs.
- Contain a Key Usage extension that specifies that the certificates can be used to
 create digital signatures.
- Contain an Extended Key Usage extension that specifies that the certificates can
 be used for code signing.

aws acm list-certificates --max-items 10 --includes
 extendedKeyUsage=CODE_SIGNING,keyUsage=DIGITAL_SIGNATURE,keyTypes=RSA_2048
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCertificates。

list-tags-for-certificate

以下代码示例演示了如何使用 list-tags-for-certificate。

Amazon CLI

列出应用于 ACM 证书的标签

以下 `list-tags-for-certificate` 命令列出了应用于您账户中证书的标签：

```
aws acm list-tags-for-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

上述命令生成类似于以下内容的输出：

```
{
  "Tags": [
      {
          "Value": "Website",
          "Key": "Purpose"
      },
      {
          "Value": "Alice",
          "Key": "Admin"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForCertificate。

`remove-tags-from-certificate`

以下代码示例演示了如何使用 `remove-tags-from-certificate`。

Amazon CLI

删除 ACM 证书的标签

以下 `remove-tags-from-certificate` 命令删除了指定证书的两个标签。使用空格分隔多个标签：

```
aws acm remove-tags-from-certificate --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
tags Key=Admin,Value=Alice Key=Purpose,Value=Website
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromCertificate。

request-certificate

以下代码示例演示了如何使用 request-certificate。

Amazon CLI

请求新的 ACM 证书

以下 request-certificate 命令使用 DNS 验证为 www.example.com 域请求新证书：

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS
```

您可以输入幂等性令牌，以区分对 request-certificate 的调用：

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS --
idempotency-token 91adc45q
```

您可以输入一个或多个主题备用名称来请求能够保护多个顶级域的证书：

```
aws acm request-certificate --domain-name example.com --validation-method DNS --
idempotency-token 91adc45q --subject-alternative-names www.example.net
```

您可以输入一个备用名称，该名称也可用于访问您的网站：

```
aws acm request-certificate --domain-name example.com --validation-method DNS --
idempotency-token 91adc45q --subject-alternative-names www.example.com
```

您可以使用星号（*）作为通配符为同一域中的多个子域创建证书：

```
aws acm request-certificate --domain-name example.com --validation-method DNS --
idempotency-token 91adc45q --subject-alternative-names *.example.com
```

您也可以输入多个备用名称：

```
aws acm request-certificate --domain-name example.com --validation-method DNS --
subject-alternative-names b.example.com c.example.com d.example.com
```

如果您使用电子邮件进行验证，则可以输入域验证选项来指定将验证电子邮件发送到的域：

```
aws acm request-certificate --domain-name example.com --validation-
method EMAIL --subject-alternative-names www.example.com --domain-validation-
options DomainName=example.com,ValidationDomain=example.com
```

以下命令会在您请求新证书时选择退出证书透明度日志：

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS --
options CertificateTransparencyLoggingPreference=DISABLED --idempotency-token 184627
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestCertificate。

resend-validation-email

以下代码示例演示了如何使用 resend-validation-email。

Amazon CLI

重新发送 ACM 证书请求的验证电子邮件

以下 resend-validation-email 命令指示 Amazon 证书颁发机构向相应的地址发送验证电子邮件：

```
aws acm resend-validation-email --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
domain www.example.com --validation-domain example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResendValidationEmail。

update-certificate-options

以下代码示例演示了如何使用 update-certificate-options。

Amazon CLI

更新证书选项

以下 update-certificate-options 命令选择退出证书透明度日志：

```
aws acm update-certificate-options --certificate-
arn arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
options CertificateTransparencyLoggingPreference=DISABLED
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCertificateOptions。

使用 Amazon CLI 的 API Gateway 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 API Gateway 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-api-key`

以下代码示例演示了如何使用 `create-api-key`。

Amazon CLI

创建为现有 API 和阶段启用的 API 密钥

命令:

```
aws apigateway create-api-key --name 'Dev API Key' --description 'Used for
 development' --enabled --stage-keys restApiId='a1b2c3d4e5',stageName='dev'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApiKey。

`create-authorizer`

以下代码示例演示了如何使用 `create-authorizer`。

Amazon CLI

示例 1：为 API 创建基于令牌的 API Gateway 自定义授权方

以下 `create-authorizer` 示例创建基于令牌的授权方。

```
aws apigateway create-authorizer \
    --rest-api-id 1234123412 \
    --name 'First_Token_Custom_Authorizer' \
    --type TOKEN \
    --authorizer-uri 'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations' \
    --identity-source 'method.request.header.Authorization' \
    --authorizer-result-ttl-in-seconds 300
```

输出：

```
{
    "authType": "custom",
    "name": "First_Token_Custom_Authorizer",
    "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations",
    "authorizerResultTtlInSeconds": 300,
    "identitySource": "method.request.header.Authorization",
    "type": "TOKEN",
    "id": "z40xj0"
}
```

示例 2：为 API 创建基于 Cognito 用户池的 API Gateway 自定义授权方

以下 `create-authorizer` 示例创建基于 Cognito 用户池的 API Gateway 自定义授权方。

```
aws apigateway create-authorizer \
    --rest-api-id 1234123412 \
    --name 'First_Cognito_Custom_Authorizer' \
    --type COGNITO_USER_POOLS \
    --provider-arns 'arn:aws:cognito-idp:us-east-1:123412341234:userpool/us-
east-1_aWcZeQbuD' \
    --identity-source 'method.request.header.Authorization'
```

输出：

```
{
```

```
    "authType": "cognito_user_pools",
    "identitySource": "method.request.header.Authorization",
    "name": "First_Cognito_Custom_Authorizer",
    "providerARNs": [
        "arn:aws:cognito-idp:us-east-1:342398297714:userpool/us-east-1_qWbZzQhzE"
    ],
    "type": "COGNITO_USER_POOLS",
    "id": "5yid1t"
}
```

示例 3：为 API 创建基于请求的 API Gateway 自定义授权方

以下 `create-authorizer` 示例创建基于请求的授权方。

```
aws apigateway create-authorizer \
    --rest-api-id 1234123412 \
    --name 'First_Request_Custom_Authorizer' \
    --type REQUEST \
    --authorizer-uri 'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations' \
    --identity-source 'method.request.header.Authorization,context.accountId' \
    --authorizer-result-ttl-in-seconds 300
```

输出：

```
{
    "id": "z40xj0",
    "name": "First_Request_Custom_Authorizer",
    "type": "REQUEST",
    "authType": "custom",
    "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations",
    "identitySource": "method.request.header.Authorization,context.accountId",
    "authorizerResultTtlInSeconds": 300
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAuthorizer。

create-base-path-mapping

以下代码示例演示了如何使用 `create-base-path-mapping`。

Amazon CLI

创建自定义域名的基础路径映射

命令:

```
aws apigateway create-base-path-mapping --domain-name subdomain.domain.tld --rest-api-id 1234123412 --stage prod --base-path v1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBasePathMapping。

create-deployment

以下代码示例演示了如何使用 create-deployment。

Amazon CLI

将为 API 配置的资源部署到新阶段

命令:

```
aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --stage-description 'Development Stage' --description 'First deployment to the dev stage'
```

将为 API 配置的资源部署到现有阶段

命令:

```
aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --description 'Second deployment to the dev stage'
```

通过 Stage 变量将为 API 配置的资源部署到现有阶段

aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --description 'Third deployment to the dev stage' --variables key='value',otherKey='otherValue'

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeployment。

create-domain-name-access-association

以下代码示例演示了如何使用 create-domain-name-access-association。

Amazon CLI

创建域名访问关联

以下 create-domain-name-access-association 示例在私有自定义域名和 VPC 端点之间创建域名访问关联。

```
aws apigateway create-domain-name-access-association \
    --domain-name-arn arn:aws:apigateway:us-west-2:111122223333:/domainnames/
my.private.domain.tld+abcd1234 \
    --access-association-source vpce-abcd1234efg \
    --access-association-source-type VPCE
```

输出：

```
{
    "domainNameAccessAssociationArn": "arn:aws:apigateway:us-west-2:012345678910:/
domainnameaccessassociations/domainname/my.private.domain.tld/vpcesource/vpce-
abcd1234efg
    "accessAssociationSource": "vpce-abcd1234efg",
    "accessAssociationSourceType": "VPCE",
    "domainNameArn" : "arn:aws:apigateway:us-west-2:111122223333:/domainnames/
private.example.com+abcd1234"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateDomainNameAccessAssociation。

create-domain-name

以下代码示例演示了如何使用 create-domain-name。

Amazon CLI

示例 1：创建公共自定义域名

以下 create-domain-name 示例创建公共自定义域名。

```
aws apigateway create-domain-name \
```

```
    --domain-name 'my.domain.tld' \
    --certificate-name 'my.domain.tld cert'\
    --certificate-arn 'arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-
a305-495d-aefb-27e5e101ff3'
```

输出：

```
{
    "domainName": "my.domain.tld",
    "certificateName": "my.domain.tld cert",
    "certificateArn": "arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-
a305-495d-aefb-27e5e101ff3",
    "certificateUploadDate": "2024-10-08T11:29:49-07:00",
    "distributionDomainName": "abcd1234.cloudfront.net",
    "distributionHostedZoneId": "Z2FDTNDATAQYW2",
    "endpointConfiguration": {
        "types": [
            "EDGE"
        ]
    },
    "domainNameStatus": "AVAILABLE",
    "securityPolicy": "TLS_1_2"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中公共 REST API 的自定义域名。

示例 2：创建私有自定义域名

以下 create-domain-name 示例创建私有自定义域名。

```
aws apigateway create-domain-name \
    --domain-name 'my.private.domain.tld' \
    --certificate-name 'my.domain.tld cert' \
    --certificate-arn 'arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-
a305-495d-aefb-27e5e101ff3' \
    --endpoint-configuration '{"types": ["PRIVATE"]}' \
    --security-policy 'TLS_1_2' \
    --policy file://policy.json
```

的内容policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "execute-api:Invoke",
            "Resource": [
                "execute-api:/*"
            ]
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "execute-api:Invoke",
            "Resource": [
                "execute-api:/*"
            ],
            "Condition" : {
                "StringNotEquals": {
                    "aws:SourceVpce": "vpce-abcd1234efg"
                }
            }
        }
    ]
}
```

输出：

```
{
    "domainName": "my.private.domain.tld",
    "domainNameId": "abcd1234",
    "domainNameArn": "arn:aws:apigateway:us-east-1:012345678910:/domainnames/
my.private.domain.tld+abcd1234",
    "certificateArn": "arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-
a305-495d-aefb-27e5e101ff3",
    "certificateUploadDate": "2024-09-10T10:31:20-07:00",
    "endpointConfiguration": {
        "types": [
            "PRIVATE"
        ]
    },
    "domainNameStatus": "AVAILABLE",
```

```
    "securityPolicy": "TLS_1_2",
    "policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":\"*\",\"Action\":\"execute-api:Invoke\",\"Resource\":
\"arn:aws:execute-api:us-east-1:012345678910:/domainnames/my.private.domain.tld
+abcd1234\"},{\"Effect\":\"Deny\",\"Principal\":\"*\",\"Action\":\"execute-
api:Invoke\",\"Resource\":\"arn:aws:execute-api:us-east-1:012345678910:/domainnames/
my.private.domain.tld+abcd1234\",\"Condition\":{\"StringNotEquals\":{\"aws:SourceVpc
\":\"vpc-1a2b3c4d\"}}}]}"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中公共 REST API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomainName。

`create-model`

以下代码示例演示了如何使用 `create-model`。

Amazon CLI

为 API 创建模型

命令:

```
aws apigateway create-model --rest-api-id 1234123412 --name 'firstModel' --
description 'The First Model' --content-type 'application/json'  --schema
 '{ "$schema": "http://json-schema.org/draft-04/schema#", "title": "firstModel",
 "type": "object", "properties": { "firstProperty" : { "type": "object",
 "properties": { "key": { "type": "string" } } } } }'
```

输出：

```
{
    "contentType": "application/json",
    "description": "The First Model",
    "name": "firstModel",
    "id": "2rzg0l",
    "schema": "{ \"$schema\": \"http://json-schema.org/draft-04/schema#\", \"title
\": \"firstModel\", \"type\": \"object\", \"properties\": { \"firstProperty
\" : { \"type\": \"object\", \"properties\": { \"key\": { \"type\": \"string
\" } } } } }"
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateModel。

create-resource

以下代码示例演示了如何使用 create-resource。

Amazon CLI

在 API 中创建资源

命令:

```
aws apigateway create-resource --rest-api-id 1234123412 --parent-id a1b2c3 --path-part 'new-resource'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResource。

create-rest-api

以下代码示例演示了如何使用 create-rest-api。

Amazon CLI

创建 API

命令:

```
aws apigateway create-rest-api --name 'My First API' --description 'This is my first API'
```

使用现有 API 创建重复的 API

命令:

```
aws apigateway create-rest-api --name 'Copy of My First API' --description 'This is a copy of my first API' --clone-from 1234123412
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRestApi。

create-stage

以下代码示例演示了如何使用 create-stage。

Amazon CLI

在 API 中创建包含现有部署的阶段

命令:

```
aws apigateway create-stage --rest-api-id 1234123412 --stage-name 'dev' --
description 'Development stage' --deployment-id a1b2c3
```

在 API 中创建包含现有部署和自定义阶段变量的阶段

命令:

```
aws apigateway create-stage --rest-api-id 1234123412 --stage-name 'dev'
 --description 'Development stage' --deployment-id a1b2c3 --variables
 key='value',otherKey='otherValue'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStage。

create-usage-plan-key

以下代码示例演示了如何使用 create-usage-plan-key。

Amazon CLI

将现有 API 密钥与使用计划关联

命令:

```
aws apigateway create-usage-plan-key --usage-plan-id a1b2c3 --key-type "API_KEY" --
key-id 4vq3yryqm5
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUsagePlanKey。

create-usage-plan

以下代码示例演示了如何使用 create-usage-plan。

Amazon CLI

创建具有节流功能和配额限制的使用计划，并在月初重置

命令:

```
aws apigateway create-usage-plan --name "New Usage Plan" --description "A new usage plan" --throttle burstLimit=10,rateLimit=5 --quota limit=500,offset=0,period=MONTH
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUsagePlan。

delete-api-key

以下代码示例演示了如何使用 delete-api-key。

Amazon CLI

删除 API 密钥

命令:

```
aws apigateway delete-api-key --api-key 8bklk8bl1k3sB38D9B3l0enyWT8c09B30lkq0blk
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApiKey。

delete-authorizer

以下代码示例演示了如何使用 delete-authorizer。

Amazon CLI

在 API 中删除自定义授权方

命令:

```
aws apigateway delete-authorizer --rest-api-id 1234123412 --authorizer-id 7gkfbo
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAuthorizer。

delete-base-path-mapping

以下代码示例演示了如何使用 delete-base-path-mapping。

Amazon CLI

删除自定义域名的基础路径映射

命令:

```
aws apigateway delete-base-path-mapping --domain-name 'api.domain.tld' --base-path
 'dev'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBasePathMapping。

delete-client-certificate

以下代码示例演示了如何使用 delete-client-certificate。

Amazon CLI

删除客户端证书

命令:

```
aws apigateway delete-client-certificate --client-certificate-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClientCertificate。

delete-deployment

以下代码示例演示了如何使用 delete-deployment。

Amazon CLI

在 API 中删除部署

命令:

```
aws apigateway delete-deployment --rest-api-id 1234123412 --deployment-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeployment。

delete-domain-name-access-association

以下代码示例演示了如何使用 delete-domain-name-access-association。

Amazon CLI

删除域名访问关联

以下 `delete-domain-name-access-association` 示例删除私有自定义域名和 VPC 端点之间的域名访问关联。

```
aws apigateway delete-domain-name-access-association \
    --domain-name-access-association-arn arn:aws:apigateway:us-west-2:012345678910:/domainnameaccessassociations/domainname/my.private.domain.tld/vpcesource/vpce-abcd1234efg
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteDomainNameAccessAssociation。

`delete-domain-name`

以下代码示例演示了如何使用 `delete-domain-name`。

Amazon CLI

删除自定义域名

命令:

```
aws apigateway delete-domain-name --domain-name 'api.domain.tld'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomainName。

`delete-integration-response`

以下代码示例演示了如何使用 `delete-integration-response`。

Amazon CLI

删除 API 中给定资源、方法和状态代码的集成响应

命令:

```
aws apigateway delete-integration-response --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --status-code 200
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIntegrationResponse。

delete-integration

以下代码示例演示了如何使用 delete-integration。

Amazon CLI

删除 API 中给定资源和方法的集成

命令:

```
aws apigateway delete-integration --rest-api-id 1234123412 --resource-id a1b2c3 --
http-method GET
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIntegration。

delete-method-response

以下代码示例演示了如何使用 delete-method-response。

Amazon CLI

删除 API 中给定资源、方法和状态代码的方法响应

命令:

```
aws apigateway delete-method-response --rest-api-id 1234123412 --resource-id a1b2c3
 --http-method GET --status-code 200
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMethodResponse。

delete-method

以下代码示例演示了如何使用 delete-method。

Amazon CLI

删除 API 中给定资源的方法

命令:

```
aws apigateway delete-method --rest-api-id 1234123412 --resource-id a1b2c3 --http-method GET
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMethod。

delete-model

以下代码示例演示了如何使用 delete-model。

Amazon CLI

删除给定 API 中的模型

命令:

```
aws apigateway delete-model --rest-api-id 1234123412 --model-name 'customModel'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteModel。

delete-resource

以下代码示例演示了如何使用 delete-resource。

Amazon CLI

删除 API 中的资源

命令:

```
aws apigateway delete-resource --rest-api-id 1234123412 --resource-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResource。

delete-rest-api

以下代码示例演示了如何使用 delete-rest-api。

Amazon CLI

删除 API

命令:

```
aws apigateway delete-rest-api --rest-api-id 1234123412
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRestApi。

delete-stage

以下代码示例演示了如何使用 delete-stage。

Amazon CLI

删除 API 中的阶段

命令:

```
aws apigateway delete-stage --rest-api-id 1234123412 --stage-name 'dev'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStage。

delete-usage-plan-key

以下代码示例演示了如何使用 delete-usage-plan-key。

Amazon CLI

从使用计划中移除 API 密钥

命令:

```
aws apigateway delete-usage-plan-key --usage-plan-id a1b2c3 --key-
id 1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUsagePlanKey。

delete-usage-plan

以下代码示例演示了如何使用 delete-usage-plan。

Amazon CLI

删除使用计划

命令:

```
aws apigateway delete-usage-plan --usage-plan-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUsagePlan。

flush-stage-authorizers-cache

以下代码示例演示了如何使用 flush-stage-authorizers-cache。

Amazon CLI

刷新阶段的所有授权方缓存条目

命令:

```
aws apigateway flush-stage-authorizers-cache --rest-api-id 1234123412 --stage-
name dev
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 FlushStageAuthorizersCache。

flush-stage-cache

以下代码示例演示了如何使用 flush-stage-cache。

Amazon CLI

刷新 API 阶段的缓存

以下 flush-stage-cache 示例刷新阶段缓存。

```
aws apigateway flush-stage-cache \
    --rest-api-id 1234123412 \
    --stage-name dev
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的刷新 API Gateway 中的 API 阶段缓存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 FlushStageCache。

generate-client-certificate

以下代码示例演示了如何使用 generate-client-certificate。

Amazon CLI

创建客户端侧 SSL 证书

命令:

```
aws apigateway generate-client-certificate --description 'My First Client
 Certificate'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateClientCertificate。

get-account

以下代码示例演示了如何使用 get-account。

Amazon CLI

获取 API Gateway 账户设置

命令:

```
aws apigateway get-account
```

输出：

```
{
    "cloudwatchRoleArn": "arn:aws:iam::123412341234:role/
APIGatewayToCloudWatchLogsRole",
    "throttleSettings": {
        "rateLimit": 500.0,
        "burstLimit": 1000
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccount。

get-api-key

以下代码示例演示了如何使用 get-api-key。

Amazon CLI

获取有关指定 API 密钥的信息

命令:

```
aws apigateway get-api-key --api-key 8bklk8bl1k3sB38D9B3l0enyWT8c09B30lkq0blk
```

输出：

```
{
    "description": "My first key",
    "enabled": true,
    "stageKeys": [
        "a1b2c3d4e5/dev",
        "e5d4c3b2a1/dev"
    ],
    "lastUpdatedDate": 1456184515,
    "createdDate": 1456184452,
    "id": "8bklk8bl1k3sB38D9B3l0enyWT8c09B30lkq0blk",
    "name": "My key"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApiKey。

get-api-keys

以下代码示例演示了如何使用 get-api-keys。

Amazon CLI

获取 API 密钥列表

命令:

```
aws apigateway get-api-keys
```

输出：

```
{
    "items": [
        {
            "description": "My first key",
            "enabled": true,
            "stageKeys": [
                "a1b2c3d4e5/dev",
                "e5d4c3b2a1/dev"
            ],
            "lastUpdatedDate": 1456184515,
            "createdDate": 1456184452,
            "id": "8bklk8bl1k3sB38D9B3l0enyWT8c09B30lkq0blk",
            "name": "My key"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApiKeys。

get-authorizer

以下代码示例演示了如何使用 get-authorizer。

Amazon CLI

获取每个 API 授权方设置的 API Gateway

命令:

```
aws apigateway get-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3
```

输出：

```
{
    "authorizerResultTtlInSeconds": 300,
    "name": "MyAuthorizer",
    "type": "TOKEN",
```

```
    "identitySource": "method.request.header.Authorization",
    "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:authorizer_function/invocations",
    "id": "gfi4n3"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizer。

get-authorizers

以下代码示例演示了如何使用 get-authorizers。

Amazon CLI

获取 REST API 的授权方列表

命令:

```
aws apigateway get-authorizers --rest-api-id 1234123412
```

输出：

```
{
    "items": [
        {
            "name": "MyAuthorizer",
            "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/
functions/arn:aws:lambda:us-west-2:123412341234:function:My_Authorizer_Function/
invocations",
            "authorizerResultTtlInSeconds": 300,
            "identitySource": "method.request.header.Authorization",
            "type": "TOKEN",
            "id": "gfi4n3"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizers。

get-base-path-mapping

以下代码示例演示了如何使用 get-base-path-mapping。

Amazon CLI

获取自定义域名的一个基础路径映射

命令:

```
aws apigateway get-base-path-mapping --domain-name subdomain.domain.tld --base-path v1
```

输出：

```
{
    "basePath": "v1",
    "restApiId": "1234w4321e",
    "stage": "api"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBasePathMapping。

get-base-path-mappings

以下代码示例演示了如何使用 get-base-path-mappings。

Amazon CLI

获取自定义域名的多个基础路径映射

命令:

```
aws apigateway get-base-path-mappings --domain-name subdomain.domain.tld
```

输出：

```
{
    "items": [
        {
            "basePath": "(none)",
            "restApiId": "1234w4321e",
            "stage": "dev"
        },
        {
            "basePath": "v1",
```

```
            "restApiId": "1234w4321e",
            "stage": "api"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBasePathMappings。

get-client-certificate

以下代码示例演示了如何使用 get-client-certificate。

Amazon CLI

获取客户端证书

命令:

```
aws apigateway get-client-certificate --client-certificate-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetClientCertificate。

get-client-certificates

以下代码示例演示了如何使用 get-client-certificates。

Amazon CLI

获取客户端证书列表

命令:

```
aws apigateway get-client-certificates
```

输出：

```
{
    "items": [
        {
            "pemEncodedCertificate": "-----BEGIN CERTIFICATE----- <certificate
 content> -----END CERTIFICATE-----",
```

```
            "clientCertificateId": "a1b2c3",
            "expirationDate": 1483556561,
            "description": "My Client Certificate",
            "createdDate": 1452020561
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetClientCertificates。

get-deployment

以下代码示例演示了如何使用 get-deployment。

Amazon CLI

获取有关部署的信息

命令:

```
aws apigateway get-deployment --rest-api-id 1234123412 --deployment-id ztt4m2
```

输出：

```
{
    "description": "myDeployment",
    "id": "ztt4m2",
    "createdDate": 1455218022
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployment。

get-deployments

以下代码示例演示了如何使用 get-deployments。

Amazon CLI

获取 REST API 的部署列表

命令:

```
aws apigateway get-deployments --rest-api-id 1234123412
```

输出：

```
{
    "items": [
        {
            "createdDate": 1453797217,
            "id": "0a2b4c",
            "description": "Deployed my API for the first time"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployments。

get-domain-name-access-associations

以下代码示例演示了如何使用 get-domain-name-access-associations。

Amazon CLI

示例 1：列出所有域名访问关联

以下 get-domain-name-access-associations 示例列出所有域名访问关联。

```
aws apigateway get-domain-name-access-associations
```

输出：

```
{
    "items": [
        {
        "domainNameAccessAssociationArn": "arn:aws:apigateway:us-
west-2:012345678910:/domainnameaccessassociations/domainname/my.private.domain.tld/
vpcesource/vpce-abcd1234efg
        "accessAssociationSource": "vpce-abcd1234efg",
        "accessAssociationSourceType": "VPCE",
        "domainNameArn" : "arn:aws:apigateway:us-west-2:111122223333:/domainnames/
private.example.com+abcd1234"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

示例 2：列出此 Amazon 账户拥有的所有域名访问关联

以下 `get-domain-name-access-associations` 示例列出当前 Amazon 账户拥有的所有域名访问关联。

```
aws apigateway get-domain-name-access-associations \
    --resource-owner SELF
```

输出：

```
{
    "items": [
        {
        "domainNameAccessAssociationArn": "arn:aws:apigateway:us-
west-2:012345678910:/domainnameaccessassociations/domainname/my.private.domain.tld/
vpcesource/vpce-abcd1234efg
        "accessAssociationSource": "vpce-abcd1234efg",
        "accessAssociationSourceType": "VPCE",
        "domainNameArn" : "arn:aws:apigateway:us-west-2:111122223333:/domainnames/
private.example.com+abcd1234"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDomainNameAccessAssociations。

get-domain-name

以下代码示例演示了如何使用 `get-domain-name`。

Amazon CLI

示例 1：获取有关公共自定义域名的信息

以下 get-domain-name 示例获取有关公共自定义域名的信息。

```
aws apigateway get-domain-name \
    --domain-name api.domain.tld
```

输出：

```
{
    "domainName": "api.domain.tld",
    "distributionDomainName": "d1a2f3a4c5o6d.cloudfront.net",
    "certificateName": "uploadedCertificate",
    "certificateUploadDate": 1462565487
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中公共 REST API 的自定义域名。

示例 2：获取有关私有自定义域名的信息

以下 get-domain-name 示例获取有关私有自定义域名的信息。

```
aws apigateway get-domain-name \
    --domain-name api.private.domain.tld \
    --domain-name-id abcd1234
```

输出：

```
{
    "domainName": "my.private.domain.tld",
    "domainNameId": "abcd1234",
    "domainNameArn": "arn:aws:apigateway:us-east-1:012345678910:/domainnames/
my.private.domain.tld+abcd1234",
    "certificateArn": "arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-
a305-495d-aefb-27e5e101ff3",
    "certificateUploadDate": "2024-09-10T10:31:20-07:00",
    "endpointConfiguration": {
        "types": [
            "PRIVATE"
        ]
    },
    "domainNameStatus": "AVAILABLE",
```

```
    "securityPolicy": "TLS_1_2",
    "policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":\"*\",\"Action\":\"execute-api:Invoke\",\"Resource\":
\"arn:aws:execute-api:us-east-1:012345678910:/domainnames/my.private.domain.tld
+abcd1234\"},{\"Effect\":\"Deny\",\"Principal\":\"*\",\"Action\":\"execute-
api:Invoke\",\"Resource\":\"arn:aws:execute-api:us-east-1:012345678910:/domainnames/
my.private.domain.tld+abcd1234\",\"Condition\":{\"StringNotEquals\":{\"aws:SourceVpc
\":\"vpc-1a2b3c4d\"}}}]}"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中公共 REST API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainName。

get-domain-names

以下代码示例演示了如何使用 get-domain-names。

Amazon CLI

示例 1：获取自定义域名列表

以下 get-domain-names 命令获取域名列表。

```
aws apigateway get-domain-names
```

输出：

```
{
    "items": [
        {
            "distributionDomainName": "d9511k3l09bkd.cloudfront.net",
            "certificateUploadDate": 1452812505,
            "certificateName": "my_custom_domain-certificate",
            "domainName": "subdomain.domain.tld"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

示例 2：获取此 Amazon 账户拥有的自定义域名列表

以下 get-domain-names 命令获取此 Amazon 账户拥有的域名列表。

```
aws apigateway get-domain-names \
    --resource-owner SELF
```

输出：

```
{
    "items": [
        {
            "domainName": "my.domain.tld",
            "domainNameArn": "arn:aws:apigateway:us-east-1::/domainnames/
my.private.domain.tld",
            "certificateUploadDate": "2024-08-15T17:02:55-07:00",
            "regionalDomainName": "d-abcd1234.execute-api.us-east-1.amazonaws.com",
            "regionalHostedZoneId": "Z1UJRXOUMOOFQ8",
            "regionalCertificateArn": "arn:aws:acm:us-
east-1:012345678910:certificate/fb1b9770-a305-495d-aefb-27e5e101ff3",
            "endpointConfiguration": {
                "types": [
                    "REGIONAL"
                ]
            },
            "domainNameStatus": "AVAILABLE",
            "securityPolicy": "TLS_1_2"
        },
        {
            "domainName": "my.private.domain.tld",
            "domainNameId": "abcd1234",
            "domainNameArn": "arn:aws:apigateway:us-east-1:012345678910:/
domainnames/my.private.domain.tld+abcd1234",
            "certificateArn": "arn:aws:acm:us-east-1:012345678910:certificate/
fb1b9770-a305-495d-aefb-27e5e101ff3",
            "certificateUploadDate": "2024-11-26T11:44:40-08:00",
            "endpointConfiguration": {
                "types": [
                    "PRIVATE"
                ]
            },
            "domainNameStatus": "AVAILABLE",
            "securityPolicy": "TLS_1_2"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

示例 3：获取您可以创建域名访问关联的其他 Amazon 账户所拥有的自定义域名列表。

以下 `get-domain-names` 命令获取您有权创建域名访问关联的其他 Amazon 账户所拥有的域名列表。

```
aws apigateway get-domain-names \
    --resource-owner OTHER_ACCOUNTS
```

输出：

```
{
    "items": [
        {
            "domainName": "my.private.domain.tld",
            "domainNameId": "abcd1234",
            "domainNameArn": "arn:aws:apigateway:us-east-1:012345678910:/
domainnames/my.private.domain.tld+abcd1234"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainNames。

`get-export`

以下代码示例演示了如何使用 `get-export`。

Amazon CLI

获取阶段的 JSON Swagger 模板

命令:

```
aws apigateway get-export --rest-api-id a1b2c3d4e5 --stage-name dev --export-
type swagger /path/to/filename.json
```

获取阶段的 JSON Swagger 模板 + API Gateway 扩展

命令:

```
aws apigateway get-export --parameters extensions='integrations' --rest-api-
id a1b2c3d4e5 --stage-name dev --export-type swagger /path/to/filename.json
```

获取阶段的 JSON Swagger 模板 + Postman 扩展

命令:

```
aws apigateway get-export --parameters extensions='postman' --rest-api-id a1b2c3d4e5
 --stage-name dev --export-type swagger /path/to/filename.json
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExport。

get-integration-response

以下代码示例演示了如何使用 get-integration-response。

Amazon CLI

获取在 REST API 资源下定义的 HTTP 方法的集成响应配置

命令:

```
aws apigateway get-integration-response --rest-api-id 1234123412 --resource-
id y9h6rt --http-method GET --status-code 200
```

输出：

```
{
    "statusCode": "200",
    "responseTemplates": {
        "application/json": null
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIntegrationResponse。

get-integration

以下代码示例演示了如何使用 get-integration。

Amazon CLI

获取在 REST API 资源下定义的 HTTP 方法的集成配置

命令:

```
aws apigateway get-integration --rest-api-id 1234123412 --resource-id y9h6rt --http-
method GET
```

输出：

```
{
    "httpMethod": "POST",
    "integrationResponses": {
        "200": {
            "responseTemplates": {
                "application/json": null
            },
            "statusCode": "200"
        }
    },
    "cacheKeyParameters": [],
    "type": "AWS",
    "uri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:My_Function/invocations",
    "cacheNamespace": "y9h6rt"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIntegration。

get-method-response

以下代码示例演示了如何使用 get-method-response。

Amazon CLI

获取在 REST API 资源下定义的 HTTP 方法的方法响应资源配置

命令:

```
aws apigateway get-method-response --rest-api-id 1234123412 --resource-id y9h6rt --
http-method GET --status-code 200
```

输出：

```
{
    "responseModels": {
        "application/json": "Empty"
    },
    "statusCode": "200"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMethodResponse。

get-method

以下代码示例演示了如何使用 get-method。

Amazon CLI

获取在 REST API 资源下定义的 HTTP 方法的方法资源配置

命令:

```
aws apigateway get-method --rest-api-id 1234123412 --resource-id y9h6rt --http-
method GET
```

输出：

```
{
    "apiKeyRequired": false,
    "httpMethod": "GET",
    "methodIntegration": {
        "integrationResponses": {
            "200": {
```

```
                "responseTemplates": {
                    "application/json": null
                },
                "statusCode": "200"
            }
        },
        "cacheKeyParameters": [],
        "uri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:My_Function/invocations",
        "httpMethod": "POST",
        "cacheNamespace": "y9h6rt",
        "type": "AWS"
    },
    "requestParameters": {},
    "methodResponses": {
        "200": {
            "responseModels": {
                "application/json": "Empty"
            },
            "statusCode": "200"
        }
    },
    "authorizationType": "NONE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMethod。

get-model-template

以下代码示例演示了如何使用 `get-model-template`。

Amazon CLI

获取在 REST API 下定义的模型的映射模板

命令:

```
aws apigateway get-model-template --rest-api-id 1234123412 --model-name Empty
```

输出：

```
{
```

```
    "value": "#set($inputRoot = $input.path('$'))\n{ }"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetModelTemplate。

get-model

以下代码示例演示了如何使用 get-model。

Amazon CLI

获取在 REST API 下定义的模型的配置

命令:

```
aws apigateway get-model --rest-api-id 1234123412 --model-name Empty
```

输出：

```
{
    "contentType": "application/json",
    "description": "This is a default empty schema model",
    "name": "Empty",
    "id": "etd5w5",
    "schema": "{\n  \"$schema\": \"http://json-schema.org/draft-04/schema#\",\n
 \"title\" : \"Empty Schema\",\n  \"type\" : \"object\"\n}"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetModel。

get-models

以下代码示例演示了如何使用 get-models。

Amazon CLI

获取 REST API 的模板列表

命令:

```
aws apigateway get-models --rest-api-id 1234123412
```

输出：

```
{
    "items": [
        {
            "description": "This is a default error schema model",
            "schema": "{\n  \"$schema\" : \"http://json-schema.org/draft-04/schema#
\",\n  \"title\" : \"Error Schema\",\n  \"type\" : \"object\",\n  \"properties\" :
 {\n    \"message\" : { \"type\" : \"string\" }\n  }\n}",
            "contentType": "application/json",
            "id": "7tpbze",
            "name": "Error"
        },
        {
            "description": "This is a default empty schema model",
            "schema": "{\n  \"$schema\": \"http://json-schema.org/draft-04/schema#
\",\n  \"title\" : \"Empty Schema\",\n  \"type\" : \"object\"\n}",
            "contentType": "application/json",
            "id": "etd5w5",
            "name": "Empty"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetModels。

get-resource

以下代码示例演示了如何使用 get-resource。

Amazon CLI

获取有关资源的信息

命令:

```
aws apigateway get-resource --rest-api-id 1234123412 --resource-id zwo0y3
```

输出：

```
{
```

```
    "path": "/path",
    "pathPart": "path",
    "id": "zwo0y3",
    "parentId": "uyokt6ij2g"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResource。

get-resources

以下代码示例演示了如何使用 get-resources。

Amazon CLI

获取 REST API 的资源列表

命令:

```
aws apigateway get-resources --rest-api-id 1234123412
```

输出：

```
{
    "items": [
        {
            "path": "/resource/subresource",
            "resourceMethods": {
                "POST": {}
            },
            "id": "024ace",
            "pathPart": "subresource",
            "parentId": "ai5b02"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResources。

get-rest-api

以下代码示例演示了如何使用 get-rest-api。

Amazon CLI

获取有关 API 的信息

命令:

```
aws apigateway get-rest-api --rest-api-id 1234123412
```

输出：

```
{
    "name": "myAPI",
    "id": "o1y243m4f5",
    "createdDate": 1453416433
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRestApi。

get-rest-apis

以下代码示例演示了如何使用 get-rest-apis。

Amazon CLI

获取 REST API 列表

命令:

```
aws apigateway get-rest-apis
```

输出：

```
{
    "items": [
        {
            "createdDate": 1438884790,
            "id": "12s44z21rb",
            "name": "My First API"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRestApis。

get-sdk

以下代码示例演示了如何使用 get-sdk。

Amazon CLI

获取适用于 REST API 阶段的 Android SDK

命令:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-type android
 --parameters
 groupId='com.mycompany',invokerPackage='com.mycompany.clientsdk',artifactId='Mycompany-
client',artifactVersion='1.0.0' /path/to/android_sdk.zip
```

输出：

```
{
    "contentType": "application/octet-stream",
    "contentDisposition": "attachment; filename=\"android_2016-02-22_23-52Z.zip\""
}
```

获取适用于 REST API 阶段的 IOS SDK

命令:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-
type objectivec --parameters classPrefix='myprefix' /path/to/iOS_sdk.zip
```

输出：

```
{
    "contentType": "application/octet-stream",
    "contentDisposition": "attachment; filename=\"objectivec_2016-02-22_23-52Z.zip
\""
}
```

获取适用于 REST API 阶段的 Javascript SDK

命令:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-
type javascript /path/to/javascript_sdk.zip
```

输出：

```
{
    "contentType": "application/octet-stream",
    "contentDisposition": "attachment; filename=\"javascript_2016-02-22_23-52Z.zip
\""
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSdk。

get-stage

以下代码示例演示了如何使用 get-stage。

Amazon CLI

获取有关 API 阶段的信息

命令:

```
aws apigateway get-stage --rest-api-id 1234123412 --stage-name dev
```

输出：

```
{
    "stageName": "dev",
    "cacheClusterSize": "0.5",
    "cacheClusterEnabled": false,
    "cacheClusterStatus": "NOT_AVAILABLE",
    "deploymentId": "rbh1fj",
    "lastUpdatedDate": 1466802961,
    "createdDate": 1460682074,
    "methodSettings": {
```

```
        "*/*": {
            "cacheTtlInSeconds": 300,
            "loggingLevel": "INFO",
            "dataTraceEnabled": false,
            "metricsEnabled": true,
            "unauthorizedCacheControlHeaderStrategy":
 "SUCCEED_WITH_RESPONSE_HEADER",
            "throttlingRateLimit": 500.0,
            "cacheDataEncrypted": false,
            "cachingEnabled": false,
            "throttlingBurstLimit": 1000,
            "requireAuthorizationForCacheControl": true
        },
        "~1resource/GET": {
            "cacheTtlInSeconds": 300,
            "loggingLevel": "INFO",
            "dataTraceEnabled": false,
            "metricsEnabled": true,
            "unauthorizedCacheControlHeaderStrategy":
 "SUCCEED_WITH_RESPONSE_HEADER",
            "throttlingRateLimit": 500.0,
            "cacheDataEncrypted": false,
            "cachingEnabled": false,
            "throttlingBurstLimit": 1000,
            "requireAuthorizationForCacheControl": true
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStage。

get-stages

以下代码示例演示了如何使用 get-stages。

Amazon CLI

获取 REST API 的阶段列表

命令:

```
aws apigateway get-stages --rest-api-id 1234123412
```

输出：

```
{
    "item": [
        {
            "stageName": "dev",
            "cacheClusterSize": "0.5",
            "cacheClusterEnabled": true,
            "cacheClusterStatus": "AVAILABLE",
            "deploymentId": "123h64",
            "lastUpdatedDate": 1456185138,
            "createdDate": 1453589092,
            "methodSettings": {
                "~1resource~1subresource/POST": {
                    "cacheTtlInSeconds": 300,
                    "loggingLevel": "INFO",
                    "dataTraceEnabled": true,
                    "metricsEnabled": true,
                    "throttlingRateLimit": 500.0,
                    "cacheDataEncrypted": false,
                    "cachingEnabled": false,
                    "throttlingBurstLimit": 1000
                }
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStages。

get-usage-plan-key

以下代码示例演示了如何使用 get-usage-plan-key。

Amazon CLI

获取与使用计划关联的 API 密钥的详细信息

命令:

```
aws apigateway get-usage-plan-key --usage-plan-id a1b2c3 --key-
id 1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUsagePlanKey。

get-usage-plan-keys

以下代码示例演示了如何使用 get-usage-plan-keys。

Amazon CLI

获取与使用计划关联的 API 密钥列表

命令:

```
aws apigateway get-usage-plan-keys --usage-plan-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUsagePlanKeys。

get-usage-plan

以下代码示例演示了如何使用 get-usage-plan。

Amazon CLI

获取使用计划的详细信息

命令:

```
aws apigateway get-usage-plan --usage-plan-id a1b2c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUsagePlan。

get-usage-plans

以下代码示例演示了如何使用 get-usage-plans。

Amazon CLI

获取所有使用计划的详细信息

命令:

```
aws apigateway get-usage-plans
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUsagePlans。

get-usage

以下代码示例演示了如何使用 get-usage。

Amazon CLI

获取使用计划的使用详细信息

命令:

```
aws apigateway get-usage --usage-plan-id a1b2c3 --start-date "2016-08-16" --end-
date "2016-08-17"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUsage。

import-rest-api

以下代码示例演示了如何使用 import-rest-api。

Amazon CLI

导入 Swagger 模板并创建 API

命令:

```
aws apigateway import-rest-api --body 'file:///path/to/API_Swagger_template.json'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportRestApi。

put-integration-response

以下代码示例演示了如何使用 put-integration-response。

Amazon CLI

使用定义的映射模板创建集成响应作为默认响应

命令:

```
aws apigateway put-integration-response --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --status-code 200 --selection-pattern "" --response-
templates '{"application/json": "{\"json\": \"template\"}"}'
```

使用正则表达式为 400 的静态定义的标头值创建集成响应

命令:

```
aws apigateway put-integration-response --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --status-code 400 --selection-pattern 400 --response-
parameters '{"method.response.header.custom-header": "'"'"'custom-value'"'"'"}'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutIntegrationResponse。

put-integration

以下代码示例演示了如何使用 put-integration。

Amazon CLI

创建 MOCK 集成请求

命令:

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-id a1b2c3 --http-
method GET --type MOCK --request-templates '{ "application/json": "{\"statusCode\":
 200}" }'
```

创建 HTTP 集成请求

命令:

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-id a1b2c3 --http-
method GET --type HTTP --integration-http-method GET --uri 'https://domain.tld/path'
```

使用 Lambda 函数端点创建 Amazon 集成请求

命令:

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --type AWS --integration-http-method POST --uri
```

```
 'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-
west-2:123412341234:function:function_name/invocations'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutIntegration。

put-method-response

以下代码示例演示了如何使用 put-method-response。

Amazon CLI

基于指定状态代码，通过自定义方法响应标头创建方法响应

命令:

```
aws apigateway put-method-response --rest-api-id 1234123412 --
resource-id a1b2c3 --http-method GET --status-code 400 --response-
parameters "method.response.header.custom-header=false"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMethodResponse。

put-method

以下代码示例演示了如何使用 put-method。

Amazon CLI

在没有授权、没有 API 密钥和自定义方法请求标头的情况下为 API 中的资源创建方法

命令:

```
aws apigateway put-method --rest-api-id 1234123412 --resource-id a1b2c3 --
http-method PUT --authorization-type "NONE" --no-api-key-required --request-
parameters "method.request.header.custom-header=false"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMethod。

put-rest-api

以下代码示例演示了如何使用 put-rest-api。

Amazon CLI

使用 Swagger 模板覆盖现有 API

命令:

```
aws apigateway put-rest-api --rest-api-id 1234123412 --mode overwrite --body
 'fileb:///path/to/API_Swagger_template.json'
```

将 Swagger 模板合并到现有 API 中

命令:

```
aws apigateway put-rest-api --rest-api-id 1234123412 --mode merge --body 'fileb:///
path/to/API_Swagger_template.json'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRestApi。

reject-domain-name-access-association

以下代码示例演示了如何使用 reject-domain-name-access-association。

Amazon CLI

拒绝域名访问关联

以下 reject-domain-name-access-association 示例拒绝私有自定义域名和 VPC 端点之间的域名访问关联。

```
aws apigateway reject-domain-name-access-association \
    --domain-name-access-association-arn arn:aws:apigateway:us-west-2:012345678910:/
domainnameaccessassociations/domainname/my.private.domain.tld/vpcesource/vpce-
abcd1234efg \
    --domain-name-arn arn:aws:apigateway:us-east-1:012345678910:/domainnames/
my.private.domain.tld+abcd1234
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API Gateway 中私有 API 的自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RejectDomainNameAccessAssociation。

test-invoke-authorizer

以下代码示例演示了如何使用 test-invoke-authorizer。

Amazon CLI

测试包含所需标头和值的自定义授权方的调用请求

命令:

```
aws apigateway test-invoke-authorizer --rest-api-id 1234123412 --authorizer-
id 5yid1t --headers Authorization='Value'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestInvokeAuthorizer。

test-invoke-method

以下代码示例演示了如何使用 test-invoke-method。

Amazon CLI

通过发出 GET 请求，在 API 中测试调用根资源

命令:

```
aws apigateway test-invoke-method --rest-api-id 1234123412 --resource-id avl5sg8fw8
 --http-method GET --path-with-query-string '/'
```

通过发出指定路径参数值的 GET 请求，在 API 中测试调用子资源

命令:

```
aws apigateway test-invoke-method --rest-api-id 1234123412 --resource-id 3gapai --
http-method GET --path-with-query-string '/pets/1'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestInvokeMethod。

update-account

以下代码示例演示了如何使用 update-account。

Amazon CLI

更改用于登录 CloudWatch Logs 的 IAM 角色 ARN

命令:

```
aws apigateway update-account --patch-operations op='replace',path='/
cloudwatchRoleArn',value='arn:aws:iam::123412341234:role/APIGatewayToCloudWatchLogs'
```

输出：

```
{
    "cloudwatchRoleArn": "arn:aws:iam::123412341234:role/
APIGatewayToCloudWatchLogs",
    "throttleSettings": {
        "rateLimit": 1000.0,
        "burstLimit": 2000
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccount。

update-api-key

以下代码示例演示了如何使用 update-api-key。

Amazon CLI

更改 API 密钥的名称

命令:

```
aws apigateway update-api-key --api-key sNvjQDMReA1eEQPNAW8r37XsU2rDD7fc7m2SiMnu --
patch-operations op='replace',path='/name',value='newName'
```

输出：

```
{
```

```
    "description": "currentDescription",
    "enabled": true,
    "stageKeys": [
        "41t2j324r5/dev"
    ],
    "lastUpdatedDate": 1470086052,
    "createdDate": 1445460347,
    "id": "sNvjQDMReA1vEQPNzW8r3dXsU2rrD7fcjm2SiMnu",
    "name": "newName"
}
```

禁用 API 密钥

命令:

```
aws apigateway update-api-key --api-key sNvjQDMReA1eEQPNAW8r37XsU2rDD7fc7m2SiMnu --
patch-operations op='replace',path='/enabled',value='false'
```

输出：

```
{
    "description": "currentDescription",
    "enabled": false,
    "stageKeys": [
        "41t2j324r5/dev"
    ],
    "lastUpdatedDate": 1470086052,
    "createdDate": 1445460347,
    "id": "sNvjQDMReA1vEQPNzW8r3dXsU2rrD7fcjm2SiMnu",
    "name": "newName"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApiKey。

update-authorizer

以下代码示例演示了如何使用 update-authorizer。

Amazon CLI

更改自定义授权方的名称

命令:

```
aws apigateway update-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3 --
patch-operations op='replace',path='/name',value='testAuthorizer'
```

输出：

```
{
    "authType": "custom",
    "name": "testAuthorizer",
    "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthorizer/invocations",
    "authorizerResultTtlInSeconds": 300,
    "identitySource": "method.request.header.Authorization",
    "type": "TOKEN",
    "id": "gfi4n3"
}
```

更改自定义授权方调用的 Lambda 函数

命令:

```
aws apigateway update-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3 --
patch-operations op='replace',path='/authorizerUri',value='arn:aws:apigateway:us-
west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-
west-2:123412341234:function:newAuthorizer/invocations'
```

输出：

```
{
    "authType": "custom",
    "name": "testAuthorizer",
    "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:newAuthorizer/invocations",
    "authorizerResultTtlInSeconds": 300,
    "identitySource": "method.request.header.Authorization",
    "type": "TOKEN",
    "id": "gfi4n3"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAuthorizer。

update-base-path-mapping

以下代码示例演示了如何使用 update-base-path-mapping。

Amazon CLI

更改自定义域名的基础路径

命令:

```
aws apigateway update-base-path-mapping --domain-name api.domain.tld --base-
path prod --patch-operations op='replace',path='/basePath',value='v1'
```

输出：

```
{
    "basePath": "v1",
    "restApiId": "1234123412",
    "stage": "api"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBasePathMapping。

update-client-certificate

以下代码示例演示了如何使用 update-client-certificate。

Amazon CLI

更新客户端证书的说明

命令:

```
aws apigateway update-client-certificate --client-certificate-id a1b2c3 --patch-
operations op='replace',path='/description',value='My new description'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateClientCertificate。

update-deployment

以下代码示例演示了如何使用 update-deployment。

Amazon CLI

更改部署的说明

命令:

```
aws apigateway update-deployment --rest-api-id 1234123412 --deployment-id ztt4m2 --
patch-operations op='replace',path='/description',value='newDescription'
```

输出：

```
{
    "description": "newDescription",
    "id": "ztt4m2",
    "createdDate": 1455218022
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeployment。

update-domain-name

以下代码示例演示了如何使用 update-domain-name。

Amazon CLI

更改自定义域名的证书名称

以下 update-domain-name 示例更改自定义域的证书名称。

```
aws apigateway update-domain-name \
    --domain-name api.domain.tld \
    --patch-operations op='replace',path='/certificateArn',value='arn:aws:acm:us-
west-2:111122223333:certificate/CERTEXAMPLE123EXAMPLE'
```

输出：

```
{
    "domainName": "api.domain.tld",
    "distributionDomainName": "d123456789012.cloudfront.net",
    "certificateArn": "arn:aws:acm:us-west-2:111122223333:certificate/
CERTEXAMPLE123EXAMPLE",
```

```
    "certificateUploadDate": 1462565487
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中为 API 设置自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainName。

update-integration-response

以下代码示例演示了如何使用 update-integration-response。

Amazon CLI

将集成响应标头更改为具有“*”的静态映射

命令:

```
aws apigateway update-integration-response --rest-api-id 1234123412 --
resource-id 3gapai --http-method GET --status-code 200 --patch-operations
 op='replace',path='/responseParameters/method.response.header.Access-Control-Allow-
Origin',value='"'"'*'"'"'
```

输出：

```
{
    "statusCode": "200",
    "responseParameters": {
        "method.response.header.Access-Control-Allow-Origin": "'*'"
    }
}
```

移除集成响应标头

命令:

```
aws apigateway update-integration-response --rest-api-id 1234123412 --resource-
id 3gapai --http-method GET --status-code 200 --patch-operations op='remove',path='/
responseParameters/method.response.header.Access-Control-Allow-Origin'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIntegrationResponse。

update-integration

以下代码示例演示了如何使用 update-integration。

Amazon CLI

添加配置了输入直通的“Content-Type: application/json”映射模板

命令:

```
aws apigateway update-integration \
    --rest-api-id a1b2c3d4e5 \
    --resource-id a1b2c3 \
    --http-method POST \
    --patch-operations "op='add',path='/requestTemplates/application~1json'"
```

更新（替换）使用自定义模板配置的“Content-Type: application/json”映射模板

命令:

```
aws apigateway update-integration \
    --rest-api-id a1b2c3d4e5 \
    --resource-id a1b2c3 \
    --http-method POST \
    --patch-operations "op='replace',path='/requestTemplates/
application~1json',value='{"example": "json"}'"
```

使用输入直通更新（替换）与“Content-Type: application/json”关联的自定义模板

命令:

```
aws apigateway update-integration \
    --rest-api-id a1b2c3d4e5 \
    --resource-id a1b2c3 \
    --http-method POST \
    --patch-operations "op='replace',path='requestTemplates/application~1json'"
```

移除“Content-Type: application/json”映射模板

命令:

```
aws apigateway update-integration \
```

```
    --rest-api-id a1b2c3d4e5 \
    --resource-id a1b2c3 \
    --http-method POST \
    --patch-operations "op='remove',path='/requestTemplates/application~1json'"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIntegration。

update-method-response

以下代码示例演示了如何使用 update-method-response。

Amazon CLI

为方法中的 200 响应创建新的方法响应标头，并将其定义为非必填（默认值）

命令:

```
aws apigateway update-method-response --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --status-code 200 --patch-operations op="add",path="/
responseParameters/method.response.header.custom-header",value="false"
```

在方法中删除 200 响应的响应模型

命令:

```
aws apigateway update-method-response --rest-api-id 1234123412 --resource-id a1b2c3
 --http-method GET --status-code 200 --patch-operations op="remove",path="/
responseModels/application~1json"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMethodResponse。

update-method

以下代码示例演示了如何使用 update-method。

Amazon CLI

示例 1：将方法修改为需要 API 密钥

以下 update-method 示例将方法修改为需要 API 密钥。

```
aws apigateway update-method \
    --rest-api-id 1234123412 \
    --resource-id a1b2c3 \
    --http-method GET \
    --patch-operations op="replace",path="/apiKeyRequired",value="true"
```

输出：

```
{
    "httpMethod": "GET",
    "authorizationType": "NONE",
    "apiKeyRequired": true,
    "methodResponses": {
        "200": {
            "statusCode": "200",
            "responseModels": {}
        }
    },
    "methodIntegration": {
        "type": "AWS",
        "httpMethod": "POST",
        "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
        "passthroughBehavior": "WHEN_NO_MATCH",
        "contentHandling": "CONVERT_TO_TEXT",
        "timeoutInMillis": 29000,
        "cacheNamespace": "h7i8j9",
        "cacheKeyParameters": [],
        "integrationResponses": {
            "200": {
                "statusCode": "200",
                "responseTemplates": {}
            }
        }
    }
}
```

示例 2：将方法修改为需要 IAM 授权

以下 update-method 示例将方法修改为需要 IAM 授权。

```
aws apigateway update-method \
```

```
    --rest-api-id 1234123412 \
    --resource-id a1b2c3 \
    --http-method GET \
    --patch-operations op="replace",path="/authorizationType",value="AWS_IAM"
```

输出：

```
 {
    "httpMethod": "GET",
    "authorizationType": "AWS_IAM",
    "apiKeyRequired": false,
    "methodResponses": {
        "200": {
            "statusCode": "200",
            "responseModels": {}
        }
    },
    "methodIntegration": {
        "type": "AWS",
        "httpMethod": "POST",
        "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
        "passthroughBehavior": "WHEN_NO_MATCH",
        "contentHandling": "CONVERT_TO_TEXT",
        "timeoutInMillis": 29000,
        "cacheNamespace": "h7i8j9",
        "cacheKeyParameters": [],
        "integrationResponses": {
            "200": {
                "statusCode": "200",
                "responseTemplates": {}
            }
        }
    }
}
```

示例 3：将方法修改为需要 Lambda 授权

以下 update-method 示例将方法修改为需要 Lambda 授权。

```
aws apigateway update-method --rest-api-id 1234123412 \
    --resource-id a1b2c3 \
    --http-method GET \
```

```
    --patch-operations op="replace",path="/authorizationType",value="CUSTOM"
 op="replace",path="/authorizerId",value="e4f5g6"
```

输出：

```
{
    "httpMethod": "GET",
    "authorizationType": "CUSTOM",
    "authorizerId" : "e4f5g6",
    "apiKeyRequired": false,
    "methodResponses": {
        "200": {
            "statusCode": "200",
            "responseModels": {}
        }
    },
    "methodIntegration": {
        "type": "AWS",
        "httpMethod": "POST",
        "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
        "passthroughBehavior": "WHEN_NO_MATCH",
        "contentHandling": "CONVERT_TO_TEXT",
        "timeoutInMillis": 29000,
        "cacheNamespace": "h7i8j9",
        "cacheKeyParameters": [],
        "integrationResponses": {
            "200": {
                "statusCode": "200",
                "responseTemplates": {}
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 API Gateway CLI 和 REST API 创建、配置和测试使用计划和在 API Gateway 中控制和管理 REST API 的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMethod。

update-model

以下代码示例演示了如何使用 update-model。

Amazon CLI

在 API 中更改模型的说明

命令:

```
aws apigateway update-model --rest-api-id 1234123412 --model-name 'Empty' --patch-operations op=replace,path=/description,value='New Description'
```

在 API 中更改模型的架构

命令:

```
aws apigateway update-model --rest-api-id 1234123412 --model-name 'Empty' --patch-operations op=replace,path=/schema,value='"{ \"$schema\": \"http://json-schema.org/draft-04/schema#\", \"title\" : \"Empty Schema\", \"type\" : \"object\" }"'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateModel。

update-resource

以下代码示例演示了如何使用 update-resource。

Amazon CLI

在 API 中移动资源并将其置于不同的父资源下

命令:

```
aws apigateway update-resource --rest-api-id 1234123412 --resource-id 1a2b3c --patch-operations op=replace,path=/parentId,value='3c2b1a'
```

输出：

```
{
    "path": "/resource",
    "pathPart": "resource",
    "id": "1a2b3c",
    "parentId": "3c2b1a"
}
```

在 API 中重命名资源（pathPart

命令:

```
aws apigateway update-resource --rest-api-id 1234123412 --resource-id 1a2b3c --
patch-operations op=replace,path=/pathPart,value=newresourcename
```

输出：

```
{
    "path": "/newresourcename",
    "pathPart": "newresourcename",
    "id": "1a2b3c",
    "parentId": "3c2b1a"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResource。

update-rest-api

以下代码示例演示了如何使用 update-rest-api。

Amazon CLI

更改 API 的名称

命令:

```
aws apigateway update-rest-api --rest-api-id 1234123412 --patch-operations
 op=replace,path=/name,value='New Name'
```

更改 API 的说明

命令:

```
aws apigateway update-rest-api --rest-api-id 1234123412 --patch-operations
 op=replace,path=/description,value='New Description'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRestApi。

update-stage

以下代码示例演示了如何使用 update-stage。

Amazon CLI

示例 1：覆盖资源和方法的阶段设置

以下 update-stage 示例覆盖阶段设置并关闭指定资源和方法的完整请求/响应日志记录。

```
aws apigateway update-stage \
    --rest-api-id 1234123412 \
    --stage-name 'dev' \
    --patch-operations op=replace,path=/~1resourceName/GET/logging/
dataTrace,value=false
```

输出：

```
{
    "deploymentId": "5ubd17",
    "stageName": "dev",
    "cacheClusterEnabled": false,
    "cacheClusterStatus": "NOT_AVAILABLE",
    "methodSettings": {
        "~1resourceName/GET": {
            "metricsEnabled": false,
            "dataTraceEnabled": false,
            "throttlingBurstLimit": 5000,
            "throttlingRateLimit": 10000.0,
            "cachingEnabled": false,
            "cacheTtlInSeconds": 300,
            "cacheDataEncrypted": false,
            "requireAuthorizationForCacheControl": true,
            "unauthorizedCacheControlHeaderStrategy": "SUCCEED_WITH_RESPONSE_HEADER"
        }
    },
    "tracingEnabled": false,
    "createdDate": "2022-07-18T10:11:18-07:00",
    "lastUpdatedDate": "2022-07-18T10:19:04-07:00"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 REST API 设置阶段。

示例 2：更新 API 阶段的所有资源和方法的阶段设置

以下 update-stage 示例为 API 阶段的所有资源和方法开启完整请求/响应日志记录。

```
aws apigateway update-stage \
    --rest-api-id 1234123412 \
    --stage-name 'dev' \
    --patch-operations 'op=replace,path=/*/*/logging/dataTrace,value=true'
```

输出：

```
{
    "deploymentId": "5ubd17",
    "stageName": "dev",
    "cacheClusterEnabled": false,
    "cacheClusterStatus": "NOT_AVAILABLE",
    "methodSettings": {
        "*/*": {
            "metricsEnabled": false,
            "dataTraceEnabled": true,
            "throttlingBurstLimit": 5000,
            "throttlingRateLimit": 10000.0,
            "cachingEnabled": false,
            "cacheTtlInSeconds": 300,
            "cacheDataEncrypted": false,
            "requireAuthorizationForCacheControl": true,
            "unauthorizedCacheControlHeaderStrategy": "SUCCEED_WITH_RESPONSE_HEADER"
        }
    },
    "tracingEnabled": false,
    "createdDate": "2022-07-18T10:11:18-07:00",
    "lastUpdatedDate": "2022-07-18T10:31:04-07:00"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 REST API 设置阶段。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStage。

update-usage-plan

以下代码示例演示了如何使用 update-usage-plan。

Amazon CLI

更改使用计划中定义的期限

命令:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
 op="replace",path="/quota/period",value="MONTH"
```

更改使用计划中定义的配额限制

命令:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
 op="replace",path="/quota/limit",value="500"
```

更改使用计划中定义的节流速率限制

命令:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
 op="replace",path="/throttle/rateLimit",value="10"
```

更改使用计划中定义的节流爆发限制

命令:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
 op="replace",path="/throttle/burstLimit",value="20"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUsagePlan。

update-usage

以下代码示例演示了如何使用 update-usage。

Amazon CLI

临时修改使用计划中定义的当前时间段内 API 密钥的配额

命令:

```
aws apigateway update-usage --usage-plan-id a1b2c3 --key-
id 1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu --patch-operations op="replace",path="/
remaining",value="50"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUsage。

update-vpc-link

以下代码示例演示了如何使用 update-vpc-link。

Amazon CLI

示例 1：更新现有 VPC 链接名称

以下 update-vpc-link 示例更新指定 VPC 链接的名称。

```
aws apigateway update-vpc-link  \
    --vpc-link-id ab3de6 \
    --patch-operations op=replace,path=/name,value=my-vpc-link
```

输出：

```
{
    "id": "ab3de6",
    "name": "my-vpc-link",
    "targetArns": [
        "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/net/my-
lb/12a456s89aaa12345"
    ],
    "status": "AVAILABLE",
    "statusMessage": "Your vpc link is ready for use",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CLI 命令参考》中的更新现有 VPC 链接。

示例 2：更新现有 VPC 链接名称和描述

以下 update-vpc-link 示例更新指定 VPC 链接的名称。

```
aws apigateway update-vpc-link  \
    --vpc-link-id ab3de6 \
    --patch-operations op=replace,path=/name,value=my-vpc-link op=replace,path=/
description,value="My custom description"
```

输出：

```
{
    "id": "ab3de6",
    "name": "my-vpc-link",
    "description": "My custom description",
    "targetArns": [
        "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/net/my-
lb/12a456s89aaa12345"
    ],
    "status": "AVAILABLE",
    "statusMessage": "Your vpc link is ready for use",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CLI 命令参考》中的更新现有 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVpcLink。

使用 Amazon CLI 的 API Gateway HTTP 和 WebSocket API 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 API Gateway HTTP 和 WebSocket API 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-api-mapping`

以下代码示例演示了如何使用 `create-api-mapping`。

Amazon CLI

为 API 创建 API 映射

以下 create-api-mapping 示例将 API 的 test 阶段映射到 regional.example.com 自定义域名的 /myApi 路径。

```
aws apigatewayv2 create-api-mapping \
    --domain-name regional.example.com \
    --api-mapping-key myApi \
    --api-id a1b2c3d4 \
    --stage test
```

输出：

```
{
    "ApiId": "a1b2c3d4",
    "ApiMappingId": "0qzs2sy7bh",
    "ApiMappingKey": "myApi"
    "Stage": "test"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApiMapping。

create-api

以下代码示例演示了如何使用 create-api。

Amazon CLI

创建 HTTP API

以下 create-api 示例使用“快速创建”功能创建一个 HTTP API。您可以使用“快速创建”功能，创建具有 Amazon Lambda 或 HTTP 集成、默认“捕获全部”路由和默认阶段（配置为自动部署更改）的 API。以下命令使用“快速创建”功能创建与 Lambda 函数集成的 HTTP API。

```
aws apigatewayv2 create-api \
    --name my-http-api \
    --protocol-type HTTP \
    --target arn:aws:lambda:us-west-2:123456789012:function:my-lambda-function
```

输出：

```
{
    "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
    "ApiId": "a1b2c3d4",
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "CreatedDate": "2020-04-08T19:05:45+00:00",
    "Name": "my-http-api",
    "ProtocolType": "HTTP",
    "RouteSelectionExpression": "$request.method $request.path"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中开发 HTTP API。

创建 WebSocket API

以下 `create-api` 示例创建一个具有指定名称的 WebSocket API。

```
aws apigatewayv2 create-api \
    --name "myWebSocketApi" \
    --protocol-type WEBSOCKET \
    --route-selection-expression '$request.body.action'
```

输出：

```
{
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "Name": "myWebSocketApi",
    "CreatedDate": "2018-11-15T06:23:51Z",
    "ProtocolType": "WEBSOCKET",
    "RouteSelectionExpression": "'$request.body.action'",
    "ApiId": "aabbccddee"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中创建 WebSocket API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApi。

`create-authorizer`

以下代码示例演示了如何使用 `create-authorizer`。

Amazon CLI

为 HTTP API 创建 JWT 授权方

以下 `create-authorizer` 示例创建使用 Amazon Cognito 作为身份提供商的 JWT 授权方。

```
aws apigatewayv2 create-authorizer \
    --name my-jwt-authorizer \
    --api-id a1b2c3d4 \
    --authorizer-type JWT \
    --identity-source '$request.header.Authorization' \
    --jwt-configuration Audience=123456abc,Issuer=https://cognito-idp.us-
west-2.amazonaws.com/us-west-2_abc123
```

输出：

```
{
    "AuthorizerId": "a1b2c3",
    "AuthorizerType": "JWT",
    "IdentitySource": [
        "$request.header.Authorization"
    ],
    "JwtConfiguration": {
        "Audience": [
            "123456abc"
        ],
        "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
    },
    "Name": "my-jwt-authorizer"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAuthorizer。

`create-deployment`

以下代码示例演示了如何使用 `create-deployment`。

Amazon CLI

为 API 创建部署

以下 create-deployment 示例创建 API 的部署并将该部署与 API 的 dev 阶段关联。

```
aws apigatewayv2 create-deployment \
    --api-id a1b2c3d4 \
    --stage-name dev
```

输出：

```
{
    "AutoDeployed": false,
    "CreatedDate": "2020-04-06T23:38:08Z",
    "DeploymentId": "53lz9l",
    "DeploymentStatus": "DEPLOYED"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API 部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeployment。

create-domain-name

以下代码示例演示了如何使用 create-domain-name。

Amazon CLI

创建自定义域名

以下 create-domain-name 示例为 API 创建一个区域自定义域名。

```
aws apigatewayv2 create-domain-name \
    --domain-name regional.example.com \
    --domain-name-configurations CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678
```

输出：

```
{
    "ApiMappingSelectionExpression": "$request.basepath",
    "DomainName": "regional.example.com",
    "DomainNameConfigurations": [
        {
            "ApiGatewayDomainName": "d-id.execute-api.us-west-2.amazonaws.com",
```

```
            "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
            "EndpointType": "REGIONAL",
            "HostedZoneId": "123456789111",
            "SecurityPolicy": "TLS_1_2",
            "DomainNameStatus": "AVAILABLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomainName。

`create-integration`

以下代码示例演示了如何使用 `create-integration`。

Amazon CLI

创建 WebSocket API 集成

以下 `create-integration` 示例为 WebSocket API 创建一个模拟集成。

```
aws apigatewayv2 create-integration \
    --api-id aabbccddee \
    --passthrough-behavior WHEN_NO_MATCH \
    --timeout-in-millis 29000 \
    --connection-type INTERNET \
    --integration-type MOCK
```

输出：

```
{
    "ConnectionType": "INTERNET",
    "IntegrationId": "0abcdef",
    "IntegrationResponseSelectionExpression": "${integration.response.statuscode}",
    "IntegrationType": "MOCK",
    "PassthroughBehavior": "WHEN_NO_MATCH",
    "PayloadFormatVersion": "1.0",
    "TimeoutInMillis": 29000
```

```
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置 WebSocket API 集成请求。

创建 HTTP API 集成

以下 `create-integration` 示例为 HTTP API 创建 Amazon Lambda 集成。

```
aws apigatewayv2 create-integration \
    --api-id a1b2c3d4 \
    --integration-type AWS_PROXY \
    --integration-uri arn:aws:lambda:us-west-2:123456789012:function:my-function \
    --payload-format-version 2.0
```

输出：

```
{
    "ConnectionType": "INTERNET",
    "IntegrationId": "0abcdef",
    "IntegrationMethod": "POST",
    "IntegrationType": "AWS_PROXY",
    "IntegrationUri": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "PayloadFormatVersion": "2.0",
    "TimeoutInMillis": 30000
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIntegration。

`create-route`

以下代码示例演示了如何使用 `create-route`。

Amazon CLI

为 WebSocket 或 HTTP API 创建 $default 路由

以下 `create-route` 示例为 WebSocket 或 HTTP API 创建 `$default` 路由。

```
aws apigatewayv2 create-route \
    --api-id aabbccddee \
```

```
    --route-key '$default'
```

输出：

```
{
    "ApiKeyRequired": false,
    "AuthorizationType": "NONE",
    "RouteKey": "$default",
    "RouteId": "1122334"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 WebSocket API 的路由。

为 HTTP API 创建路由

以下 create-route 示例创建一个接受 POST 请求的名为 signup 的路由。

```
aws apigatewayv2 create-route \
    --api-id aabbccddee \
    --route-key 'POST /signup'
```

输出：

```
{
    "ApiKeyRequired": false,
    "AuthorizationType": "NONE",
    "RouteKey": "POST /signup",
    "RouteId": "1122334"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoute。

create-routing-rule

以下代码示例演示了如何使用 create-routing-rule。

Amazon CLI

创建路由规则

以下 create-routing-rule 示例创建优先级为 50 的路由规则。

```
aws apigatewayv2 create-routing-rule \
    --domain-name 'regional.example.com' \
    --priority 50 \
    --conditions '[ \
        { \
            "MatchBasePaths": { \
                "AnyOf": [ \
                    "PetStoreShopper" \
                ] \
            } \
        } \
    ]' \
    --actions '[ \
        { \
            "InvokeApi": { \
                "ApiId": "abcd1234", \
                "Stage": "prod" \
            } \
        } \
    ]'
```

输出：

```
{
    "Actions": [
        {
            "InvokeApi": {
                "ApiId": "abcd1234",
                "Stage": "prod",
                "StripBasePath": false
            }
        }
    ],
    "Conditions": [
        {
            "MatchBasePaths": {
                "AnyOf": [
                    "PetStoreShopper"
                ]
            }
        }
```

```
    ],
    "Priority": 50,
    "RoutingRuleArn": "arn:aws:apigateway:us-east-2:123456789012:/domainnames/
regional.example.com/routingrules/aaa111",
    "RoutingRuleId": "aaa111"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的将 API 阶段连接到 REST API 的自定义域名的路由规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateRoutingRule。

create-stage

以下代码示例演示了如何使用 create-stage。

Amazon CLI

创建阶段

以下 create-stage 示例为 API 创建一个名为 dev 的阶段。

```
aws apigatewayv2 create-stage \
    --api-id a1b2c3d4 \
    --stage-name dev
```

输出：

```
{
    "CreatedDate": "2020-04-06T23:23:46Z",
    "DefaultRouteSettings": {
        "DetailedMetricsEnabled": false
    },
    "LastUpdatedDate": "2020-04-06T23:23:46Z",
    "RouteSettings": {},
    "StageName": "dev",
    "StageVariables": {},
    "Tags": {}
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的阶段。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStage。

create-vpc-link

以下代码示例演示了如何使用 create-vpc-link。

Amazon CLI

为 HTTP API 创建 VPC 链接

以下 create-vpc-link 示例为 HTTP API 创建 VPC 链接。

```
aws apigatewayv2 create-vpc-link \
    --name MyVpcLink \
    --subnet-ids subnet-aaaa subnet-bbbb \
    --security-group-ids sg1234 sg5678
```

输出：

```
{
    "CreatedDate": "2020-04-07T00:11:46Z",
    "Name": "MyVpcLink",
    "SecurityGroupIds": [
        "sg1234",
        "sg5678"
    ],
    "SubnetIds": [
        "subnet-aaaa",
        "subnet-bbbb"
    ],
    "Tags": {},
    "VpcLinkId": "abcd123",
    "VpcLinkStatus": "PENDING",
    "VpcLinkStatusMessage": "VPC link is provisioning ENIs",
    "VpcLinkVersion": "V2"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcLink。

delete-access-log-settings

以下代码示例演示了如何使用 delete-access-log-settings。

Amazon CLI

禁用 API 的访问日志记录

以下 `delete-access-log-settings` 示例删除 API `$default` 阶段的访问日志设置。要禁用阶段的访问日志记录，请删除其访问日志设置。

```
aws apigatewayv2 delete-access-log-settings \
    --api-id a1b2c3d4 \
    --stage-name '$default'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置日志记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessLogSettings。

delete-api-mapping

以下代码示例演示了如何使用 `delete-api-mapping`。

Amazon CLI

删除 API 映射

以下 `delete-api-mapping` 示例删除 `api.example.com` 自定义域名的 API 映射。

```
aws apigatewayv2 delete-api-mapping \
    --api-mapping-id a1b2c3 \
    --domain-name api.example.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApiMapping。

delete-api

以下代码示例演示了如何使用 `delete-api`。

Amazon CLI

删除 API

以下 `delete-api` 示例删除 API。

```
aws apigatewayv2 delete-api \
    --api-id a1b2c3d4
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 和使用 WebSocket API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApi。

`delete-authorizer`

以下代码示例演示了如何使用 `delete-authorizer`。

Amazon CLI

删除授权方

以下 `delete-authorizer` 示例删除授权方。

```
aws apigatewayv2 delete-authorizer \
    --api-id a1b2c3d4 \
    --authorizer-id a1b2c3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAuthorizer。

`delete-cors-configuration`

以下代码示例演示了如何使用 `delete-cors-configuration`。

Amazon CLI

删除 HTTP API 的 CORS 配置

以下 `delete-cors-configuration` 示例通过删除 HTTP API 的 CORS 配置来禁用 HTTP API 的 CORS。

```
aws apigatewayv2 delete-cors-configuration \
    --api-id a1b2c3d4
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置 CORS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCorsConfiguration。

delete-deployment

以下代码示例演示了如何使用 `delete-deployment`。

Amazon CLI

删除部署

以下 `delete-deployment` 示例删除 API 的部署。

```
aws apigatewayv2 delete-deployment \
    --api-id a1b2c3d4 \
    --deployment-id a1b2c3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API 部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeployment。

delete-domain-name

以下代码示例演示了如何使用 `delete-domain-name`。

Amazon CLI

删除自定义域名

以下 delete-domain-name 示例删除一个自定义域名。

```
aws apigatewayv2 delete-domain-name \
    --domain-name api.example.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomainName。

delete-integration

以下代码示例演示了如何使用 delete-integration。

Amazon CLI

删除集成

以下 delete-integration 示例删除 API 集成。

```
aws apigatewayv2 delete-integration \
    --api-id a1b2c3d4 \
    --integration-id a1b2c3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置集成和设置 WebSocket API 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIntegration。

delete-route-settings

以下代码示例演示了如何使用 delete-route-settings。

Amazon CLI

删除路由设置

以下 delete-route-settings 示例删除指定路由的路由设置。

```
aws apigatewayv2 delete-route-settings \
    --api-id a1b2c3d4 \
    --stage-name dev \
    --route-key 'GET /pets'
```

此命令不生成任何输出。

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRouteSettings。

delete-route

以下代码示例演示了如何使用 delete-route。

Amazon CLI

删除路由

以下 delete-route 示例删除 API 路由。

```
aws apigatewayv2 delete-route \
    --api-id a1b2c3d4 \
    --route-id a1b2c3
```

此命令不生成任何输出。

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoute。

delete-routing-rule

以下代码示例演示了如何使用 delete-routing-rule。

Amazon CLI

删除路由规则

以下 delete-routing-rule 示例删除自定义域名的路由规则。

```
aws apigatewayv2 delete-routing-rule \
    --domain-name 'regional.example.com' \
```

```
    --routing-rule-id aaa111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的将 API 阶段连接到 REST API 的自定义域名的路由规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteRoutingRule。

delete-stage

以下代码示例演示了如何使用 delete-stage。

Amazon CLI

删除阶段

以下 delete-stage 示例删除 API 的 test 阶段。

```
aws apigatewayv2 delete-stage \
    --api-id a1b2c3d4 \
    --stage-name test
```

此命令不生成任何输出。

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的阶段。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStage。

delete-vpc-link

以下代码示例演示了如何使用 delete-vpc-link。

Amazon CLI

删除 HTTP API 的 VPC 链接

以下 delete-vpc-link 示例删除 VPC 链接。

```
aws apigatewayv2 delete-vpc-link \
    --vpc-link-id abcd123
```

此命令不生成任何输出。

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpcLink。

`export-api`

以下代码示例演示了如何使用 `export-api`。

Amazon CLI

导出 HTTP API 的 OpenAPI 定义

以下 `export-api` 示例将名为 `prod` 的 API 阶段的 OpenAPI 3.0 定义导出到名为 `stage-definition.yaml` 的 YAML 文件。默认情况下，导出的定义文件包含 API Gateway 扩展名。

```
aws apigatewayv2 export-api \
    --api-id a1b2c3d4 \
    --output-type YAML \
    --specification OAS30 \
    --stage-name prod \
    stage-definition.yaml
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的从 API Gateway 导出 HTTP API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportApi。

`get-api-mapping`

以下代码示例演示了如何使用 `get-api-mapping`。

Amazon CLI

获取有关自定义域名的 API 映射的信息

以下 `get-api-mapping` 示例显示有关 `api.example.com` 自定义域名的 API 映射的信息。

```
aws apigatewayv2 get-api-mapping \
    --api-mapping-id a1b2c3 \
    --domain-name api.example.com
```

输出：

```
{
    "ApiId": "a1b2c3d4",
    "ApiMappingId": "a1b2c3d5",
    "ApiMappingKey": "myTestApi"
    "Stage": "test"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApiMapping。

get-api-mappings

以下代码示例演示了如何使用 get-api-mappings。

Amazon CLI

获取自定义域名的 API 映射

以下 get-api-mappings 示例显示 api.example.com 自定义域名的所有 API 映射列表。

```
aws apigatewayv2 get-api-mappings \
    --domain-name api.example.com
```

输出：

```
{
    "Items": [
        {
            "ApiId": "a1b2c3d4",
            "ApiMappingId": "a1b2c3d5",
            "ApiMappingKey": "myTestApi"
            "Stage": "test"
        },
        {
            "ApiId": "a5b6c7d8",
            "ApiMappingId": "a1b2c3d6",
            "ApiMappingKey": "myDevApi"
            "Stage": "dev"
```

```
        },
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApiMappings。

get-api

以下代码示例演示了如何使用 get-api。

Amazon CLI

检索有关 API 的信息

以下 get-api 示例显示有关 API 的信息。

```
aws apigatewayv2 get-api \
    --api-id a1b2c3d4
```

输出：

```
{
    "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
    "ApiId": "a1b2c3d4",
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "CreatedDate": "2020-03-28T00:32:37Z",
    "Name": "my-api",
    "ProtocolType": "HTTP",
    "RouteSelectionExpression": "$request.method $request.path",
    "Tags": {
        "department": "finance"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApi。

get-apis

以下代码示例演示了如何使用 get-apis。

Amazon CLI

检索 API 列表

以下 `get-apis` 示例列出当前用户的所有 API。

```
aws apigatewayv2 get-apis
```

输出：

```
{
    "Items": [
        {
            "ApiEndpoint": "wss://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
            "ApiId": "a1b2c3d4",
            "ApiKeySelectionExpression": "$request.header.x-api-key",
            "CreatedDate": "2020-04-07T20:21:59Z",
            "Name": "my-websocket-api",
            "ProtocolType": "WEBSOCKET",
            "RouteSelectionExpression": "$request.body.message",
            "Tags": {}
        },
        {
            "ApiEndpoint": "https://a1b2c3d5.execute-api.us-west-2.amazonaws.com",
            "ApiId": "a1b2c3d5",
            "ApiKeySelectionExpression": "$request.header.x-api-key",
            "CreatedDate": "2020-04-07T20:23:50Z",
            "Name": "my-http-api",
            "ProtocolType": "HTTP",
            "RouteSelectionExpression": "$request.method $request.path",
            "Tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 和使用 WebSocket API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApis。

`get-authorizer`

以下代码示例演示了如何使用 `get-authorizer`。

Amazon CLI

检索有关授权方的信息

以下 get-authorizer 示例显示有关授权方的信息。

```
aws apigatewayv2 get-authorizer \
    --api-id a1b2c3d4 \
    --authorizer-id a1b2c3
```

输出：

```
{
    "AuthorizerId": "a1b2c3",
    "AuthorizerType": "JWT",
    "IdentitySource": [
        "$request.header.Authorization"
    ],
    "JwtConfiguration": {
        "Audience": [
            "123456abc"
        ],
        "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
    },
    "Name": "my-jwt-authorizer"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizer。

get-authorizers

以下代码示例演示了如何使用 get-authorizers。

Amazon CLI

检索 API 的授权方列表

以下 get-authorizers 示例显示 API 的所有授权方的列表。

```
aws apigatewayv2 get-authorizers \
    --api-id a1b2c3d4
```

输出：

```
{
    "Items": [
        {
            "AuthorizerId": "a1b2c3",
            "AuthorizerType": "JWT",
            "IdentitySource": [
                "$request.header.Authorization"
            ],
            "JwtConfiguration": {
                "Audience": [
                    "123456abc"
                ],
                "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_abc123"
            },
            "Name": "my-jwt-authorizer"
        },
        {
            "AuthorizerId": "a1b2c4",
            "AuthorizerType": "JWT",
            "IdentitySource": [
                "$request.header.Authorization"
            ],
            "JwtConfiguration": {
                "Audience": [
                    "6789abcde"
                ],
                "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_abc234"
            },
            "Name": "new-jwt-authorizer"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizers。

get-deployment

以下代码示例演示了如何使用 get-deployment。

Amazon CLI

检索有关部署的信息

以下 get-deployment 示例显示有关部署的信息。

```
aws apigatewayv2 get-deployment \
    --api-id a1b2c3d4 \
    --deployment-id abcdef
```

输出：

```
{
    "AutoDeployed": true,
    "CreatedDate": "2020-04-07T23:58:40Z",
    "DeploymentId": "abcdef",
    "DeploymentStatus": "DEPLOYED",
    "Description": "Automatic deployment triggered by changes to the Api
 configuration"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API 部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployment。

get-deployments

以下代码示例演示了如何使用 get-deployments。

Amazon CLI

检索部署列表

以下 get-deployments 示例显示所有 API 部署的列表。

```
aws apigatewayv2 get-deployments \
```

```
    --api-id a1b2c3d4
```

输出：

```
{
    "Items": [
        {
            "AutoDeployed": true,
            "CreatedDate": "2020-04-07T23:58:40Z",
            "DeploymentId": "abcdef",
            "DeploymentStatus": "DEPLOYED",
            "Description": "Automatic deployment triggered by changes to the Api
 configuration"
        },
        {
            "AutoDeployed": true,
            "CreatedDate": "2020-04-06T00:33:00Z",
            "DeploymentId": "bcdefg",
            "DeploymentStatus": "DEPLOYED",
            "Description": "Automatic deployment triggered by changes to the Api
 configuration"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的 API 部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployments。

get-domain-name

以下代码示例演示了如何使用 get-domain-name。

Amazon CLI

检索有关自定义域名的信息

以下 get-domain-name 示例显示有关自定义域名的信息。

```
aws apigatewayv2 get-domain-name \
    --domain-name api.example.com
```

输出：

```
{
    "ApiMappingSelectionExpression": "$request.basepath",
    "DomainName": "api.example.com",
    "DomainNameConfigurations": [
        {
            "ApiGatewayDomainName": "d-1234.execute-api.us-west-2.amazonaws.com",
            "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
            "EndpointType": "REGIONAL",
            "HostedZoneId": "123456789111",
            "SecurityPolicy": "TLS_1_2",
            "DomainNameStatus": "AVAILABLE"
        }
    ],
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainName。

get-domain-names

以下代码示例演示了如何使用 get-domain-names。

Amazon CLI

检索自定义域名列表

以下 get-domain-names 示例显示当前用户的所有自定义域名列表。

```
aws apigatewayv2 get-domain-names
```

输出：

```
{
    "Items": [
        {
            "ApiMappingSelectionExpression": "$request.basepath",
            "DomainName": "api.example.com",
            "DomainNameConfigurations": [
```

```
                {
                    "ApiGatewayDomainName": "d-1234.execute-api.us-
west-2.amazonaws.com",
                    "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
                    "EndpointType": "REGIONAL",
                    "HostedZoneId": "123456789111",
                    "SecurityPolicy": "TLS_1_2",
                    "DomainNameStatus": "AVAILABLE"
                }
            ]
        },
        {
            "ApiMappingSelectionExpression": "$request.basepath",
            "DomainName": "newApi.example.com",
            "DomainNameConfigurations": [
                {
                    "ApiGatewayDomainName": "d-5678.execute-api.us-
west-2.amazonaws.com",
                    "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
                    "EndpointType": "REGIONAL",
                    "HostedZoneId": "123456789222",
                    "SecurityPolicy": "TLS_1_2",
                    "DomainNameStatus": "AVAILABLE"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainNames。

`get-integration`

以下代码示例演示了如何使用 `get-integration`。

Amazon CLI

检索有关集成的信息

以下 get-integration 示例显示有关集成的信息。

```
aws apigatewayv2 get-integration \
    --api-id a1b2c3d4 \
    --integration-id a1b2c3
```

输出：

```
{
    "ApiGatewayManaged": true,
    "ConnectionType": "INTERNET",
    "IntegrationId": "a1b2c3",
    "IntegrationMethod": "POST",
    "IntegrationType": "AWS_PROXY",
    "IntegrationUri": "arn:aws:lambda:us-west-2:12356789012:function:hello12",
    "PayloadFormatVersion": "2.0",
    "TimeoutInMillis": 30000
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置集成和设置 WebSocket API 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIntegration。

get-integrations

以下代码示例演示了如何使用 get-integrations。

Amazon CLI

检索集成列表

以下 get-integrations 示例显示所有 API 集成的列表。

```
aws apigatewayv2 get-integrations \
    --api-id a1b2c3d4
```

输出：

```
{
    "Items": [
```

```
        {
            "ApiGatewayManaged": true,
            "ConnectionType": "INTERNET",
            "IntegrationId": "a1b2c3",
            "IntegrationMethod": "POST",
            "IntegrationType": "AWS_PROXY",
            "IntegrationUri": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
            "PayloadFormatVersion": "2.0",
            "TimeoutInMillis": 30000
        },
        {
            "ConnectionType": "INTERNET",
            "IntegrationId": "a1b2c4",
            "IntegrationMethod": "ANY",
            "IntegrationType": "HTTP_PROXY",
            "IntegrationUri": "https://www.example.com",
            "PayloadFormatVersion": "1.0",
            "TimeoutInMillis": 30000
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置集成和设置 WebSocket API 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIntegrations。

get-route

以下代码示例演示了如何使用 get-route。

Amazon CLI

检索有关路由的信息

以下 get-route 示例显示有关路由的信息。

```
aws apigatewayv2 get-route \
    --api-id a1b2c3d4 \
    --route-id 72jz1wk
```

输出：

```
{
    "ApiKeyRequired": false,
    "AuthorizationType": "NONE",
    "RouteId": "72jz1wk",
    "RouteKey": "ANY /pets",
    "Target": "integrations/a1b2c3"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRoute。

get-routes

以下代码示例演示了如何使用 get-routes。

Amazon CLI

检索路由列表

以下 get-routes 示例显示所有 API 路由的列表。

```
aws apigatewayv2 get-routes \
    --api-id a1b2c3d4
```

输出：

```
{
    "Items": [
        {
            "ApiKeyRequired": false,
            "AuthorizationType": "NONE",
            "RouteId": "72jz1wk",
            "RouteKey": "ANY /admin",
            "Target": "integrations/a1b2c3"
        },
        {
            "ApiGatewayManaged": true,
            "ApiKeyRequired": false,
            "AuthorizationType": "NONE",
            "RouteId": "go65gqi",
            "RouteKey": "$default",
```

```
            "Target": "integrations/a1b2c4"
        }
    ]
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRoutes。

get-routing-rule

以下代码示例演示了如何使用 get-routing-rule。

Amazon CLI

获取路由规则

以下 get-routing-rule 示例获取域名的路由规则。

```
aws apigatewayv2 get-routing-rule \
    --domain-name 'regional.example.com' \
    --routing-rule-id aaa111
```

输出：

```
{
    "Actions": [
        {
            "InvokeApi": {
                "ApiId": "abcd1234",
                "Stage": "prod",
                "StripBasePath": false
            }
        }
    ],
    "Conditions": [
        {
            "MatchBasePaths": {
                "AnyOf": [
                    "PetStoreShopper"
                ]
            }
        }
```

```
    ],
    "Priority": 50,
    "RoutingRuleArn": "arn:aws:apigateway:us-east-2:123456789012:/domainnames/
regional.example.com/routingrules/aaa111",
    "RoutingRuleId": "aaa111"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的将 API 阶段连接到 REST API 的自定义域名的路由规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetRoutingRule。

get-stage

以下代码示例演示了如何使用 get-stage。

Amazon CLI

检索有关阶段的信息

以下 get-stage 示例显示有关 API prod 阶段的信息。

```
aws apigatewayv2 get-stage \
    --api-id a1b2c3d4 \
    --stage-name prod
```

输出：

```
{
    "CreatedDate": "2020-04-08T00:36:05Z",
    "DefaultRouteSettings": {
        "DetailedMetricsEnabled": false
    },
    "DeploymentId": "x1zwyv",
    "LastUpdatedDate": "2020-04-08T00:36:13Z",
    "RouteSettings": {},
    "StageName": "prod",
    "StageVariables": {
        "function": "my-prod-function"
    },
    "Tags": {}
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的阶段。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStage。

get-stages

以下代码示例演示了如何使用 get-stages。

Amazon CLI

检索阶段列表

以下 get-stages 示例列出 API 的所有阶段。

```
aws apigatewayv2 get-stages \
    --api-id a1b2c3d4
```

输出：

```
{
    "Items": [
        {
            "ApiGatewayManaged": true,
            "AutoDeploy": true,
            "CreatedDate": "2020-04-08T00:08:44Z",
            "DefaultRouteSettings": {
                "DetailedMetricsEnabled": false
            },
            "DeploymentId": "dty748",
            "LastDeploymentStatusMessage": "Successfully deployed stage with
 deployment ID 'dty748'",
            "LastUpdatedDate": "2020-04-08T00:09:49Z",
            "RouteSettings": {},
            "StageName": "$default",
            "StageVariables": {},
            "Tags": {}
        },
        {
            "AutoDeploy": true,
            "CreatedDate": "2020-04-08T00:35:06Z",
            "DefaultRouteSettings": {
                "DetailedMetricsEnabled": false
            },
```

```
            "LastUpdatedDate": "2020-04-08T00:35:48Z",
            "RouteSettings": {},
            "StageName": "dev",
            "StageVariables": {
                "function": "my-dev-function"
            },
            "Tags": {}
        },
        {
            "CreatedDate": "2020-04-08T00:36:05Z",
            "DefaultRouteSettings": {
                "DetailedMetricsEnabled": false
            },
            "DeploymentId": "x1zwyv",
            "LastUpdatedDate": "2020-04-08T00:36:13Z",
            "RouteSettings": {},
            "StageName": "prod",
            "StageVariables": {
                "function": "my-prod-function"
            },
            "Tags": {}
        }
    ]
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的阶段。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStages。

get-tags

以下代码示例演示了如何使用 get-tags。

Amazon CLI

检索资源的标签列表

以下 get-tags 示例列出 API 的所有标签。

```
aws apigatewayv2 get-tags \
    --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4
```

输出：

```
{
    "Tags": {
        "owner": "dev-team",
        "environment": "prod"
    }
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的标记您的 API Gateway 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTags。

get-vpc-link

以下代码示例演示了如何使用 get-vpc-link。

Amazon CLI

检索有关 VPC 链接的信息

以下 get-vpc-link 示例显示有关 VPC 链接的信息。

```
aws apigatewayv2 get-vpc-link \
    --vpc-link-id abcd123
```

输出：

```
{
    "CreatedDate": "2020-04-07T00:27:47Z",
    "Name": "MyVpcLink",
    "SecurityGroupIds": [
        "sg1234",
        "sg5678"
    ],
    "SubnetIds": [
        "subnet-aaaa",
        "subnet-bbbb"
    ],
    "Tags": {},
    "VpcLinkId": "abcd123",
    "VpcLinkStatus": "AVAILABLE",
    "VpcLinkStatusMessage": "VPC link is ready to route traffic",
    "VpcLinkVersion": "V2"
```

```
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVpcLink。

get-vpc-links

以下代码示例演示了如何使用 get-vpc-links。

Amazon CLI

检索 VPC 链接列表

以下 get-vpc-links 示例显示当前用户所有 VPC 链接的列表。

```
aws apigatewayv2 get-vpc-links
```

输出：

```
{
    "Items": [
        {
            "CreatedDate": "2020-04-07T00:27:47Z",
            "Name": "MyVpcLink",
            "SecurityGroupIds": [
                "sg1234",
                "sg5678"
            ],
            "SubnetIds": [
                "subnet-aaaa",
                "subnet-bbbb"
            ],
            "Tags": {},
            "VpcLinkId": "abcd123",
            "VpcLinkStatus": "AVAILABLE",
            "VpcLinkStatusMessage": "VPC link is ready to route traffic",
            "VpcLinkVersion": "V2"
        }
        {
            "CreatedDate": "2020-04-07T00:27:47Z",
            "Name": "MyOtherVpcLink",
```

```
            "SecurityGroupIds": [
                "sg1234",
                "sg5678"
            ],
            "SubnetIds": [
                "subnet-aaaa",
                "subnet-bbbb"
            ],
            "Tags": {},
            "VpcLinkId": "abcd456",
            "VpcLinkStatus": "AVAILABLE",
            "VpcLinkStatusMessage": "VPC link is ready to route traffic",
            "VpcLinkVersion": "V2"
        }
    ]
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVpcLinks。

import-api

以下代码示例演示了如何使用 import-api。

Amazon CLI

导入 HTTP API

以下 import-api 示例从名为 api-definition.yaml 的 OpenAPI 3.0 定义文件创建 HTTP API。

```
aws apigatewayv2 import-api \
    --body file://api-definition.yaml
```

的内容api-definition.yaml：

```
openapi: 3.0.1
info:
    title: My Lambda API
    version: v1.0
paths:
```

```
    /hello:
        x-amazon-apigateway-any-method:
            x-amazon-apigateway-integration:
                payloadFormatVersion: 2.0
                type: aws_proxy
                httpMethod: POST
                uri: arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123456789012:function:hello/invocations
                connectionType: INTERNET
```

输出：

```
{
    "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
    "ApiId": "a1b2c3d4",
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "CreatedDate": "2020-04-08T17:19:38+00:00",
    "Name": "My Lambda API",
    "ProtocolType": "HTTP",
    "RouteSelectionExpression": "$request.method $request.path",
    "Tags": {},
    "Version": "v1.0"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 OpenAPI 定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportApi。

`list-routing-rules`

以下代码示例演示了如何使用 `list-routing-rules`。

Amazon CLI

列出路由规则

以下 `list-routing-rules` 示例列出域名的路由规则。

```
aws apigatewayv2 list-routing-rules \
    --domain-name 'regional.example.com'
```

输出：

```
{
    "RoutingRules": [
        {
            "Actions": [
                {
                    "InvokeApi": {
                        "ApiId": "abcd1234",
                        "Stage": "prod",
                        "StripBasePath": false
                    }
                }
            ],
            "Conditions": [
                {
                    "MatchBasePaths": {
                        "AnyOf": [
                            "PetStoreShopper"
                        ]
                    }
                }
            ],
            "Priority": 150,
            "RoutingRuleArn": "arn:aws:apigateway:us-east-1:123456789012:/
domainnames/regional.example.com/routingrules/aaa111",
            "RoutingRuleId": "aaa111"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的将 API 阶段连接到 REST API 的自定义域名的路由规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListRoutingRules。

put-routing-rule

以下代码示例演示了如何使用 put-routing-rule。

Amazon CLI

更新路由规则

以下 put-routing-rule 示例更新路由规则的优先级。

```
aws apigatewayv2 put-routing-rule \
    --domain-name 'regional.example.com' \
    --priority 150 \
    --conditions '[ \
        { \
            "MatchBasePaths": { \
                "AnyOf": [ \
                    "PetStoreShopper" \
                ] \
            } \
        } \
    ]' \
    --actions '[ \
        { \
            "InvokeApi": { \
                "ApiId": "abcd1234", \
                "Stage": "prod" \
            } \
        } \
    ]'
```

输出：

```
{
    "Actions": [
        {
            "InvokeApi": {
                "ApiId": "abcd1234",
                "Stage": "prod",
                "StripBasePath": false
            }
        }
    ],
    "Conditions": [
        {
            "MatchBasePaths": {
                "AnyOf": [
                    "PetStoreShopper"
                ]
            }
        }
    ],
    "Priority": 150,
```

```
    "RoutingRuleArn": "arn:aws:apigateway:us-east-2:123456789012:/domainnames/
regional.example.com/routingrules/aaa111",
    "RoutingRuleId": "aaa111"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的将 API 阶段连接到 REST API 的自定义域名的路由规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutRoutingRule。

reimport-api

以下代码示例演示了如何使用 reimport-api。

Amazon CLI

重新导入 HTTP API

以下 reimport-api 示例将现有 HTTP API 更新为使用 api-definition.yaml 中指定的 OpenAPI 3.0 定义。

```
aws apigatewayv2 reimport-api \
    --body file://api-definition.yaml \
    --api-id a1b2c3d4
```

的内容api-definition.yaml：

```
openapi: 3.0.1
info:
    title: My Lambda API
    version: v1.0
paths:
    /hello:
        x-amazon-apigateway-any-method:
            x-amazon-apigateway-integration:
                payloadFormatVersion: 2.0
                type: aws_proxy
                httpMethod: POST
                uri: arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:12356789012:function:hello/invocations
                connectionType: INTERNET
```

输出：

```
{
    "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
    "ApiId": "a1b2c3d4",
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "CreatedDate": "2020-04-08T17:19:38+00:00",
    "Name": "My Lambda API",
    "ProtocolType": "HTTP",
    "RouteSelectionExpression": "$request.method $request.path",
    "Tags": {},
    "Version": "v1.0"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 OpenAPI 定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReimportApi。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例向指定的 API 添加了键名称为 Department 和值为 Accounting 的标签。

```
aws apigatewayv2 tag-resource \
    --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4 \
    --tags Department=Accounting
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的标记您的 API Gateway 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从指定 API 中移除键名为 Project 和 Owner 的标签。

```
aws apigatewayv2 untag-resource \
    --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4 \
    --tag-keys Project Owner
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的标记您的 API Gateway 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-api-mapping

以下代码示例演示了如何使用 update-api-mapping。

Amazon CLI

更新 API 映射

以下 update-api-mapping 示例更改自定义域名的 API 映射。这样，使用指定 API 和阶段的自定义域名的基础 URL 会变为 https://api.example.com/dev。

```
aws apigatewayv2 update-api-mapping \
    --api-id a1b2c3d4 \
    --stage dev \
    --domain-name api.example.com \
    --api-mapping-id 0qzs2sy7bh \
    --api-mapping-key dev
```

输出：

```
{
    "ApiId": "a1b2c3d4",
    "ApiMappingId": "0qzs2sy7bh",
    "ApiMappingKey": "dev"
    "Stage": "dev"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApiMapping。

update-api

以下代码示例演示了如何使用 update-api。

Amazon CLI

为 HTTP API 启用 CORS

以下 update-api 示例更新指定 API 的 CORS 配置，以允许来自 https://www.example.com 的请求。

```
aws apigatewayv2 update-api \
    --api-id a1b2c3d4 \
    --cors-configuration AllowOrigins=https://www.example.com
```

输出：

```
{
    "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
    "ApiId": "a1b2c3d4",
    "ApiKeySelectionExpression": "$request.header.x-api-key",
    "CorsConfiguration": {
        "AllowCredentials": false,
        "AllowHeaders": [
            "header1",
            "header2"
        ],
        "AllowMethods": [
            "GET",
            "OPTIONS"
        ],
        "AllowOrigins": [
            "https://www.example.com"
        ]
    },
    "CreatedDate": "2020-04-08T18:39:37+00:00",
    "Name": "my-http-api",
    "ProtocolType": "HTTP",
```

```
    "RouteSelectionExpression": "$request.method $request.path",
    "Tags": {},
    "Version": "v1.0"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置 CORS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApi。

update-authorizer

以下代码示例演示了如何使用 update-authorizer。

Amazon CLI

更新授权方

以下 update-authorizer 示例将 JWT 授权方的身份源更改为名为 Authorization 的标头。

```
aws apigatewayv2 update-authorizer \
    --api-id a1b2c3d4 \
    --authorizer-id a1b2c3 \
    --identity-source '$request.header.Authorization'
```

输出：

```
{
    "AuthorizerId": "a1b2c3",
    "AuthorizerType": "JWT",
    "IdentitySource": [
        "$request.header.Authorization"
    ],
    "JwtConfiguration": {
        "Audience": [
            "123456abc"
        ],
        "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
    },
    "Name": "my-jwt-authorizer"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAuthorizer。

update-deployment

以下代码示例演示了如何使用 update-deployment。

Amazon CLI

更改部署的说明

以下 update-deployment 示例更新部署的说明。

```
aws apigatewayv2 update-deployment \
    --api-id a1b2c3d4 \
    --deployment-id abcdef \
    --description 'Manual deployment to fix integration test failures.'
```

输出：

```
{
    "AutoDeployed": false,
    "CreatedDate": "2020-02-05T16:21:48+00:00",
    "DeploymentId": "abcdef",
    "DeploymentStatus": "DEPLOYED",
    "Description": "Manual deployment to fix integration test failures."
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中开发 HTTP API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeployment。

update-domain-name

以下代码示例演示了如何使用 update-domain-name。

Amazon CLI

更新自定义域名

以下 update-domain-name 示例为 api.example.com 自定义域名指定新的 ACM 证书。

```
aws apigatewayv2 update-domain-name \
    --domain-name api.example.com \
    --domain-name-configurations CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678
```

输出：

```
{
    "ApiMappingSelectionExpression": "$request.basepath",
    "DomainName": "regional.example.com",
    "DomainNameConfigurations": [
        {
            "ApiGatewayDomainName": "d-id.execute-api.us-west-2.amazonaws.com",
            "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
            "EndpointType": "REGIONAL",
            "HostedZoneId": "123456789111",
            "SecurityPolicy": "TLS_1_2",
            "DomainNameStatus": "AVAILABLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在 API Gateway 中设置区域自定义域名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainName。

update-integration

以下代码示例演示了如何使用 update-integration。

Amazon CLI

更新 Lambda 集成

以下 update-integration 示例将现有 Amazon Lambda 集成更新为使用指定的 Lambda 函数。

```
aws apigatewayv2 update-integration \
    --api-id a1b2c3d4 \
    --integration-id a1b2c3 \
```

```
    --integration-uri arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123456789012:function:my-new-function/invocations
```

输出：

```
{
    "ConnectionType": "INTERNET",
    "IntegrationId": "a1b2c3",
    "IntegrationMethod": "POST",
    "IntegrationType": "AWS_PROXY",
    "IntegrationUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/
functions/arn:aws:lambda:us-west-2:123456789012:function:my-new-function/
invocations",
    "PayloadFormatVersion": "2.0",
    "TimeoutInMillis": 5000
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的为 HTTP API 配置集成和设置 WebSocket API 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIntegration。

update-route

以下代码示例演示了如何使用 update-route。

Amazon CLI

示例 1：更新路由的集成

以下 update-route 示例更新指定路由的集成。

```
aws apigatewayv2 update-route \
    --api-id a1b2c3d4 \
    --route-id a1b2c3 \
    --target integrations/a1b2c6
```

输出：

```
{
    "ApiKeyRequired": false,
```

```
    "AuthorizationType": "NONE",
    "RouteId": "a1b2c3",
    "RouteKey": "ANY /pets",
    "Target": "integrations/a1b2c6"
}
```

示例 2：向路由添加授权方

以下 update-route 示例将指定路由更新为使用 JWT 授权方。

```
aws apigatewayv2 update-route \
    --api-id a1b2c3d4  \
    --route-id a1b2c3  \
    --authorization-type JWT \
    --authorizer-id a1b2c5 \
    --authorization-scopes user.id user.email
```

输出：

```
{
    "ApiKeyRequired": false,
    "AuthorizationScopes": [
        "user.id",
        "user.email"
    ],
    "AuthorizationType": "JWT",
    "AuthorizerId": "a1b2c5",
    "OperationName": "GET HTTP",
    "RequestParameters": {},
    "RouteId": "a1b2c3",
    "RouteKey": "GET /pets",
    "Target": "integrations/a1b2c6"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 JWT 授权方控制对 HTTP API 的访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoute。

update-stage

以下代码示例演示了如何使用 update-stage。

Amazon CLI

配置自定义节流

以下 update-stage 示例为 API 的指定阶段和路由配置自定义节流。

```
aws apigatewayv2 update-stage \
    --api-id a1b2c3d4 \
    --stage-name dev \
    --route-settings '{"GET /pets":
{"ThrottlingBurstLimit":100,"ThrottlingRateLimit":2000}}'
```

输出：

```
{
    "CreatedDate": "2020-04-05T16:21:16+00:00",
    "DefaultRouteSettings": {
        "DetailedMetricsEnabled": false
    },
    "DeploymentId": "shktxb",
    "LastUpdatedDate": "2020-04-08T22:23:17+00:00",
    "RouteSettings": {
        "GET /pets": {
            "ThrottlingBurstLimit": 100,
            "ThrottlingRateLimit": 2000.0
        }
    },
    "StageName": "dev",
    "StageVariables": {},
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的保护 HTTP API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStage。

update-vpc-link

以下代码示例演示了如何使用 update-vpc-link。

Amazon CLI

更新 VPC 链接

以下 update-vpc-link 示例更新 VPC 链接的名称。创建 VPC 链接后，您无法更改其安全组或子网。

```
aws apigatewayv2 update-vpc-link \
    --vpc-link-id abcd123 \
    --name MyUpdatedVpcLink
```

输出：

```
{
    "CreatedDate": "2020-04-07T00:27:47Z",
    "Name": "MyUpdatedVpcLink",
    "SecurityGroupIds": [
        "sg1234",
        "sg5678"
    ],
    "SubnetIds": [
        "subnet-aaaa",
        "subnet-bbbb"
    ],
    "Tags": {},
    "VpcLinkId": "abcd123",
    "VpcLinkStatus": "AVAILABLE",
    "VpcLinkStatusMessage": "VPC link is ready to route traffic",
    "VpcLinkVersion": "V2"
}
```

要了解更多信息，请参阅《Amazon API Gateway 开发人员指南》中的使用 HTTP API 的 VPC 链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVpcLink。

使用 Amazon CLI 的 API Gateway Management API 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 API Gateway Management API 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-connection`

以下代码示例演示了如何使用 `delete-connection`。

Amazon CLI

删除 WebSocket 连接

以下 `delete-connection` 示例断开客户端与指定 WebSocket API 的连接。

```
aws apigatewaymanagementapi delete-connection \
    --connection-id L0SM9cOFvHcCIhw= \
    --endpoint-url https://aabbccddee.execute-api.us-west-2.amazonaws.com/prod
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在后端服务中使用 @connections 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnection。

`get-connection`

以下代码示例演示了如何使用 `get-connection`。

Amazon CLI

获取有关 WebSocket 连接的信息

以下 `get-connection` 示例描述与指定 WebSocket API 的连接。

```
aws apigatewaymanagementapi get-connection \
    --connection-id L0SM9cOFvHcCIhw= \
    --endpoint-url https://aabbccddee.execute-api.us-west-2.amazonaws.com/prod
```

输出：

```
{
    "ConnectedAt": "2020-04-30T20:10:33.236Z",
    "Identity": {
        "SourceIp": "192.0.2.1"
    },
    "LastActiveAt": "2020-04-30T20:10:42.997Z"
}
```

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在后端服务中使用 @connections 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnection。

`post-to-connection`

以下代码示例演示了如何使用 `post-to-connection`。

Amazon CLI

向 WebSocket 连接发送数据

以下 `post-to-connection` 示例向连接到指定 WebSocket API 的客户端发送一条消息。

```
aws apigatewaymanagementapi post-to-connection \
    --connection-id L0SM9cOFvHcCIhw= \
    --data "Hello from API Gateway!" \
    --endpoint-url https://aabbccddee.execute-api.us-west-2.amazonaws.com/prod
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon API Gateway 开发人员指南》中的在后端服务中使用 @connections 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PostToConnection。

使用 Amazon CLI 的 App Mesh 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 App Mesh 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

create-mesh

以下代码示例演示了如何使用 create-mesh。

Amazon CLI

示例 1：创建新的服务网格

以下 create-mesh 示例创建一个服务网格。

```
aws appmesh create-mesh \
    --mesh-name app1
```

输出：

```
{
    "mesh":{
        "meshName":"app1",
        "metadata":{
            "arn":"arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
            "createdAt":1563809909.282,
            "lastUpdatedAt":1563809909.282,
            "uid":"a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version":1
        },
        "spec":{},
        "status":{
            "status":"ACTIVE"
        }
    }
}
```

示例 2：创建带有多个标签的新服务网格

以下 create-mesh 示例创建一个带有多个标签的服务网格。

```
aws appmesh create-mesh \
    --mesh-name app2 \
    --tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

输出：

```
{
    "mesh":{
        "meshName":"app2",
        "metadata":{
            "arn":"arn:aws:appmesh:us-east-1:123456789012:mesh/app2",
            "createdAt":1563822121.877,
            "lastUpdatedAt":1563822121.877,
            "uid":"a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version":1
        },
        "spec":{},
        "status":{
            "status":"ACTIVE"
        }
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的服务网格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMesh。

create-route

以下代码示例演示了如何使用 create-route。

Amazon CLI

创建新的 gRPC 路由

以下 create-route 示例使用 JSON 输入文件创建 gRPC 路由。元数据以 123 开头的 GRPC 流量将路由到名为 serviceBgrpc 的虚拟节点。如果在尝试与路由目标通信时出现特定的 gRPC、HTTP 或 TCP 故障，则会重试该路由三次。每次重试之间有 15 秒的延迟。

```
aws appmesh create-route \
```

```
    --cli-input-json file://create-route-grpc.json
```

的内容create-route-grpc.json：

```
{
    "meshName" : "apps",
    "routeName" : "grpcRoute",
    "spec" : {
       "grpcRoute" : {
          "action" : {
             "weightedTargets" : [
                {
                   "virtualNode" : "serviceBgrpc",
                   "weight" : 100
                }
             ]
          },
          "match" : {
             "metadata" : [
                {
                   "invert" : false,
                   "match" : {
                      "prefix" : "123"
                   },
                   "name" : "myMetadata"
                }
             ],
             "methodName" : "GetColor",
             "serviceName" : "com.amazonaws.services.ColorService"
          },
          "retryPolicy" : {
             "grpcRetryEvents" : [ "deadline-exceeded" ],
             "httpRetryEvents" : [ "server-error", "gateway-error" ],
             "maxRetries" : 3,
             "perRetryTimeout" : {
                "unit" : "s",
                "value" : 15
             },
             "tcpRetryEvents" : [ "connection-error" ]
          }
       },
       "priority" : 100
    },
```

```
    "virtualRouterName" : "serviceBgrpc"
}
```

输出：

```
{
    "route": {
        "meshName": "apps",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBgrpc/route/grpcRoute",
            "createdAt": 1572010806.008,
            "lastUpdatedAt": 1572010806.008,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "routeName": "grpcRoute",
        "spec": {
            "grpcRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "serviceBgrpc",
                            "weight": 100
                        }
                    ]
                },
                "match": {
                    "metadata": [
                        {
                            "invert": false,
                            "match": {
                                "prefix": "123"
                            },
                            "name": "mymetadata"
                        }
                    ],
                    "methodName": "GetColor",
                    "serviceName": "com.amazonaws.services.ColorService"
                },
                "retryPolicy": {
                    "grpcRetryEvents": [
                        "deadline-exceeded"
```

```
                    ],
                    "httpRetryEvents": [
                        "server-error",
                        "gateway-error"
                    ],
                    "maxRetries": 3,
                    "perRetryTimeout": {
                        "unit": "s",
                        "value": 15
                    },
                    "tcpRetryEvents": [
                        "connection-error"
                    ]
                }
            },
            "priority": 100
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "serviceBgrpc"
    }
}
```

创建新的 HTTP 或 HTTP/2 路由

以下 `create-route` 示例使用 JSON 输入文件创建 HTTP/2 路由。要创建 HTTP 路由，请在规范下将 http2Route 替换为 httpRoute。所有发往标头值以 123 开头的任何 URL 前缀的 HTTP/2 流量都将路由到名为 serviceBhttp2 的虚拟节点。如果在尝试与路由目标通信时出现特定的 HTTP 或 TCP 故障，则会重试该路由三次。每次重试之间有 15 秒的延迟。

```
aws appmesh create-route \
    --cli-input-json file://create-route-http2.json
```

的内容`create-route-http2.json`：

```
{
    "meshName": "apps",
    "routeName": "http2Route",
    "spec": {
        "http2Route": {
            "action": {
```

```
                "weightedTargets": [
                    {
                        "virtualNode": "serviceBhttp2",
                        "weight": 100
                    }
                ]
            },
            "match": {
                "headers": [
                    {
                        "invert": false,
                        "match": {
                            "prefix": "123"
                        },
                        "name": "clientRequestId"
                    }
                ],
                "method": "POST",
                "prefix": "/",
                "scheme": "http"
            },
            "retryPolicy": {
                "httpRetryEvents": [
                    "server-error",
                    "gateway-error"
                ],
                "maxRetries": 3,
                "perRetryTimeout": {
                    "unit": "s",
                    "value": 15
                },
                "tcpRetryEvents": [
                    "connection-error"
                ]
            }
        },
        "priority": 200
    },
    "virtualRouterName": "serviceBhttp2"
}
```

输出：

```
{
    "route": {
        "meshName": "apps",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBhttp2/route/http2Route",
            "createdAt": 1572011008.352,
            "lastUpdatedAt": 1572011008.352,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "routeName": "http2Route",
        "spec": {
            "http2Route": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "serviceBhttp2",
                            "weight": 100
                        }
                    ]
                },
                "match": {
                    "headers": [
                        {
                            "invert": false,
                            "match": {
                                "prefix": "123"
                            },
                            "name": "clientRequestId"
                        }
                    ],
                    "method": "POST",
                    "prefix": "/",
                    "scheme": "http"
                },
                "retryPolicy": {
                    "httpRetryEvents": [
                        "server-error",
                        "gateway-error"
                    ],
                    "maxRetries": 3,
                    "perRetryTimeout": {
```

```
                        "unit": "s",
                        "value": 15
                    },
                    "tcpRetryEvents": [
                        "connection-error"
                    ]
                }
            },
            "priority": 200
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "serviceBhttp2"
    }
}
```

创建新的 TCP 路由

以下 `create-route` 示例使用 JSON 输入文件创建 TCP 路由。75% 的流量路由到名为 serviceBtcp 的虚拟节点，25% 的流量路由到名为 serviceBv2tcp 的虚拟节点。为不同的目标指定不同的权重是部署应用程序新版本的有效方法。您可以调整权重，以便最终将 100% 的流量路由到具有新版本应用程序的目标。

```
aws appmesh create-route \
    --cli-input-json file://create-route-tcp.json
```

create-route-tcp.json 的内容：

```
{
    "meshName": "apps",
    "routeName": "tcpRoute",
    "spec": {
        "priority": 300,
        "tcpRoute": {
            "action": {
                "weightedTargets": [
                    {
                        "virtualNode": "serviceBtcp",
                        "weight": 75
                    },
                    {
```

```
                        "virtualNode": "serviceBv2tcp",
                        "weight": 25
                    }
                ]
            }
        }
    },
    "virtualRouterName": "serviceBtcp"
}
```

输出：

```
{
    "route": {
        "meshName": "apps",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBtcp/route/tcpRoute",
            "createdAt": 1572011436.26,
            "lastUpdatedAt": 1572011436.26,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "routeName": "tcpRoute",
        "spec": {
            "priority": 300,
            "tcpRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "serviceBtcp",
                            "weight": 75
                        },
                        {
                            "virtualNode": "serviceBv2tcp",
                            "weight": 25
                        }
                    ]
                }
            }
        },
        "status": {
            "status": "ACTIVE"
```

```
        },
        "virtualRouterName": "serviceBtcp"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoute。

create-virtual-gateway

以下代码示例演示了如何使用 create-virtual-gateway。

Amazon CLI

创建新的虚拟网关

以下 create-virtual-gateway 示例使用 JSON 输入文件创建带有使用端口 9080 的 HTTP 侦听器的虚拟网关。

```
aws appmesh create-virtual-gateway \
    --mesh-name meshName \
    --virtual-gateway-name virtualGatewayName \
    --cli-input-json file://create-virtual-gateway.json
```

的内容create-virtual-gateway.json：

```
{
    "spec": {
      "listeners": [
        {
          "portMapping": {
            "port": 9080,
            "protocol": "http"
          }
        }
      ]
    }
}
```

输出：

```
{
    "virtualGateway": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/meshName/
virtualGateway/virtualGatewayName",
            "createdAt": "2022-04-06T10:42:42.015000-05:00",
            "lastUpdatedAt": "2022-04-06T10:42:42.015000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "123456789012",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 9080,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualGatewayName": "virtualGatewayName"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVirtualGateway。

`create-virtual-node`

以下代码示例演示了如何使用 `create-virtual-node`。

Amazon CLI

示例 1：创建使用 DNS 进行发现的新虚拟节点

以下 create-virtual-node 示例使用 JSON 输入文件创建使用 DNS 进行服务发现的虚拟节点。

```
aws appmesh create-virtual-node \
    --cli-input-json file://create-virtual-node-dns.json
```

的内容create-virtual-node-dns.json：

```
{
    "meshName": "app1",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http"
                }
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceBv1.svc.cluster.local"
            }
        }
    },
    "virtualNodeName": "vnServiceBv1"
}
```

输出：

```
{
    "virtualNode": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv1",
            "createdAt": 1563810019.874,
            "lastUpdatedAt": 1563810019.874,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
```

```
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv1.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "vnServiceBv1"
    }
}
```

示例 2：创建使用 Amazon Cloud Map 进行发现的新虚拟节点

以下 `create-virtual-node` 示例使用 JSON 输入文件创建使用 Amazon Cloud Map 进行服务发现的虚拟节点。

```
aws appmesh create-virtual-node \
    --cli-input-json file://create-virtual-node-cloud-map.json
```

的内容`create-virtual-node-cloud-map.json`：

```
{
    "meshName": "app1",
    "spec": {
        "backends": [
            {
                "virtualService": {
                    "virtualServiceName": "serviceA.svc.cluster.local"
                }
            }
        ],
        "listeners": [
            {
```

```
                "portMapping": {
                    "port": 80,
                    "protocol": "http"
                }
            }
        ],
        "serviceDiscovery": {
            "awsCloudMap": {
                "attributes": [
                    {
                        "key": "Environment",
                        "value": "Testing"
                    }
                ],
                "namespaceName": "namespace1",
                "serviceName": "serviceA"
            }
        }
    },
    "virtualNodeName": "vnServiceA"
}
```

输出：

```
{
    "virtualNode": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceA",
            "createdAt": 1563810859.465,
            "lastUpdatedAt": 1563810859.465,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "backends": [
                {
                    "virtualService": {
                        "virtualServiceName": "serviceA.svc.cluster.local"
                    }
                }
            ],
```

```
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "awsCloudMap": {
                    "attributes": [
                        {
                            "key": "Environment",
                            "value": "Testing"
                        }
                    ],
                    "namespaceName": "namespace1",
                    "serviceName": "serviceA"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "vnServiceA"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVirtualNode。

create-virtual-router

以下代码示例演示了如何使用 create-virtual-router。

Amazon CLI

创建新的虚拟路由器

以下 create-virtual-router 示例使用 JSON 输入文件创建带有使用端口 80 的 HTTP 侦听器的虚拟路由器。

```
aws appmesh create-virtual-router \
    --cli-input-json file://create-virtual-router.json
```

的内容create-virtual-router.json：

```
{
    "meshName": "app1",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http"
                }
            }
        ]
    },
    "virtualRouterName": "vrServiceB"
}
```

输出：

```
{
    "virtualRouter": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
            "createdAt": 1563810546.59,
            "lastUpdatedAt": 1563810546.59,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
```

```
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟路由器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVirtualRouter。

create-virtual-service

以下代码示例演示了如何使用 create-virtual-service。

Amazon CLI

示例 1：创建带有虚拟节点提供程序的新虚拟服务

以下 create-virtual-service 示例使用 JSON 输入文件创建带有虚拟节点提供程序的虚拟服务。

```
aws appmesh create-virtual-service \
    --cli-input-json file://create-virtual-service-virtual-node.json
```

的内容create-virtual-service-virtual-node.json：

```
{
    "meshName": "app1",
    "spec": {
        "provider": {
            "virtualNode": {
                "virtualNodeName": "vnServiceA"
            }
        }
    },
    "virtualServiceName": "serviceA.svc.cluster.local"
}
```

输出：

```
{
    "virtualService": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
            "createdAt": 1563810859.474,
            "lastUpdatedAt": 1563810967.179,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "provider": {
                "virtualNode": {
                    "virtualNodeName": "vnServiceA"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualServiceName": "serviceA.svc.cluster.local"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

示例 2：创建带有虚拟路由器提供程序的新虚拟服务

以下 `create-virtual-service` 示例使用 JSON 输入文件创建带有虚拟路由器提供程序的虚拟服务。

```
aws appmesh create-virtual-service \
    --cli-input-json file://create-virtual-service-virtual-router.json
```

的内容`create-virtual-service-virtual-router.json`：

```
{
    "meshName": "app1",
    "spec": {
        "provider": {
            "virtualRouter": {
```

```
                "virtualRouterName": "vrServiceB"
            }
        }
    },
    "virtualServiceName": "serviceB.svc.cluster.local"
}
```

输出：

```
{
    "virtualService": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
            "createdAt": 1563908363.999,
            "lastUpdatedAt": 1563908363.999,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "provider": {
                "virtualRouter": {
                    "virtualRouterName": "vrServiceB"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualServiceName": "serviceB.svc.cluster.local"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的“虚拟服务<https://docs.aws.amazon.com/app-mesh/latest/userguide/virtual_services.html>”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVirtualService。

delete-mesh

以下代码示例演示了如何使用 delete-mesh。

Amazon CLI

删除服务网格

以下 delete-mesh 示例删除指定的服务网格。

```
aws appmesh delete-mesh \
    --mesh-name app1
```

输出：

```
{
    "mesh": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
            "createdAt": 1563809909.282,
            "lastUpdatedAt": 1563824981.248,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "egressFilter": {
                "type": "ALLOW_ALL"
            }
        },
        "status": {
            "status": "DELETED"
        }
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的服务网格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMesh。

delete-route

以下代码示例演示了如何使用 delete-route。

Amazon CLI

删除路由

以下 delete-route 示例删除指定路由。

```
aws appmesh delete-route \
    --mesh-name app1 \
    --virtual-router-name vrServiceB \
    --route-name toVnServiceB-weighted
```

输出：

```
{
    "route": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB/route/toVnServiceB-weighted",
            "createdAt": 1563811384.015,
            "lastUpdatedAt": 1563823915.936,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 3
        },
        "routeName": "toVnServiceB-weighted",
        "spec": {
            "httpRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "vnServiceBv1",
                            "weight": 80
                        },
                        {
                            "virtualNode": "vnServiceBv2",
                            "weight": 20
                        }
                    ]
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "DELETED"
        },
```

```
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoute。

delete-virtual-node

以下代码示例演示了如何使用 delete-virtual-node。

Amazon CLI

删除虚拟节点

以下 delete-virtual-node 示例删除指定的虚拟节点。

```
aws appmesh delete-virtual-node \
    --mesh-name app1 \
    --virtual-node-name vnServiceBv2
```

输出：

```
{
    "virtualNode": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv2",
            "createdAt": 1563810117.297,
            "lastUpdatedAt": 1563824700.678,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "backends": [],
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
```

```
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv2.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "DELETED"
        },
        "virtualNodeName": "vnServiceBv2"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVirtualNode。

delete-virtual-router

以下代码示例演示了如何使用 delete-virtual-router。

Amazon CLI

删除虚拟路由器

以下 delete-virtual-router 示例删除指定的虚拟路由器。

```
aws appmesh delete-virtual-router \
    --mesh-name app1 \
    --virtual-router-name vrServiceB
```

输出：

```
{
    "virtualRouter": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
            "createdAt": 1563810546.59,
```

```
            "lastUpdatedAt": 1563824253.467,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 3
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "DELETED"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟路由器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVirtualRouter。

delete-virtual-service

以下代码示例演示了如何使用 delete-virtual-service。

Amazon CLI

删除虚拟服务

以下 delete-virtual-service 示例删除指定的虚拟服务。

```
aws appmesh delete-virtual-service \
    --mesh-name app1 \
    --virtual-service-name serviceB.svc.cluster.local
```

输出：

```
{
    "virtualService": {
```

```
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
            "createdAt": 1563908363.999,
            "lastUpdatedAt": 1563913940.866,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 3
        },
        "spec": {},
        "status": {
            "status": "DELETED"
        },
        "virtualServiceName": "serviceB.svc.cluster.local"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVirtualService。

describe-mesh

以下代码示例演示了如何使用 describe-mesh。

Amazon CLI

描述服务网格

以下 describe-mesh 示例返回有关指定服务网格的详细信息。

```
aws appmesh describe-mesh \
    --mesh-name app1
```

输出：

```
{
    "mesh": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
            "createdAt": 1563809909.282,
            "lastUpdatedAt": 1563809909.282,
```

```
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {},
        "status": {
            "status": "ACTIVE"
        }
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的服务网格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMesh。

describe-route

以下代码示例演示了如何使用 describe-route。

Amazon CLI

描述路由

以下 describe-route 示例返回有关指定路由的详细信息。

```
aws appmesh describe-route \
    --mesh-name app1 \
    --virtual-router-name vrServiceB \
    --route-name toVnServiceB-weighted
```

输出：

```
{
    "route": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB/route/toVnServiceB-weighted",
            "createdAt": 1563811384.015,
            "lastUpdatedAt": 1563811384.015,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "routeName": "toVnServiceB-weighted",
```

```
        "spec": {
            "httpRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "vnServiceBv1",
                            "weight": 90
                        },
                        {
                            "virtualNode": "vnServiceBv2",
                            "weight": 10
                        }
                    ]
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRoute。

describe-virtual-node

以下代码示例演示了如何使用 describe-virtual-node。

Amazon CLI

描述虚拟节点

以下 describe-virtual-node 示例返回有关指定虚拟节点的详细信息。

```
aws appmesh describe-virtual-node \
    --mesh-name app1 \
    --virtual-node-name vnServiceBv1
```

输出：

```
{
    "virtualNode": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv1",
            "createdAt": 1563810019.874,
            "lastUpdatedAt": 1563810019.874,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "backends": [],
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv1.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "vnServiceBv1"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVirtualNode。

describe-virtual-router

以下代码示例演示了如何使用 describe-virtual-router。

Amazon CLI

描述虚拟路由器

以下 describe-virtual-router 示例返回有关指定虚拟路由器的详细信息。

```
aws appmesh describe-virtual-router \
    --mesh-name app1 \
    --virtual-router-name vrServiceB
```

输出：

```
{
    "virtualRouter": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
            "createdAt": 1563810546.59,
            "lastUpdatedAt": 1563810546.59,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟路由器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVirtualRouter。

describe-virtual-service

以下代码示例演示了如何使用 describe-virtual-service。

Amazon CLI

描述虚拟服务

以下 describe-virtual-service 示例返回有关指定虚拟服务的详细信息。

```
aws appmesh describe-virtual-service \
    --mesh-name app1 \
    --virtual-service-name serviceB.svc.cluster.local
```

输出：

```
{
    "virtualService": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
            "createdAt": 1563908363.999,
            "lastUpdatedAt": 1563908363.999,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "provider": {
                "virtualRouter": {
                    "virtualRouterName": "vrServiceB"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualServiceName": "serviceB.svc.cluster.local"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVirtualService。

list-meshes

以下代码示例演示了如何使用 list-meshes。

Amazon CLI

列出服务网格

以下 list-meshes 示例列出当前 Amazon 区域中的所有服务网格。

```
aws appmesh list-meshes
```

输出：

```
{
    "meshes": [
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
            "meshName": "app1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的服务网格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMeshes。

list-routes

以下代码示例演示了如何使用 list-routes。

Amazon CLI

列出路由

以下 list-routes 示例列出指定虚拟路由器的所有路由。

```
aws appmesh list-routes \
    --mesh-name app1 \
    --virtual-router-name vrServiceB
```

输出：

```
{
    "routes": [
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB/route/toVnServiceB",
            "meshName": "app1",
            "routeName": "toVnServiceB-weighted",
            "virtualRouterName": "vrServiceB"
        }
    ]
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoutes。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出分配给指定资源的所有标签。

```
aws appmesh list-tags-for-resource \
    --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1
```

输出：

```
{
    "tags": [
        {
            "key": "key1",
            "value": "value1"
        },
        {
            "key": "key2",
            "value": "value2"
        },
        {
```

```
            "key": "key3",
            "value": "value3"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-virtual-nodes

以下代码示例演示了如何使用 list-virtual-nodes。

Amazon CLI

列出虚拟节点

以下 list-virtual-nodes 示例列出指定服务网格中的所有虚拟节点。

```
aws appmesh list-virtual-nodes \
    --mesh-name app1
```

输出：

```
{
    "virtualNodes": [
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv1",
            "meshName": "app1",
            "virtualNodeName": "vnServiceBv1"
        },
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv2",
            "meshName": "app1",
            "virtualNodeName": "vnServiceBv2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVirtualNodes。

list-virtual-routers

以下代码示例演示了如何使用 list-virtual-routers。

Amazon CLI

列出虚拟路由器

以下 list-virtual-routers 示例列出指定服务网格中的所有虚拟路由器。

```
aws appmesh list-virtual-routers \
    --mesh-name app1
```

输出：

```
{
    "virtualRouters": [
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
            "meshName": "app1",
            "virtualRouterName": "vrServiceB"
        }
    ]
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟路由器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVirtualRouters。

list-virtual-services

以下代码示例演示了如何使用 list-virtual-services。

Amazon CLI

列出虚拟服务

以下 list-virtual-services 示例列出指定服务网格中的所有虚拟服务。

```
aws appmesh list-virtual-services \
    --mesh-name app1
```

输出：

```
{
    "virtualServices": [
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
            "meshName": "app1",
            "virtualServiceName": "serviceA.svc.cluster.local"
        },
        {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
            "meshName": "app1",
            "virtualServiceName": "serviceB.svc.cluster.local"
        }
    ]
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVirtualServices。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例将值为 value1 的标签 key1 添加到指定的资源。

```
aws appmesh tag-resource \
    --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1 \
    --tags key=key1,value=value1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

取消标记资源

以下 untag-resource 示例从指定资源中移除键为 key1 的标签。

```
aws appmesh untag-resource \
    --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1 \
    --tag-keys key1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-mesh

以下代码示例演示了如何使用 update-mesh。

Amazon CLI

更新服务网格

以下 update-mesh 示例使用 JSON 输入文件更新服务网格，使所有外部出口流量都能不受任何影响地通过 Envoy 代理转发。

```
aws appmesh update-mesh \
    --cli-input-json file://update-mesh.json
```

的内容update-mesh.json：

```
{
    "meshName": "app1",
    "spec": {
        "egressFilter": {
            "type": "ALLOW_ALL"
        }
```

```
    }
}
```

输出：

```
{
    "mesh": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
            "createdAt": 1563809909.282,
            "lastUpdatedAt": 1563812829.687,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "egressFilter": {
                "type": "ALLOW_ALL"
            }
        },
        "status": {
            "status": "ACTIVE"
        }
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的服务网格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMesh。

update-route

以下代码示例演示了如何使用 update-route。

Amazon CLI

更新路由

以下 update-route 示例使用 JSON 输入文件更新路由的权重。

```
aws appmesh update-route \
    --cli-input-json file://update-route-weighted.json
```

的内容update-route-weighted.json：

```
{
    "meshName": "app1",
    "routeName": "toVnServiceB-weighted",
    "spec": {
        "httpRoute": {
            "action": {
                "weightedTargets": [
                    {
                        "virtualNode": "vnServiceBv1",
                        "weight": 80
                    },
                    {
                        "virtualNode": "vnServiceBv2",
                        "weight": 20
                    }
                ]
            },
            "match": {
                "prefix": "/"
            }
        }
    },
    "virtualRouterName": "vrServiceB"
}
```

输出：

```
{
    "route": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB/route/toVnServiceB-weighted",
            "createdAt": 1563811384.015,
            "lastUpdatedAt": 1563819600.022,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "routeName": "toVnServiceB-weighted",
        "spec": {
            "httpRoute": {
```

```
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "vnServiceBv1",
                            "weight": 80
                        },
                        {
                            "virtualNode": "vnServiceBv2",
                            "weight": 20
                        }
                    ]
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoute。

update-virtual-node

以下代码示例演示了如何使用 update-virtual-node。

Amazon CLI

更新虚拟节点

以下 update-virtual-node 示例使用 JSON 输入文件向虚拟节点添加运行状况检查。

```
aws appmesh update-virtual-node \
    --cli-input-json file://update-virtual-node.json
```

的内容update-virtual-node.json：

```
{
    "clientToken": "500",
    "meshName": "app1",
    "spec": {
        "listeners": [
            {
                "healthCheck": {
                    "healthyThreshold": 5,
                    "intervalMillis": 10000,
                    "path": "/",
                    "port": 80,
                    "protocol": "http",
                    "timeoutMillis": 3000,
                    "unhealthyThreshold": 3
                },
                "portMapping": {
                    "port": 80,
                    "protocol": "http"
                }
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceBv1.svc.cluster.local"
            }
        }
    },
    "virtualNodeName": "vnServiceBv1"
}
```

输出：

```
{
    "virtualNode": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv1",
            "createdAt": 1563810019.874,
            "lastUpdatedAt": 1563819234.825,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
```

```
        "spec": {
            "listeners": [
                {
                    "healthCheck": {
                        "healthyThreshold": 5,
                        "intervalMillis": 10000,
                        "path": "/",
                        "port": 80,
                        "protocol": "http",
                        "timeoutMillis": 3000,
                        "unhealthyThreshold": 3
                    },
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv1.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "vnServiceBv1"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVirtualNode。

update-virtual-router

以下代码示例演示了如何使用 update-virtual-router。

Amazon CLI

更新虚拟路由器

以下 update-virtual-router 示例使用 JSON 输入文件更新虚拟路由器的侦听器端口。

```
aws appmesh update-virtual-router \
    --cli-input-json file://update-virtual-router.json
```

的内容update-virtual-router.json：

```
{
    "meshName": "app1",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 8080,
                    "protocol": "http"
                }
            }
        ]
    },
    "virtualRouterName": "vrServiceB"
}
```

输出：

```
{
    "virtualRouter": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
            "createdAt": 1563810546.59,
            "lastUpdatedAt": 1563819431.352,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 8080,
                        "protocol": "http"
                    }
                }
            ]
```

```
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "vrServiceB"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟路由器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVirtualRouter。

update-virtual-service

以下代码示例演示了如何使用 update-virtual-service。

Amazon CLI

更新虚拟服务

以下 update-virtual-service 示例使用 JSON 输入文件更新虚拟服务，以使用虚拟路由器提供程序。

```
aws appmesh update-virtual-service \
    --cli-input-json file://update-virtual-service.json
```

的内容update-virtual-service.json：

```
{
    "meshName": "app1",
    "spec": {
        "provider": {
            "virtualRouter": {
                "virtualRouterName": "vrServiceA"
            }
        }
    },
    "virtualServiceName": "serviceA.svc.cluster.local"
}
```

输出：

```
{
    "virtualService": {
        "meshName": "app1",
        "metadata": {
            "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
            "createdAt": 1563810859.474,
            "lastUpdatedAt": 1563820257.411,
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 3
        },
        "spec": {
            "provider": {
                "virtualRouter": {
                    "virtualRouterName": "vrServiceA"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualServiceName": "serviceA.svc.cluster.local"
    }
}
```

有关更多信息，请参阅《Amazon App Mesh 用户指南》中的虚拟服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVirtualService。

使用 Amazon CLI 的 App Runner 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 App Runner 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-custom-domain`

以下代码示例演示了如何使用 `associate-custom-domain`。

Amazon CLI

将域名和 www 子域与服务关联

以下 `associate-custom-domain` 示例将您控制的自定义域名与 App Runner 服务关联。域名是根域 `example.com`，包括特殊情况的子域 `www.example.com`。

```
aws apprunner associate-custom-domain \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "DomainName": "example.com",
    "EnableWWWSubdomain": true
}
```

输出：

```
{
    "CustomDomain": {
        "CertificateValidationRecords": [
            {
                "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",
                "Status": "PENDING_VALIDATION",
                "Type": "CNAME",
                "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-
validations.aws."
            },
            {
                "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",
                "Status": "PENDING_VALIDATION",
                "Type": "CNAME",
```

```
                "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-
validations.aws."
            }
        ],
        "DomainName": "example.com",
        "EnableWWWSubdomain": true,
        "Status": "CREATING"
    },
    "DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateCustomDomain。

create-auto-scaling-configuration

以下代码示例演示了如何使用 create-auto-scaling-configuration。

Amazon CLI

创建高可用性自动扩缩配置

以下 create-auto-scaling-configuration 示例通过将 MinSize 设置为 5 来创建针对高可用性进行优化的自动扩缩配置。使用此配置，App Runner 会尝试将您的服务实例分布在尽可能多的可用区（最多五个），具体取决于 Amazon 区域。

该调用将返回一个其他设置为默认值的 AutoScalingConfiguration 对象。在示例中，这是第一次调用创建名为 high-availability 的配置。修订版设置为 1，这是最新的修订版。

```
aws apprunner create-auto-scaling-configuration \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoScalingConfigurationName": "high-availability",
    "MinSize": 5
}
```

输出：

```
{
    "AutoScalingConfiguration": {
        "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-
availability/1/2f50e7656d7819fead0f59672e68042e",
        "AutoScalingConfigurationName": "high-availability",
        "AutoScalingConfigurationRevision": 1,
        "CreatedAt": "2020-11-03T00:29:17Z",
        "Latest": true,
        "Status": "ACTIVE",
        "MaxConcurrency": 100,
        "MaxSize": 50,
        "MinSize": 5
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAutoScalingConfiguration。

create-connection

以下代码示例演示了如何使用 create-connection。

Amazon CLI

创建 GitHub 连接

以下 create-connection 示例创建与私有 GitHub 代码存储库的连接。成功调用后的连接状态为 PENDING_HANDSHAKE。这是因为还未与提供商进行身份验证握手。使用 App Runner 控制台完成握手。

```
aws apprunner create-connection \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ConnectionName": "my-github-connection",
    "ProviderType": "GITHUB"
}
```

输出：

```
{
    "Connection": {
        "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-
github-connection",
        "ConnectionName": "my-github-connection",
        "Status": "PENDING_HANDSHAKE",
        "CreatedAt": "2020-11-03T00:32:51Z",
        "ProviderType": "GITHUB"
    }
}
```

有关更多信息，请参阅《Amazon App Runner 开发人员指南》中的管理 App Runner 连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnection。

create-service

以下代码示例演示了如何使用 create-service。

Amazon CLI

示例 1：创建源代码存储库服务

以下 create-service 示例基于 Python 源代码存储库创建 App Runner 服务。

```
aws apprunner create-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceName": "python-app",
    "SourceConfiguration": {
        "AuthenticationConfiguration": {
            "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-connection/e7656250f67242d7819feade6800f59e"
        },
        "AutoDeploymentsEnabled": true,
        "CodeRepository": {
            "RepositoryUrl": "https://github.com/my-account/python-hello",
            "SourceCodeVersion": {
                "Type": "BRANCH",
```

```
                "Value": "main"
            },
            "CodeConfiguration": {
                "ConfigurationSource": "API",
                "CodeConfigurationValues": {
                    "Runtime": "PYTHON_3",
                    "BuildCommand": "pip install -r requirements.txt",
                    "StartCommand": "python server.py",
                    "Port": "8080",
                    "RuntimeEnvironmentVariables": [
                        {
                            "NAME": "Jane"
                        }
                    ]
                }
            }
        },
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-20T19:05:25Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
```

```
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

示例 2：创建源代码存储库服务

以下 `create-service` 示例基于 Python 源代码存储库创建 App Runner 服务。

```
aws apprunner create-service \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "ServiceName": "python-app",
    "SourceConfiguration": {
        "AuthenticationConfiguration": {
```

```
            "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-connection/e7656250f67242d7819feade6800f59e"
        },
        "AutoDeploymentsEnabled": true,
        "CodeRepository": {
            "RepositoryUrl": "https://github.com/my-account/python-hello",
            "SourceCodeVersion": {
                "Type": "BRANCH",
                "Value": "main"
            },
            "CodeConfiguration": {
                "ConfigurationSource": "API",
                "CodeConfigurationValues": {
                    "Runtime": "PYTHON_3",
                    "BuildCommand": "pip install -r requirements.txt",
                    "StartCommand": "python server.py",
                    "Port": "8080",
                    "RuntimeEnvironmentVariables": [
                        {
                            "NAME": "Jane"
                        }
                    ]
                }
            }
        },
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-20T19:05:25Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
```

```
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

示例 3：创建源映像存储库服务

以下 `create-service` 示例基于存储在 Elastic Container Registry（ECR）中的映像创建 App Runner 服务。

```
aws apprunner create-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceName": "golang-container-app",
    "SourceConfiguration": {
        "AuthenticationConfiguration": {
            "AccessRoleArn": "arn:aws:iam::123456789012:role/my-ecr-role"
        },
        "AutoDeploymentsEnabled": true,
        "ImageRepository": {
            "ImageIdentifier": "123456789012.dkr.ecr.us-east-1.amazonaws.com/golang-
app:latest",
            "ImageConfiguration": {
                "Port": "8080",
                "RuntimeEnvironmentVariables": [
                    {
                        "NAME": "Jane"
                    }
                ]
            },
            "ImageRepositoryType": "ECR"
        }
    },
    "InstanceConfiguration": {
        "CPU": "1 vCPU",
        "Memory": "3 GB"
    }
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-06T23:15:30Z",
        "UpdatedAt": "2020-11-06T23:15:30Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/golang-
container-app/51728f8a20ce46d39b25398a6c8e9d1a",
        "ServiceId": "51728f8a20ce46d39b25398a6c8e9d1a",
```

```
        "ServiceName": "golang-container-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "AccessRoleArn": "arn:aws:iam::123456789012:role/my-ecr-role"
            },
            "AutoDeploymentsEnabled": true,
            "ImageRepository": {
                "ImageIdentifier": "123456789012.dkr.ecr.us-east-1.amazonaws.com/
golang-app:latest",
                "ImageConfiguration": {
                    "Port": "8080",
                    "RuntimeEnvironmentVariables": [
                        {
                            "NAME": "Jane"
                        }
                    ]
                },
                "ImageRepositoryType": "ECR"
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateService。

delete-auto-scaling-configuration

以下代码示例演示了如何使用 delete-auto-scaling-configuration。

Amazon CLI

示例 1：删除自动扩缩配置的最新有效修订版

以下 delete-auto-scaling-configuration 示例删除 App Runner 自动扩缩配置的最新有效修订版。要删除最新的有效修订版，请指定以配置名称结尾的 Amazon 资源名称（ARN），不包括修订组件。

在示例中，在此操作之前存在两个修订版。因此，修订版 2（最新）已删除。但是，它现在会显示 "Latest": false，因为删除后，它不再是最新的有效修订版。

```
aws apprunner delete-auto-scaling-configuration \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability"
}
```

输出：

```
{
    "AutoScalingConfiguration": {
        "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/2/
e76562f50d78042e819fead0f59672e6",
        "AutoScalingConfigurationName": "high-availability",
        "AutoScalingConfigurationRevision": 2,
        "CreatedAt": "2021-02-25T17:42:59Z",
        "DeletedAt": "2021-03-02T08:07:06Z",
        "Latest": false,
        "Status": "INACTIVE",
        "MaxConcurrency": 30,
        "MaxSize": 90,
        "MinSize": 5
    }
}
```

示例 2：删除自动扩缩配置的指定修订版

以下 delete-auto-scaling-configuration 示例删除 App Runner 自动扩缩配置的指定修订版。要删除指定修订版，请指定包含修订号的 ARN。

在示例中，在此操作之前存在多个修订版。该操作会删除修订版 1。

```
aws apprunner delete-auto-scaling-configuration \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/1"
}
```

输出：

```
{
    "AutoScalingConfiguration": {
        "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-
availability/1/2f50e7656d7819fead0f59672e68042e",
        "AutoScalingConfigurationName": "high-availability",
        "AutoScalingConfigurationRevision": 1,
        "CreatedAt": "2020-11-03T00:29:17Z",
        "DeletedAt": "2021-03-02T08:07:06Z",
        "Latest": false,
        "Status": "INACTIVE",
        "MaxConcurrency": 100,
        "MaxSize": 50,
        "MinSize": 5
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAutoScalingConfiguration。

delete-connection

以下代码示例演示了如何使用 delete-connection。

Amazon CLI

删除连接

以下 delete-connection 示例删除一个 App Runner 连接。成功调用后的连接状态为 DELETED。这是因为连接不再可用。

```
aws apprunner delete-connection \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-github-
connection"
}
```

输出：

```
{
    "Connection": {
        "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-
github-connection",
        "ConnectionName": "my-github-connection",
        "Status": "DELETED",
        "CreatedAt": "2020-11-03T00:32:51Z",
        "ProviderType": "GITHUB"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnection。

delete-service

以下代码示例演示了如何使用 delete-service。

Amazon CLI

删除服务

以下 delete-service 示例删除一个 App Runner 服务。

```
aws apprunner delete-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：


```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-20T19:05:25Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
```

```
            "Memory": "3 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteService。

describe-auto-scaling-configuration

以下代码示例演示了如何使用 describe-auto-scaling-configuration。

Amazon CLI

示例 1：描述自动扩缩配置的最新有效修订版

以下 describe-auto-scaling-configuration 示例获取 App Runner 自动扩缩配置的最新有效修订版的说明。要描述最新的有效修订版，请指定以配置名称结尾的 ARN，不包括修订组件。

在示例中，存在两个修订版。因此，描述的是修订版 2（最新）。生成的对象显示 "Latest": true。

```
aws apprunner describe-auto-scaling-configuration \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability"
}
```

输出：

```
{
    "AutoScalingConfiguration": {
        "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/2/
e76562f50d78042e819fead0f59672e6",
        "AutoScalingConfigurationName": "high-availability",
        "AutoScalingConfigurationRevision": 2,
        "CreatedAt": "2021-02-25T17:42:59Z",
```

```
        "Latest": true,
        "Status": "ACTIVE",
        "MaxConcurrency": 30,
        "MaxSize": 90,
        "MinSize": 5
    }
}
```

示例 2：描述自动扩缩配置的指定修订版

以下 describe-auto-scaling-configuration 示例获取 App Runner 自动扩缩配置的指定修订版的说明。要描述指定修订版，请指定包含修订号的 ARN。

在示例中，存在多个修订版，查询了修订版 1。生成的对象显示 "Latest": false。

```
aws apprunner describe-auto-scaling-configuration \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/1"
}
```

输出：

```
{
    "AutoScalingConfiguration": {
        "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-
availability/1/2f50e7656d7819fead0f59672e68042e",
        "AutoScalingConfigurationName": "high-availability",
        "AutoScalingConfigurationRevision": 1,
        "CreatedAt": "2020-11-03T00:29:17Z",
        "Latest": false,
        "Status": "ACTIVE",
        "MaxConcurrency": 100,
        "MaxSize": 50,
        "MinSize": 5
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAutoScalingConfiguration。

describe-custom-domains

以下代码示例演示了如何使用 describe-custom-domains。

Amazon CLI

获取与服务关联的自定义域名描述

以下 describe-custom-domains 示例获取与 App Runner 服务关联的自定义域名的描述和状态。

```
aws apprunner describe-custom-domains \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "DomainName": "example.com",
    "EnableWWWSubdomain": true
}
```

输出：

```
{
    "CustomDomains": [
        {
            "CertificateValidationRecords": [
                {
                    "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",
                    "Status": "PENDING_VALIDATION",
                    "Type": "CNAME",
                    "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-
validations.aws."
                },
                {
                    "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",
                    "Status": "PENDING_VALIDATION",
                    "Type": "CNAME",
```

```
                    "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-
validations.aws."
                }
            ],
            "DomainName": "example.com",
            "EnableWWWSubdomain": true,
            "Status": "PENDING_CERTIFICATE_DNS_VALIDATION"
        },
        {
            "CertificateValidationRecords": [
                {
                    "Name": "_a94f784c70d3f507c72dc28f55db2f6b.deals.example.com",
                    "Status": "SUCCESS",
                    "Type": "CNAME",
                    "Value": "_2db02504c1270c137383c6307b6834b0.bsgbmzkfwj.acm-
validations.aws."
                }
            ],
            "DomainName": "deals.example.com",
            "EnableWWWSubdomain": false,
            "Status": "ACTIVE"
        }
    ],
    "DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomDomains。

describe-service

以下代码示例演示了如何使用 `describe-service`。

Amazon CLI

描述服务

以下 `describe-service` 示例获取一个 App Runner 服务的说明。

```
aws apprunner describe-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-20T19:05:25Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
```

```
                    "Value": "main"
                }
            }
        },
        "Status": "RUNNING",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeService。

disassociate-custom-domain

以下代码示例演示了如何使用 disassociate-custom-domain。

Amazon CLI

取消域名与服务的关联

以下 disassociate-custom-domain 示例取消域 example.com 与 App Runner 服务的关联。该调用还会取消与根域名关联的子域 www.example.com 的关联。

```
aws apprunner disassociate-custom-domain \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "DomainName": "example.com"
}
```

输出：

```
{
    "CustomDomain": {
        "CertificateValidationRecords": [
            {
```

```
                "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",
                "Status": "PENDING_VALIDATION",
                "Type": "CNAME",
                "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-
validations.aws."
            },
            {
                "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",
                "Status": "PENDING_VALIDATION",
                "Type": "CNAME",
                "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-
validations.aws."
            }
        ],
        "DomainName": "example.com",
        "EnableWWWSubdomain": true,
        "Status": "DELETING"
    },
    "DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateCustomDomain。

`list-auto-scaling-configurations`

以下代码示例演示了如何使用 `list-auto-scaling-configurations`。

Amazon CLI

获取 App Runner 自动扩缩配置的分页列表

以下 `list-auto-scaling-configurations` 示例列出您 Amazon 账户中的所有 App Runner 自动扩缩配置。每个响应中最多列出五个自动扩缩配置。未指定 `AutoScalingConfigurationName` 和 `LatestOnly`。其默认值会列出所有活动配置的最新版本。

在此示例中，响应只包含两个结果，没有其他结果，因此不返回 `NextToken`。

```
aws apprunner list-auto-scaling-configurations \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "MaxResults": 5
}
```

输出：

```
{
    "AutoScalingConfigurationSummaryList": [
        {
            "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/2/
e76562f50d78042e819fead0f59672e6",
            "AutoScalingConfigurationName": "high-availability",
            "AutoScalingConfigurationRevision": 2
        },
        {
            "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/low-
cost/1/50d7804e7656fead0f59672e62f2e819",
            "AutoScalingConfigurationName": "low-cost",
            "AutoScalingConfigurationRevision": 1
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAutoScalingConfigurations。

list-connections

以下代码示例演示了如何使用 list-connections。

Amazon CLI

示例 1：列出所有连接

以下 list-connections 示例列出 Amazon 账户中的所有 App Runner 配置。

```
aws apprunner list-connections
```

输出：

```
{
    "ConnectionSummaryList": [
        {
            "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-connection",
            "ConnectionName": "my-github-connection",
            "Status": "AVAILABLE",
            "CreatedAt": "2020-11-03T00:32:51Z",
            "ProviderType": "GITHUB"
        },
        {
            "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-org-connection",
            "ConnectionName": "my-github-org-connection",
            "Status": "AVAILABLE",
            "CreatedAt": "2020-11-03T02:54:17Z",
            "ProviderType": "GITHUB"
        }
    ]
}
```

示例 2：按名称列出连接

以下 `list-connections` 示例按名称列出连接。

```
aws apprunner list-connections \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "ConnectionName": "my-github-org-connection"
}
```

输出：

```
{
    "ConnectionSummaryList": [
        {
            "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-org-connection",
            "ConnectionName": "my-github-org-connection",
```

```
            "Status": "AVAILABLE",
            "CreatedAt": "2020-11-03T02:54:17Z",
            "ProviderType": "GITHUB"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConnections。

`list-operations`

以下代码示例演示了如何使用 `list-operations`。

Amazon CLI

列出服务上发生的操作

以下 `list-operations` 示例列出迄今在 App Runner 服务上发生的所有操作。在此示例中，该服务是新的，并且只发生了单一 CREATE_SERVICE 类型的操作。

```
aws apprunner list-operations \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "OperationSummaryList": [
        {
            "EndedAt": 1606156217,
            "Id": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
            "StartedAt": 1606156014,
            "Status": "SUCCEEDED",
            "TargetArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
```

```
            "Type": "CREATE_SERVICE",
            "UpdatedAt": 1606156217
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOperations。

`list-services`

以下代码示例演示了如何使用 `list-services`。

Amazon CLI

获取 App Runner 服务的分页列表

以下 `list-services` 示例列出 Amazon 账户中的所有 App Runner 服务。每个响应中最多列出两个服务。此示例显示了第一个请求。响应包括两个结果和一个可在下一个请求中使用的令牌。如果后续响应不包含令牌，则表示所有服务都已列出。

```
aws apprunner list-services \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "MaxResults": 2
}
```

输出：

```
{
    "NextToken":
 "eyJDdXN0b21lckFjY291bnRJZCI6IjI3MDIwNTQwMjg0NSIsIlNlcnZpY2VTdGF0dXNDb2RlIjoiUFJPVklTU9OU
    "ServiceSummaryList": [
        {
            "CreatedAt": "2020-11-20T19:05:25Z",
            "UpdatedAt": "2020-11-23T12:41:37Z",
            "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
```

```
            "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
            "ServiceName": "python-app",
            "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
            "Status": "RUNNING"
        },
        {
            "CreatedAt": "2020-11-06T23:15:30Z",
            "UpdatedAt": "2020-11-23T13:21:22Z",
            "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/golang-
container-app/ab8f94cfe29a460fb8760afd2ee87555",
            "ServiceId": "ab8f94cfe29a460fb8760afd2ee87555",
            "ServiceName": "golang-container-app",
            "ServiceUrl": "e2m8rrrx33.us-east-1.awsapprunner.com",
            "Status": "RUNNING"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出与 App Runner 服务关联的标签

以下 `list-tags-for-resource` 示例列出与 App Runner 服务关联的所有标签。

```
aws apprunner list-tags-for-resource \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Retail"
        },
        {
            "Key": "CustomerId",
            "Value": "56439872357912"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

pause-service

以下代码示例演示了如何使用 pause-service。

Amazon CLI

暂停服务

以下 pause-service 示例暂停一个 App Runner 服务。

```
aws apprunner pause-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
```

```
        "UpdatedAt": "2020-11-23T12:41:37Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PauseService。

resume-service

以下代码示例演示了如何使用 resume-service。

Amazon CLI

恢复服务

以下 resume-service 示例恢复一个 App Runner 服务。

```
aws apprunner resume-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-23T12:41:37Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
```

```
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "3 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResumeService。

start-deployment

以下代码示例演示了如何使用 start-deployment。

Amazon CLI

启动手动部署

以下 start-deployment 示例对 App Runner 服务执行手动部署。

```
aws apprunner start-deployment \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

输出：

```
{
    "OperationId": "853a7d5b-fc9f-4730-831b-fd8037ab832a"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDeployment。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向 App Runner 服务添加标签

以下 tag-resource 示例向一个 App Runner 服务添加两个标签。

```
aws apprunner tag-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "Tags": [
        {
            "Key": "Department",
            "Value": "Retail"
        },
        {
            "Key": "CustomerId",
            "Value": "56439872357912"
        }
    ]
```

```
}
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从 App Runner 服务中移除标签

以下 untag-resource 示例从 App Runner 服务中移除两个标签。

```
aws apprunner untag-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "TagKeys": [
        "Department",
        "CustomerId"
    ]
}
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-service

以下代码示例演示了如何使用 update-service。

Amazon CLI

更新内存大小

以下 update-service 示例将 App Runner 服务实例（扩缩单位）的内存大小更新为 2048 MiB。

调用成功后，App Runner 将启动异步更新进程。调用返回的 Service 结构反映了此调用应用的新内存值。

```
aws apprunner update-service \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "InstanceConfiguration": {
        "Memory": "4 GB"
    }
}
```

输出：

```
{
    "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
    "Service": {
        "CreatedAt": "2020-11-20T19:05:25Z",
        "UpdatedAt": "2020-11-23T12:41:37Z",
        "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
        "ServiceName": "python-app",
        "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
        "SourceConfiguration": {
            "AuthenticationConfiguration": {
                "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
            },
            "AutoDeploymentsEnabled": true,
            "CodeRepository": {
                "CodeConfiguration": {
                    "CodeConfigurationValues": {
                        "BuildCommand": "pip install -r requirements.txt",
                        "Port": "8080",
                        "Runtime": "PYTHON_3",
                        "RuntimeEnvironmentVariables": [
```

```
                            {
                                "NAME": "Jane"
                            }
                        ],
                        "StartCommand": "python server.py"
                    },
                    "ConfigurationSource": "Api"
                },
                "RepositoryUrl": "https://github.com/my-account/python-hello",
                "SourceCodeVersion": {
                    "Type": "BRANCH",
                    "Value": "main"
                }
            }
        },
        "Status": "OPERATION_IN_PROGRESS",
        "InstanceConfiguration": {
            "CPU": "1 vCPU",
            "Memory": "4 GB"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateService。

Amazon AppConfig使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon AppConfig 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-application`

以下代码示例演示了如何使用 `create-application`。

Amazon CLI

创建应用程序

以下 `create-application` 示例在 Amazon AppConfig 中创建一个应用程序。

```
aws appconfig create-application \
    --name "example-application" \
    --description "An application used for creating an example."
```

输出：

```
{
    "Description": "An application used for creating an example.",
    "Id": "339ohji",
    "Name": "example-application"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApplication。

`create-configuration-profile`

以下代码示例演示了如何使用 `create-configuration-profile`。

Amazon CLI

创建配置文件

以下 `create-configuration-profile` 示例使用存储在 Parameter Store（Systems Manager 的一项功能）中的配置创建配置文件。

```
aws appconfig create-configuration-profile \
    --application-id "339ohji" \
```

```
    --name "Example-Configuration-Profile" \
    --location-uri "ssm-parameter://Example-Parameter" \
    --retrieval-role-arn "arn:aws:iam::111122223333:role/Example-App-Config-Role"
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Description": null,
    "Id": "ur8hx2f",
    "LocationUri": "ssm-parameter://Example-Parameter",
    "Name": "Example-Configuration-Profile",
    "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role",
    "Type": null,
    "Validators": null
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConfigurationProfile。

`create-environment`

以下代码示例演示了如何使用 `create-environment`。

Amazon CLI

创建环境

以下 `create-environment` 示例使用您使用 create-application 创建的应用程序创建一个名为 Example-Environment 的 Amazon AppConfig 环境。

```
aws appconfig create-environment \
    --application-id "339ohji" \
    --name "Example-Environment"
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Description": null,
    "Id": "54j1r29",
```

```
    "Monitors": null,
    "Name": "Example-Environment",
    "State": "ReadyForDeployment"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 2：创建环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEnvironment。

create-extension-association

以下代码示例演示了如何使用 create-extension-association。

Amazon CLI

创建扩展关联

以下 create-extension-association 示例在 Amazon AppConfig 中创建一个新的扩展关联。

```
aws appconfig create-extension-association \
    --region us-west-2 \
    --extension-identifier S3-backup-extension \
    --resource-identifier "arn:aws:appconfig:us-west-2:123456789012:application/
Finance" \
    --parameters S3bucket=FinanceConfigurationBackup
```

输出：

```
{
  "Id": "a1b2c3d4",
  "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
  "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
  "Parameters": {
    "S3bucket": "FinanceConfigurationBackup"
  },
  "ExtensionVersionNumber": 1
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateExtensionAssociation。

create-extension

以下代码示例演示了如何使用 create-extension。

Amazon CLI

创建扩展

以下 create-extension 示例在 Amazon AppConfig 中创建一个新的扩展。

```
aws appconfig create-extension \
    --region us-west-2 \
    --name S3-backup-extension \
    --
actions PRE_CREATE_HOSTED_CONFIGURATION_VERSION=[{Name=S3backup,Uri=arn:aws:lambda:us-
west-2:123456789012:function:s3backupfunction,RoleArn=arn:aws:iam::123456789012:role/
appconfigextensionrole}] \
    --parameters S3bucket={Required=true}
```

输出：

```
{
    "Id": "1A2B3C4D",
    "Name": "S3-backup-extension",
    "VersionNumber": 1,
    "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1",
    "Actions": {
        "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
            {
                "Name": "S3backup",
                "Uri": "arn:aws:lambda:us-
west-2:123456789012:function:s3backupfunction",
                "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"
            }
        ]
    },
    "Parameters": {
        "S3bucket": {
            "Required": true
        }
    }
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateExtension。

`create-hosted-configuration-version`

以下代码示例演示了如何使用 `create-hosted-configuration-version`。

Amazon CLI

创建托管配置版本

以下 `create-hosted-configuration-version` 示例在 Amazon AppConfig 托管配置存储中创建一个新配置。必须先将配置内容转换为 base64。

```
aws appconfig create-hosted-configuration-version \
    --application-id "339ohji" \
    --configuration-profile-id "ur8hx2f" \
    --
content eyAiTmFtZSI6ICJFeGFtcGxlQXBwbGljYXRpb24iLCAiSWQiOiBFeGFtcGxlSUQsICJSYW5rIjogNyB9
 \
    --content-type "application/json" \
    configuration_version_output_file
```

的内容`configuration_version_output_file`：

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

输出：

```
{
    "ApplicationId": "339ohji",
    "ConfigurationProfileId": "ur8hx2f",
    "VersionNumber": "1",
    "ContentType": "application/json"
}
```

有关更多信息，请参阅《Amazon Appconfig 用户指南》中的关于 Amazon AppConfig 托管配置存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHostedConfigurationVersion。

delete-application

以下代码示例演示了如何使用 delete-application。

Amazon CLI

删除 应用程序

以下 delete-application 示例删除指定的应用程序。

```
aws appconfig delete-application \
--application-id 339ohji
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApplication。

delete-configuration-profile

以下代码示例演示了如何使用 delete-configuration-profile。

Amazon CLI

删除配置文件

以下 delete-configuration-profile 示例删除指定的配置文件。

```
aws appconfig delete-configuration-profile \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConfigurationProfile。

delete-deployment-strategy

以下代码示例演示了如何使用 delete-deployment-strategy。

Amazon CLI

删除部署策略

以下 delete-deployment-strategy 示例删除指定的部署策略。

```
aws appconfig delete-deployment-strategy \
    --deployment-strategy-id 1225qzk
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 4：创建部署策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeploymentStrategy。

delete-environment

以下代码示例演示了如何使用 delete-environment。

Amazon CLI

删除环境

以下 delete-environment 示例删除指定的应用程序环境。

```
aws appconfig delete-environment \
    --application-id 339ohji \
    --environment-id 54j1r29
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 2：创建环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEnvironment。

delete-extension-association

以下代码示例演示了如何使用 delete-extension-association。

Amazon CLI

删除扩展关联

以下 delete-extension-association 示例删除 Amazon AppConfig 中的扩展关联。

```
aws appconfig delete-extension-association \
    --region us-west-2 \
    --extension-association-id a1b2c3d4
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteExtensionAssociation。

delete-extension

以下代码示例演示了如何使用 delete-extension。

Amazon CLI

删除扩展

以下 delete-extension 示例删除 Amazon AppConfig 中的扩展。

```
aws appconfig delete-extension \
    --region us-west-2 \
    --extension-identifier S3-backup-extension
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteExtension。

delete-hosted-configuration-version

以下代码示例演示了如何使用 delete-hosted-configuration-version。

Amazon CLI

删除托管配置版本

以下 delete-hosted-configuration-version 示例删除 Amazon AppConfig 托管配置存储中的托管配置版本。

```
aws appconfig delete-hosted-configuration-version \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f \
    --version-number 1
```

输出：此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHostedConfigurationVersion。

get-application

以下代码示例演示了如何使用 get-application。

Amazon CLI

列出应用程序的详细信息

以下 get-application 示例列出指定应用程序的详细信息。

```
aws appconfig get-application \
    --application-id 339ohji
```

输出：

```
{
    "Description": "An application used for creating an example.",
    "Id": "339ohji",
    "Name": "example-application"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的 Amazon AppConfig 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApplication。

get-configuration-profile

以下代码示例演示了如何使用 get-configuration-profile。

Amazon CLI

检索配置文件的详细信息

以下 get-configuration-profile 示例返回指定配置文件的详细信息。

```
aws appconfig get-configuration-profile \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Id": "ur8hx2f",
    "Name": "Example-Configuration-Profile",
    "LocationUri": "ssm-parameter://Example-Parameter",
    "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConfigurationProfile。

get-configuration

以下代码示例演示了如何使用 get-configuration。

Amazon CLI

检索配置的详细信息

以下 get-configuration 示例返回示例应用程序的配置详细信息。在后续调用 get-configuration 时，仅在版本已更改时使用 client-configuration-version 参数更新应用程序的配置。仅在版本已更改时才更新配置，可以避免因调用 get-configuration 而产生额外费用。

```
aws appconfig get-configuration \
    --application "example-application" \
    --environment "Example-Environment" \
    --configuration "Example-Configuration-Profile" \
    --client-id "test-id" \
```

```
    configuration-output-file
```

的内容configuration-output-file：

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

输出：

```
{
    "ConfigurationVersion": "1",
    "ContentType": "application/json"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 6：接收配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConfiguration。

get-deployment-strategy

以下代码示例演示了如何使用 get-deployment-strategy。

Amazon CLI

检索部署策略的详细信息

以下 get-deployment-strategy 示例列出指定部署策略的详细信息。

```
aws appconfig get-deployment-strategy \
    --deployment-strategy-id 1225qzk
```

输出：

```
{
    "Id": "1225qzk",
    "Name": "Example-Deployment",
    "DeploymentDurationInMinutes": 15,
    "GrowthType": "LINEAR",
    "GrowthFactor": 25.0,
    "FinalBakeTimeInMinutes": 0,
    "ReplicateTo": "SSM_DOCUMENT"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 4：创建部署策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentStrategy。

get-deployment

以下代码示例演示了如何使用 get-deployment。

Amazon CLI

检索部署的详细信息

以下 get-deployment 示例列出在指定环境和部署中部署到应用程序的详细信息。

```
aws appconfig get-deployment \
    --application-id 339ohji \
    --environment-id 54j1r29 \
    --deployment-number 1
```

输出：

```
{
    "ApplicationId": "339ohji",
    "EnvironmentId": "54j1r29",
    "DeploymentStrategyId": "1225qzk",
    "ConfigurationProfileId": "ur8hx2f",
    "DeploymentNumber": 1,
    "ConfigurationName": "Example-Configuration-Profile",
    "ConfigurationLocationUri": "ssm-parameter://Example-Parameter",
    "ConfigurationVersion": "1",
    "DeploymentDurationInMinutes": 15,
    "GrowthType": "LINEAR",
    "GrowthFactor": 25.0,
    "FinalBakeTimeInMinutes": 0,
    "State": "COMPLETE",
    "EventLog": [
        {
            "EventType": "DEPLOYMENT_COMPLETED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Deployment completed",
            "OccurredAt": "2021-09-17T21:59:03.888000+00:00"
        },
        {
```

```
            "EventType": "BAKE_TIME_STARTED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Deployment bake time started",
            "OccurredAt": "2021-09-17T21:58:57.722000+00:00"
        },
        {
            "EventType": "PERCENTAGE_UPDATED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Configuration available to 100.00% of clients",
            "OccurredAt": "2021-09-17T21:55:56.816000+00:00"
        },
        {
            "EventType": "PERCENTAGE_UPDATED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Configuration available to 75.00% of clients",
            "OccurredAt": "2021-09-17T21:52:56.567000+00:00"
        },
        {
            "EventType": "PERCENTAGE_UPDATED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Configuration available to 50.00% of clients",
            "OccurredAt": "2021-09-17T21:49:55.737000+00:00"
        },
        {
            "EventType": "PERCENTAGE_UPDATED",
            "TriggeredBy": "APPCONFIG",
            "Description": "Configuration available to 25.00% of clients",
            "OccurredAt": "2021-09-17T21:46:55.187000+00:00"
        },
        {
            "EventType": "DEPLOYMENT_STARTED",
            "TriggeredBy": "USER",
            "Description": "Deployment started",
            "OccurredAt": "2021-09-17T21:43:54.205000+00:00"
        }
    ],
    "PercentageComplete": 100.0,
    "StartedAt": "2021-09-17T21:43:54.205000+00:00",
    "CompletedAt": "2021-09-17T21:59:03.888000+00:00"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 5：部署配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployment。

get-environment

以下代码示例演示了如何使用 get-environment。

Amazon CLI

检索环境的详细信息

以下 get-environment 示例返回指定环境的详细信息和状态。

```
aws appconfig get-environment \
    --application-id 339ohji \
    --environment-id 54j1r29
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Id": "54j1r29",
    "Name": "Example-Environment",
    "State": "ReadyForDeployment"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 2：创建环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEnvironment。

get-extension-association

以下代码示例演示了如何使用 get-extension-association。

Amazon CLI

获取扩展关联的详细信息

以下 get-extension-association 示例显示有关扩展关联的信息。

```
aws appconfig get-extension-association \
    --region us-west-2 \
    --extension-association-id a1b2c3d4
```

输出：

```
{
    "Id": "a1b2c3d4",
    "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
    "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
    "Parameters": {
        "S3bucket": "FinanceConfigurationBackup"
    },
    "ExtensionVersionNumber": 1
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExtensionAssociation。

get-extension

以下代码示例演示了如何使用 get-extension。

Amazon CLI

获取扩展的详细信息

以下 get-extension 示例显示有关扩展的信息。

```
aws appconfig get-extension \
    --region us-west-2 \
    --extension-identifier S3-backup-extension
```

输出：

```
{
    "Id": "1A2B3C4D",
    "Name": "S3-backup-extension",
    "VersionNumber": 1,
    "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
    "Actions": {
        "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
            {
                "Name": "S3backup",
```

```
                "Uri": "arn:aws:lambda:us-
west-2:123456789012:function:S3backupfunction",
                "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"
            }
        ]
    },
    "Parameters": {
        "S3bucket": {
            "Required": true
        }
    }
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExtension。

get-hosted-configuration-version

以下代码示例演示了如何使用 get-hosted-configuration-version。

Amazon CLI

检索托管配置的详细信息

以下 get-hosted-configuration-version 示例检索 Amazon AppConfig 托管配置的配置详细信息。

```
aws appconfig get-hosted-configuration-version \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f \
    --version-number 1 \
    hosted-configuration-version-output
```

的内容hosted-configuration-version-output：

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

输出：

```
{
    "ApplicationId": "339ohji",
```

```
    "ConfigurationProfileId": "ur8hx2f",
    "VersionNumber": "1",
    "ContentType": "application/json"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的关于 Amazon AppConfig 托管配置存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetHostedConfigurationVersion。

`list-applications`

以下代码示例演示了如何使用 `list-applications`。

Amazon CLI

列出可用的应用程序

以下 `list-applications` 示例列出您 Amazon 账户中可用的应用程序。

```
aws appconfig list-applications
```

输出：

```
{
    "Items": [
        {
            "Id": "339ohji",
            "Name": "test-application",
            "Description": "An application used for creating an example."
        },
        {
            "Id": "rwalwu7",
            "Name": "Test-Application"
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListApplications。

list-configuration-profiles

以下代码示例演示了如何使用 list-configuration-profiles。

Amazon CLI

列出可用的配置文件

以下 list-configuration-profiles 示例列出指定应用程序的可用配置文件。

```
aws appconfig list-configuration-profiles \
    --application-id 339ohji
```

输出：

```
{
    "Items": [
        {
            "ApplicationId": "339ohji",
            "Id": "ur8hx2f",
            "Name": "Example-Configuration-Profile",
            "LocationUri": "ssm-parameter://Example-Parameter"
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConfigurationProfiles。

list-deployment-strategies

以下代码示例演示了如何使用 list-deployment-strategies。

Amazon CLI

列出可用的部署策略

以下 list-deployment-strategies 示例列出您 Amazon 账户中的可用部署策略。

```
aws appconfig list-deployment-strategies
```

输出：

```
{
    "Items": [
        {
            "Id": "1225qzk",
            "Name": "Example-Deployment",
            "DeploymentDurationInMinutes": 15,
            "GrowthType": "LINEAR",
            "GrowthFactor": 25.0,
            "FinalBakeTimeInMinutes": 0,
            "ReplicateTo": "SSM_DOCUMENT"
        },
        {
            "Id": "AppConfig.AllAtOnce",
            "Name": "AppConfig.AllAtOnce",
            "Description": "Quick",
            "DeploymentDurationInMinutes": 0,
            "GrowthType": "LINEAR",
            "GrowthFactor": 100.0,
            "FinalBakeTimeInMinutes": 10,
            "ReplicateTo": "NONE"
        },
        {
            "Id": "AppConfig.Linear50PercentEvery30Seconds",
            "Name": "AppConfig.Linear50PercentEvery30Seconds",
            "Description": "Test/Demo",
            "DeploymentDurationInMinutes": 1,
            "GrowthType": "LINEAR",
            "GrowthFactor": 50.0,
            "FinalBakeTimeInMinutes": 1,
            "ReplicateTo": "NONE"
        },
        {
            "Id": "AppConfig.Canary10Percent20Minutes",
            "Name": "AppConfig.Canary10Percent20Minutes",
            "Description": "AWS Recommended",
            "DeploymentDurationInMinutes": 20,
            "GrowthType": "EXPONENTIAL",
            "GrowthFactor": 10.0,
            "FinalBakeTimeInMinutes": 10,
            "ReplicateTo": "NONE"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 4：创建部署策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeploymentStrategies。

`list-deployments`

以下代码示例演示了如何使用 `list-deployments`。

Amazon CLI

列出可用的部署

以下 `list-deployments` 示例列出您 Amazon 账户中针对指定应用程序和环境的可用部署。

```
aws appconfig list-deployments \
    --application-id 339ohji \
    --environment-id 54j1r29
```

输出：

```
{
    "Items": [
        {
            "DeploymentNumber": 1,
            "ConfigurationName": "Example-Configuration-Profile",
            "ConfigurationVersion": "1",
            "DeploymentDurationInMinutes": 15,
            "GrowthType": "LINEAR",
            "GrowthFactor": 25.0,
            "FinalBakeTimeInMinutes": 0,
            "State": "COMPLETE",
            "PercentageComplete": 100.0,
            "StartedAt": "2021-09-17T21:43:54.205000+00:00",
            "CompletedAt": "2021-09-17T21:59:03.888000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 5：部署配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeployments。

list-environments

以下代码示例演示了如何使用 list-environments。

Amazon CLI

列出可用的环境

以下 list-environments 示例列出您 Amazon 账户中针对指定应用程序的可用环境。

```
aws appconfig list-environments \
    --application-id 339ohji
```

输出：

```
{
    "Items": [
        {
            "ApplicationId": "339ohji",
            "Id": "54j1r29",
            "Name": "Example-Environment",
            "State": "ReadyForDeployment"
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 2：创建环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEnvironments。

list-extension-associations

以下代码示例演示了如何使用 list-extension-associations。

Amazon CLI

列出您 Amazon 账户中 Amazon 区域的所有 Amazon AppConfig 扩展关联

以下 list-extension-associations 示例列出特定 Amazon 区域中当前 Amazon 账户的所有 Amazon AppConfig 扩展关联。

```
aws appconfig list-extension-associations \
    --region us-west-2
```

输出：

```
{
    "Items": [
        {
            "Id": "a1b2c3d4",
            "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-
backup-extension/1",
            "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/
Finance"
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListExtensionAssociations。

`list-extensions`

以下代码示例演示了如何使用 `list-extensions`。

Amazon CLI

列出您 Amazon 账户中 Amazon 区域的所有 Amazon AppConfig 扩展

以下 `list-extensions` 示例列出特定 Amazon 区域中当前 Amazon 账户的所有 Amazon AppConfig 扩展。该命令返回自定义扩展和 Amazon 编写的扩展。

```
aws appconfig list-extensions \
    --region us-west-2
```

输出：

```
{
    "Items": [
        {
            "Id": "1A2B3C4D",
            "Name": "S3-backup-extension",
            "VersionNumber": 1,
            "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1"
        },
        {
```

```
            "Id": "AWS.AppConfig.FeatureFlags",
            "Name": "AppConfig Feature Flags Helper",
            "VersionNumber": 1,
            "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.FeatureFlags/1",
            "Description": "Validates AppConfig feature flag data automatically
 against a JSON schema that includes structure and constraints. Also transforms
 feature flag data prior to sending to the client. This extension is automatically
 associated to configuration profiles with type \"AWS.AppConfig.FeatureFlags\"."
        },
        {
            "Id": "AWS.AppConfig.JiraIntegration",
            "Name": "AppConfig integration with Atlassian Jira",
            "VersionNumber": 1,
            "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.JiraIntegration/1",
            "Description": "Exports feature flag data from AWS AppConfig into
 Jira. The lifecycle of each feature flag in AppConfig is tracked in Jira as an
 individual issue. Customers can see in Jira when flags are updated, turned on or
 off. Works in conjunction with the AppConfig app in the Atlassian Marketplace and
 is automatically associated to configuration profiles configured within that app."
        },
        {
            "Id": "AWS.AppConfig.DeploymentNotificationsToEventBridge",
            "Name": "AppConfig deployment events to Amazon EventBridge",
            "VersionNumber": 1,
            "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.DeploymentNotificationsToEventBridge/1",
            "Description": "Sends events to Amazon EventBridge when a deployment
 of configuration data in AppConfig is started, completed, or rolled back. Can
 be associated to the following resources in AppConfig: Application, Environment,
 Configuration Profile."
        },
        {
            "Id": "AWS.AppConfig.DeploymentNotificationsToSqs",
            "Name": "AppConfig deployment events to Amazon SQS",
            "VersionNumber": 1,
            "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.DeploymentNotificationsToSqs/1",
            "Description": "Sends messages to the configured Amazon SQS queue when
 a deployment of configuration data in AppConfig is started, completed, or rolled
 back. Can be associated to the following resources in AppConfig: Application,
 Environment, Configuration Profile."
        },
```

```
        {
            "Id": "AWS.AppConfig.DeploymentNotificationsToSns",
            "Name": "AppConfig deployment events to Amazon SNS",
            "VersionNumber": 1,
            "Description": "Sends events to the configured Amazon SNS topic when
 a deployment of configuration data in AppConfig is started, completed, or rolled
 back. Can be associated to the following resources in AppConfig: Application,
 Environment, Configuration Profile."
        }
    ]
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListExtensions。

`list-hosted-configuration-versions`

以下代码示例演示了如何使用 `list-hosted-configuration-versions`。

Amazon CLI

列出可用的托管配置版本

以下 `list-hosted-configuration-versions` 示例列出 Amazon AppConfig 托管配置存储中针对指定应用程序和配置文件的托管配置版本。

```
aws appconfig list-hosted-configuration-versions \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f
```

输出：

```
{
    "Items": [
        {
            "ApplicationId": "339ohji",
            "ConfigurationProfileId": "ur8hx2f",
            "VersionNumber": 1,
            "ContentType": "application/json"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的关于 Amazon AppConfig 托管配置存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHostedConfigurationVersions。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出应用程序的标签

以下 `list-tags-for-resource` 示例列出指定应用程序的标签。

```
aws appconfig list-tags-for-resource \
    --resource-arn arn:aws:appconfig:us-east-1:682428703967:application/339ohji
```

输出：

```
{
    "Tags": {
        "group1": "1"
    }
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`start-deployment`

以下代码示例演示了如何使用 `start-deployment`。

Amazon CLI

开始配置部署

以下 start-deployment 示例使用指定的环境、部署策略和配置文件，开始对应用程序进行部署。

```
aws appconfig start-deployment \
    --application-id 339ohji \
    --environment-id 54j1r29 \
    --deployment-strategy-id 1225qzk \
    --configuration-profile-id ur8hx2f \
    --configuration-version 1
```

输出：

```
{
    "ApplicationId": "339ohji",
    "EnvironmentId": "54j1r29",
    "DeploymentStrategyId": "1225qzk",
    "ConfigurationProfileId": "ur8hx2f",
    "DeploymentNumber": 1,
    "ConfigurationName": "Example-Configuration-Profile",
    "ConfigurationLocationUri": "ssm-parameter://Example-Parameter",
    "ConfigurationVersion": "1",
    "DeploymentDurationInMinutes": 15,
    "GrowthType": "LINEAR",
    "GrowthFactor": 25.0,
    "FinalBakeTimeInMinutes": 0,
    "State": "DEPLOYING",
    "EventLog": [
        {
            "EventType": "DEPLOYMENT_STARTED",
            "TriggeredBy": "USER",
            "Description": "Deployment started",
            "OccurredAt": "2021-09-17T21:43:54.205000+00:00"
        }
    ],
    "PercentageComplete": 0.0,
    "StartedAt": "2021-09-17T21:43:54.205000+00:00"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 5：部署配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDeployment。

stop-deployment

以下代码示例演示了如何使用 stop-deployment。

Amazon CLI

停止配置部署

以下 stop-deployment 示例停止向指定环境部署应用程序配置。

```
aws appconfig stop-deployment \
    --application-id 339ohji \
    --environment-id 54j1r29 \
    --deployment-number 2
```

输出：

```
{
    "DeploymentNumber": 0,
    "DeploymentDurationInMinutes": 0,
    "GrowthFactor": 0.0,
    "FinalBakeTimeInMinutes": 0,
    "PercentageComplete": 0.0
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 5：部署配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDeployment。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记应用程序

以下 tag-resource 示例标记应用程序资源。

```
aws appconfig tag-resource \
    --resource-arn arn:aws:appconfig:us-east-1:682428703967:application/339ohji \
```

```
    --tags '{"group1" : "1"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除应用程序的标签

以下 untag-resource 示例移除指定应用程序的“group1”标签。

```
aws appconfig untag-resource \
    --resource-arn arn:aws:appconfig:us-east-1:111122223333:application/339ohji \
    --tag-keys '["group1"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-application

以下代码示例演示了如何使用 update-application。

Amazon CLI

更新应用程序

以下 update-application 示例更新指定应用程序的名称。

```
aws appconfig update-application \
```

```
    --application-id 339ohji \
    --name "Example-Application"
```

输出：

```
{
    "Id": "339ohji",
    "Name": "Example-Application",
    "Description": "An application used for creating an example."
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 1：创建 Amazon AppConfig 应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApplication。

update-configuration-profile

以下代码示例演示了如何使用 update-configuration-profile。

Amazon CLI

更新配置文件

以下 update-configuration-profile 示例更新指定配置文件的说明。

```
aws appconfig update-configuration-profile \
    --application-id 339ohji \
    --configuration-profile-id ur8hx2f \
    --description "Configuration profile used for examples."
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Id": "ur8hx2f",
    "Name": "Example-Configuration-Profile",
    "Description": "Configuration profile used for examples.",
    "LocationUri": "ssm-parameter://Example-Parameter",
    "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConfigurationProfile。

update-deployment-strategy

以下代码示例演示了如何使用 update-deployment-strategy。

Amazon CLI

更新部署策略

以下 update-deployment-strategy 示例将指定部署策略中的最终烘焙时间更新为 20 分钟。

```
aws appconfig update-deployment-strategy \
    --deployment-strategy-id 1225qzk \
    --final-bake-time-in-minutes 20
```

输出：

```
{
    "Id": "1225qzk",
    "Name": "Example-Deployment",
    "DeploymentDurationInMinutes": 15,
    "GrowthType": "LINEAR",
    "GrowthFactor": 25.0,
    "FinalBakeTimeInMinutes": 20,
    "ReplicateTo": "SSM_DOCUMENT"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 4：创建部署策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeploymentStrategy。

update-environment

以下代码示例演示了如何使用 update-environment。

Amazon CLI

更新环境

以下 update-environment 示例更新环境的说明。

```
aws appconfig update-environment \
    --application-id 339ohji \
    --environment-id 54j1r29 \
    --description "An environment for examples."
```

输出：

```
{
    "ApplicationId": "339ohji",
    "Id": "54j1r29",
    "Name": "Example-Environment",
    "Description": "An environment for examples.",
    "State": "RolledBack"
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 2：创建环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEnvironment。

update-extension-association

以下代码示例演示了如何使用 update-extension-association。

Amazon CLI

更新 Amazon AppConfig 扩展关联

以下 update-extension-association 示例向 Amazon AppConfig 中的扩展关联添加一个新参数值。

```
aws appconfig update-extension-association \
    --region us-west-2 \
    --extension-association-id a1b2c3d4 \
    --parameters S3bucket=FinanceMobileApp
```

输出：

```
{
    "Id": "a1b2c3d4",
```

```
    "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
    "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
    "Parameters": {
        "S3bucket": "FinanceMobileApp"
    },
    "ExtensionVersionNumber": 1
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateExtensionAssociation。

update-extension

以下代码示例演示了如何使用 update-extension。

Amazon CLI

更新 Amazon AppConfig 扩展

以下 update-extension 示例向 Amazon AppConfig 中的扩展添加一个附加参数键。

```
aws appconfig update-extension \
    --region us-west-2 \
    --extension-identifier S3-backup-extension \
    --parameters S3bucket={Required=true},CampaignID={Required=false}
```

输出：

```
{
    "Id": "1A2B3C4D",
    "Name": "S3-backup-extension",
    "VersionNumber": 1,
    "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1",
    "Actions": {
        "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
            {
                "Name": "S3backup",
                "Uri": "arn:aws:lambda:us-
west-2:123456789012:function:S3backupfunction",
                "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"
```

```
            }
        ]
    },
    "Parameters": {
        "CampaignID": {
            "Required": false
        },
        "S3bucket": {
            "Required": true
        }
    }
}
```

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的使用 Amazon AppConfig 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateExtension。

`validate-configuration`

以下代码示例演示了如何使用 `validate-configuration`。

Amazon CLI

验证配置

以下 `validate-configuration` 示例使用配置文件中的验证器来验证配置。

```
aws appconfig validate-configuration \
    --application-id abc1234 \
    --configuration-profile-id ur8hx2f \
    --configuration-version 1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon AppConfig 用户指南》中的步骤 3：创建配置和配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateConfiguration。

使用 Amazon CLI 的 Application Auto Scaling 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Application Auto Scaling 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-scaling-policy`

以下代码示例演示了如何使用 `delete-scaling-policy`。

Amazon CLI

删除扩展策略

此示例删除在默认集群中运行的 Amazon ECS 服务 web-app 的扩展策略。

命令:

```
aws application-autoscaling delete-scaling-policy --policy-name web-app-cpu-lt-25 --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app --service-namespace ecs
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScalingPolicy。

`delete-scheduled-action`

以下代码示例演示了如何使用 `delete-scheduled-action`。

Amazon CLI

删除计划操作

以下 `delete-scheduled-action` 示例从指定的 Amazon AppStream 2.0 实例集中删除指定的计划操作：

```
aws application-autoscaling delete-scheduled-action \
```

```
    --service-namespace appstream \
    --scalable-dimension appstream:fleet:DesiredCapacity \
    --resource-id fleet/sample-fleet \
    --scheduled-action-name my-recurring-action
```

此命令不生成任何输出。

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的计划扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScheduledAction。

deregister-scalable-target

以下代码示例演示了如何使用 deregister-scalable-target。

Amazon CLI

取消注册可扩展目标

此示例取消注册在默认集群中运行的 Amazon ECS 服务（名为 web-app）的可扩展目标。

命令:

```
aws application-autoscaling deregister-scalable-target --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app
```

此示例取消注册自定义资源的可扩展目标。custom-resource-id.txt 文件包含一个标识资源 ID 的字符串，对于自定义资源，该 ID 是通过 Amazon API Gateway 端点指向自定义资源的路径。

命令:

```
aws application-autoscaling deregister-scalable-target --service-namespace custom-
resource --scalable-dimension custom-resource:ResourceType:Property --resource-
id file://~/custom-resource-id.txt
```

custom-resource-id.txt 文件的内容：

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterScalableTarget。

describe-scalable-targets

以下代码示例演示了如何使用 describe-scalable-targets。

Amazon CLI

描述可扩展目标

以下 describe-scalable-targets 示例描述 ecs 服务命名空间的可扩展目标。

```
aws application-autoscaling describe-scalable-targets \
    --service-namespace ecs
```

输出：

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "ecs",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "MinCapacity": 1,
            "MaxCapacity": 10,
            "RoleARN": "arn:aws:iam::123456789012:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService",
            "CreationTime": 1462558906.199,
            "SuspendedState": {
                "DynamicScalingOutSuspended": false,
                "ScheduledScalingSuspended": false,
                "DynamicScalingInSuspended": false
            },
            "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
        }
    ]
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的可与 Application Auto Scaling 一起使用的 Amazon 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalableTargets。

describe-scaling-activities

以下代码示例演示了如何使用 describe-scaling-activities。

Amazon CLI

示例 1：描述指定 Amazon ECS 服务的扩展活动

以下 describe-scaling-activities 示例描述在 default 集群中运行的 Amazon ECS 服务（名为 web-app）的扩展活动。输出显示了由扩展策略启动的扩展活动。

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace ecs \
    --resource-id service/default/web-app
```

输出：

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/default/web-app",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered
 policy web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change
 successfully fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的 Application Auto Scaling 的扩展活动。

示例 2：描述指定 DynamoDB 表的扩展活动

以下 describe-scaling-activities 示例描述名为 TestTable 的 DynamoDB 表的扩展活动。输出显示了由两个不同的计划操作启动的扩展活动。

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace dynamodb \
    --resource-id table/TestTable
```

输出：

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
```

```
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity
 to 20",
            "StatusCode": "Successful"
        }
    ]
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的 Application Auto Scaling 的扩展活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingActivities。

`describe-scaling-policies`

以下代码示例演示了如何使用 `describe-scaling-policies`。

Amazon CLI

描述扩展策略

此示例命令描述 ecs 服务命名空间的扩展策略。

命令:

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

输出：

```
{
    "ScalingPolicies": [
```

```
        {
            "PolicyName": "web-app-cpu-gt-75",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462561899.23,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 60,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": 200,
                        "MetricIntervalLowerBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/
service/default/web-app:policyName/web-app-cpu-gt-75",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-gt-75",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-gt-75"
                }
            ],
            "ServiceNamespace": "ecs"
        },
        {
            "PolicyName": "web-app-cpu-lt-25",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462562575.099,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 1,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": -50,
                        "MetricIntervalUpperBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
```

```
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/
service/default/web-app:policyName/web-app-cpu-lt-25",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-lt-25",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-lt-25"
                }
            ],
            "ServiceNamespace": "ecs"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingPolicies。

describe-scheduled-actions

以下代码示例演示了如何使用 describe-scheduled-actions。

Amazon CLI

描述计划的操作

以下 describe-scheduled-actions 示例显示指定服务命名空间的计划操作的详细信息：

```
aws application-autoscaling describe-scheduled-actions \
    --service-namespace dynamodb
```

输出：

```
{
    "ScheduledActions": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-first-scheduled-action",
```

```
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
            "ScheduledActionName": "my-first-scheduled-action",
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-second-scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的计划扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledActions。

list-tags-for-resource

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出可扩展目标的标签

以下 `list-tags-for-resource` 示例列出附加到由其 ARN 指定的可扩展目标的标签键名称和值。

```
aws application-autoscaling list-tags-for-resource \
    --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-
target/1234abcd56ab78cd901ef1234567890ab123
```

输出：

```
{
    "Tags": {
        "environment": "production"
    }
}
```

有关更多信息，请参阅 Application Auto Scaling 用户指南中的 Application Auto Scaling 的标签支持。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-scaling-policy

以下代码示例演示了如何使用 put-scaling-policy。

Amazon CLI

示例 1：应用具有预定义指标规范的目标跟踪扩展策略

以下 put-scaling-policy 示例将具有预定义指标规范的目标跟踪扩展策略应用于默认集群中名为 web-app 的 Amazon ECS 服务。该策略将服务的平均 CPU 利用率保持在 75%，横向扩展和横向缩减冷却时间为 60 秒。输出包含代表您创建的两个 CloudWatch 警报的 ARN 和名称。

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cpu75-target-tracking-scaling-policy --policy-
type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

此示例假设您在当前目录中有一个 config.json 文件，其中包含以下内容：

```
{
     "TargetValue": 75.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

输出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/service/default/web-app:policyName/cpu75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-service/default/web-app-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

示例 2：应用具有自定义指标规范的目标跟踪扩展策略

以下 put-scaling-policy 示例将具有自定义指标规范的目标跟踪扩展策略应用于默认集群中名为 web-app 的 Amazon ECS 服务。该策略将服务的平均利用率保持在 75%，横向扩展和横向缩减冷却时间为 60 秒。输出包含代表您创建的两个 CloudWatch 警报的 ARN 和名称。

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cms75-target-tracking-scaling-policy
--policy-type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

此示例假设您在当前目录中有一个 config.json 文件，其中包含以下内容：

```
{
    "TargetValue":75.0,
    "CustomizedMetricSpecification":{
```

```
        "MetricName":"MyUtilizationMetric",
        "Namespace":"MyNamespace",
        "Dimensions": [
            {
                "Name":"MyOptionalMetricDimensionName",
                "Value":"MyOptionalMetricDimensionValue"
            }
        ],
        "Statistic":"Average",
        "Unit":"Percent"
    },
    "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

输出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/default/web-
app:policyName/cms75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

示例 3：为仅向外扩展应用目标跟踪扩展策略

以下 put-scaling-policy 示例将目标跟踪扩展策略应用于默认集群中名为 web-app 的 Amazon ECS 服务。当来自应用程序负载均衡器的 RequestCountPerTarget 指标超过阈值时，该策略用于横向扩展 ECS 服务。输出包含代表您创建的 CloudWatch 警报的 ARN 和名称。

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --policy-name alb-scale-out-target-tracking-scaling-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

的内容config.json：

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ALBRequestCountPerTarget",
         "ResourceLabel": "app/EC2Co-EcsEl-1TKLTMITMM0EO/f37c06a68c1748aa/
targetgroup/EC2Co-Defau-LDNM7Q3ZH1ZN/6d4ea56ca2d6a18d"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60,
    "DisableScaleIn": true
}
```

输出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:6d8972f3-
efc8-437c-92d1-6270f29a66e7:resource/ecs/service/default/web-app:policyName/alb-
scale-out-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Application Auto Scaling 用户指南》中的 Application Auto Scaling 的目标跟踪扩展策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutScalingPolicy。

put-scheduled-action

以下代码示例演示了如何使用 put-scheduled-action。

Amazon CLI

向 DynamoDB 表添加计划操作

此示例向名为 TestTable 的 DynamoDB 表添加一个计划操作，以便按周期性计划进行横向扩展。按照指定的计划（UTC 时间每天中午 12:15），如果当前容量低于为 MinCapacity 指定的值，则 Application Auto Scaling 将横向扩展到由 MinCapacity 指定的值。

命令:

```
aws application-autoscaling put-scheduled-action --service-namespace dynamodb
 --scheduled-action-name my-recurring-action --schedule "cron(15 12 * * ? *)" --
resource-id table/TestTable --scalable-dimension dynamodb:table:WriteCapacityUnits
 --scalable-target-action MinCapacity=6
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的“计划扩展”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutScheduledAction。

register-scalable-target

以下代码示例演示了如何使用 register-scalable-target。

Amazon CLI

示例 1：将 ECS 服务注册为可扩展目标

以下 register-scalable-target 示例向 Application Auto Scaling 注册 Amazon ECS 服务。它还向可扩展目标添加一个带有键名称 environment 和值 production 的标签。

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
```

```
    --resource-id service/default/web-app \
    --min-capacity 1 --max-capacity 10 \
    --tags environment=production
```

输出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

有关其它 Amazon 服务和自定义资源的示例，请参阅《Application Auto Scaling 用户指南》中的可与 Application Auto Scaling 结合使用的 Amazon 服务中的主题。

示例 2：暂停可扩展目标的扩展活动

以下 `register-scalable-target` 示例暂停现有可扩展目标的扩展活动。

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=true,DynamicScalingOutSuspended=true,ScheduledScalingSuspend
```

输出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的暂停和恢复 Application Auto Scaling 扩展。

示例 3：恢复可扩展目标的扩展活动

以下 `register-scalable-target` 示例恢复现有可扩展目标的扩展活动。

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
```

```
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=false,DynamicScalingOutSuspended=false,ScheduledScalingSuspe
```

输出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

有关更多信息，请参阅《Application Auto Scaling 用户指南》中的暂停和恢复 Application Auto Scaling 扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterScalableTarget。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向可扩展目标添加标签

以下 tag-resource 示例将带有键名称 environment 和值 production 的标签添加到由其 ARN 指定的可扩展目标。

```
aws application-autoscaling tag-resource \
    --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-
target/1234abcd56ab78cd901ef1234567890ab123 \
    --tags environment=production
```

此命令不生成任何输出。

有关更多信息，请参阅 Application Auto Scaling 用户指南中的 Application Auto Scaling 的标签支持。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从可扩展目标中移除标签

以下 `untag-resource` 示例从由其 ARN 指定的可扩展目标中移除带有键名称 `environment` 的标签对。

```
aws application-autoscaling untag-resource \
    --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123 \
    --tag-keys "environment"
```

此命令不生成任何输出。

有关更多信息，请参阅 Application Auto Scaling 用户指南中的 Application Auto Scaling 的标签支持。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Application Discovery Service 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Application Discovery Service 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-agents`

以下代码示例演示了如何使用 `describe-agents`。

Amazon CLI

描述具有指定 collectionStatus 状态的座席

此示例命令描述收集状态为“已启动”或“已停止”的收集座席。

命令:

```
aws discovery describe-agents --filters
 name="collectionStatus",values="STARTED","STOPPED",condition="EQUALS" --max-
results 3
```

输出：

```
{
       "Snapshots": [
      {
            "version": "1.0.40.0",
            "agentType": "EC2",
            "hostName": "ip-172-31-40-234",
            "collectionStatus": "STOPPED",
            "agentNetworkInfoList": [
                {
                    "macAddress": "06:b5:97:14:fc:0d",
                    "ipAddress": "172.31.40.234"
                }
            ],
            "health": "UNKNOWN",
            "agentId": "i-003305c02a776e883",
            "registeredTime": "2016-12-09T19:05:06Z",
            "lastHealthPingTime": "2016-12-09T19:05:10Z"
        },
        {
            "version": "1.0.40.0",
            "agentType": "EC2",
            "hostName": "ip-172-31-39-64",
            "collectionStatus": "STARTED",
            "agentNetworkInfoList": [
                {
                    "macAddress": "06:a1:0e:c7:b2:73",
                    "ipAddress": "172.31.39.64"
                }
            ],
            "health": "SHUTDOWN",
            "agentId": "i-003a5e5e2b36cf8bd",
            "registeredTime": "2016-11-16T16:36:25Z",
            "lastHealthPingTime": "2016-11-16T16:47:37Z"
```

```
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAgents。

describe-configurations

以下代码示例演示了如何使用 describe-configurations。

Amazon CLI

描述选定的资产配置

此示例命令描述两台指定服务器的配置。该操作会从配置 ID 中检测资产类型。每个命令只允许使用一种类型的资产。

命令:

```
aws discovery describe-configurations --configuration-ids "d-server-099385097ef9fbcfb" "d-server-0c4f2dd1fee22c6c1"
```

输出：

```
{
    "configurations": [
        {
                "server.performance.maxCpuUsagePct": "0.0",
                "server.performance.maxDiskReadIOPS": "0.0",
                "server.performance.avgCpuUsagePct": "0.0",
                "server.type": "EC2",
                "server.performance.maxNetworkReadsPerSecondInKB": "0.19140625",
                "server.hostName": "ip-172-31-35-152",
                "server.configurationId": "d-server-0c4f2dd1fee22c6c1",
                "server.tags.hasMoreValues": "false",
                "server.performance.minFreeRAMInKB": "1543496.0",
                "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
                "server.performance.maxDiskReadsPerSecondInKB": "0.0",
                "server.applications": "[]",
                "server.performance.numDisks": "1",
                "server.performance.numCpus": "1",
                "server.performance.numCores": "1",
                "server.performance.maxDiskWriteIOPS": "0.0",
```

```
                "server.performance.maxNetworkWritesPerSecondInKB": "0.82421875",
                "server.performance.avgDiskWritesPerSecondInKB": "0.0",
                "server.networkInterfaceInfo": "[{\"name\":\"eth0\",
\"macAddress\":\"06:A7:7D:3F:54:57\",\"ipAddress\":\"172.31.35.152\",\"netMask\":
\"255.255.240.0\"},{\"name\":\"lo\",\"macAddress\":\"00:00:00:00:00:00\",\"ipAddress
\":\"127.0.0.1\",\"netMask\":\"255.0.0.0\"},{\"name\":\"eth0\",\"macAddress\":
\"06:A7:7D:3F:54:57\",\"ipAddress\":\"fe80::4a7:7dff:fe3f:5457\"},{\"name\":\"lo\",
\"macAddress\":\"00:00:00:00:00:00\",\"ipAddress\":\"::1\"}]",
                "server.performance.avgNetworkReadsPerSecondInKB":
 "0.04915364583333333",
                "server.tags": "[]",
                "server.applications.hasMoreValues": "false",
                "server.timeOfCreation": "2016-10-28 23:44:00.0",
                "server.agentId": "i-4447bc1b",
                "server.performance.maxDiskWritesPerSecondInKB": "0.0",
                "server.performance.avgDiskReadIOPS": "0.0",
                "server.performance.avgFreeRAMInKB": "1547210.1333333333",
                "server.performance.avgDiskReadsPerSecondInKB": "0.0",
                "server.performance.avgDiskWriteIOPS": "0.0",
                "server.performance.numNetworkCards": "2",
                "server.hypervisor": "xen",
                "server.networkInterfaceInfo.hasMoreValues": "false",
                "server.performance.avgNetworkWritesPerSecondInKB": "0.1380859375",
                "server.osName": "Linux - Amazon Linux AMI release 2015.03",
                "server.performance.totalRAMInKB": "1694732.0",
                "server.cpuType": "x64"
        },
        {
                "server.performance.maxCpuUsagePct": "100.0",
                "server.performance.maxDiskReadIOPS": "0.0",
                "server.performance.avgCpuUsagePct": "14.733333333333338",
                "server.type": "EC2",
                "server.performance.maxNetworkReadsPerSecondInKB": "13.400390625",
                "server.hostName": "ip-172-31-42-208",
                "server.configurationId": "d-server-099385097ef9fbcfb",
                "server.tags.hasMoreValues": "false",
                "server.performance.minFreeRAMInKB": "1531104.0",
                "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
                "server.performance.maxDiskReadsPerSecondInKB": "0.0",
                "server.applications": "[]",
                "server.performance.numDisks": "1",
                "server.performance.numCpus": "1",
                "server.performance.numCores": "1",
                "server.performance.maxDiskWriteIOPS": "1.0",
```

```
                "server.performance.maxNetworkWritesPerSecondInKB": "12.271484375",
                "server.performance.avgDiskWritesPerSecondInKB":
 "0.5333333333333334",
                "server.networkInterfaceInfo": "[{\"name\":\"eth0\",
\"macAddress\":\"06:4A:79:60:75:61\",\"ipAddress\":\"172.31.42.208\",\"netMask
\":\"255.255.240.0\"},{\"name\":\"eth0\",\"macAddress\":\"06:4A:79:60:75:61\",
\"ipAddress\":\"fe80::44a:79ff:fe60:7561\"},{\"name\":\"lo\",\"macAddress\":
\"00:00:00:00:00:00\",\"ipAddress\":\"::1\"},{\"name\":\"lo\",\"macAddress\":
\"00:00:00:00:00:00\",\"ipAddress\":\"127.0.0.1\",\"netMask\":\"255.0.0.0\"}]",
                "server.performance.avgNetworkReadsPerSecondInKB":
 "2.8720052083333334",
                "server.tags": "[]",
                "server.applications.hasMoreValues": "false",
                "server.timeOfCreation": "2016-10-28 23:44:30.0",
                "server.agentId": "i-c142b99e",
                "server.performance.maxDiskWritesPerSecondInKB": "4.0",
                "server.performance.avgDiskReadIOPS": "0.0",
                "server.performance.avgFreeRAMInKB": "1534946.4",
                "server.performance.avgDiskReadsPerSecondInKB": "0.0",
                "server.performance.avgDiskWriteIOPS": "0.13333333333333336",
                "server.performance.numNetworkCards": "2",
                "server.hypervisor": "xen",
                "server.networkInterfaceInfo.hasMoreValues": "false",
                "server.performance.avgNetworkWritesPerSecondInKB":
 "1.7977864583333332",
                "server.osName": "Linux - Amazon Linux AMI release 2015.03",
                "server.performance.totalRAMInKB": "1694732.0",
                "server.cpuType": "x64"
        }
    ]
}
```

描述选定的资产配置

此示例命令描述两个指定应用程序的配置。该操作会从配置 ID 中检测资产类型。每个命令只允许使用一种类型的资产。

命令:

```
aws discovery describe-configurations --configuration-ids "d-
application-0ac39bc0e4fad0e42" "d-application-02444a45288013764q"
```

输出：

```
{
    "configurations": [
        {
                "application.serverCount": "0",
                "application.name": "Application-12345",
                "application.lastModifiedTime": "2016-12-13 23:53:27.0",
                "application.description": "",
                "application.timeOfCreation": "2016-12-13 23:53:27.0",
                "application.configurationId": "d-application-0ac39bc0e4fad0e42"
        },
        {
                "application.serverCount": "0",
                "application.name": "Application-67890",
                "application.lastModifiedTime": "2016-12-13 23:53:33.0",
                "application.description": "",
                "application.timeOfCreation": "2016-12-13 23:53:33.0",
                "application.configurationId": "d-application-02444a45288013764"
         }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigurations。

`list-configurations`

以下代码示例演示了如何使用 `list-configurations`。

Amazon CLI

列出符合一组筛选条件的所有已发现的服务器

此示例命令列出已发现的与两种主机名模式中的任何一个匹配且未运行 Ubuntu 的服务器。

命令:

```
aws discovery list-configurations --configuration-type SERVER --filters
 name="server.hostName",values="172-31-35","172-31-42",condition="CONTAINS"
 name="server.osName",values="Ubuntu",condition="NOT_CONTAINS"
```

输出：

```
{
```

```
    "configurations": [
      {
            "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
            "server.type": "EC2",
            "server.hostName": "ip-172-31-42-208",
            "server.timeOfCreation": "2016-10-28 23:44:30.0",
            "server.configurationId": "d-server-099385097ef9fbcfb",
            "server.osName": "Linux - Amazon Linux AMI release 2015.03",
            "server.agentId": "i-c142b99e"
        },
        {
            "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
            "server.type": "EC2",
            "server.hostName": "ip-172-31-35-152",
            "server.timeOfCreation": "2016-10-28 23:44:00.0",
            "server.configurationId": "d-server-0c4f2dd1fee22c6c1",
            "server.osName": "Linux - Amazon Linux AMI release 2015.03",
            "server.agentId": "i-4447bc1b"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConfigurations。

使用 Amazon CLI 的 Application Signals 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Application Signals 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-get-service-level-objective-budget-report

以下代码示例演示了如何使用 batch-get-service-level-objective-budget-report。

Amazon CLI

检索一个或多个服务级别目标（SLO）预算报告。

以下 batch-get-service-level-objective-budget-report 示例检索一个或多个服务级别目标（SLO）预算报告。

```
aws application-signals batch-get-service-level-objective-budget-report \
--timestamp 1735059869 \
--slo-ids "arn:aws:application-signals:us-east-1:123456789101:slo/
SLOName1" "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName2"
```

输出：

```
{
    "Timestamp": "2024-12-24T22:34:29+05:30",
    "Reports": [{
            "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/
SLOName1",
            "Name": "Your SLO Name",
            "EvaluationType": "PeriodBased",
            "BudgetStatus": "OK",
            "Attainment": 100.0,
            "TotalBudgetSeconds": 6048,
            "BudgetSecondsRemaining": 6048,
            "Sli": {
                "SliMetric": {
                    "MetricDataQueries": [{
                        "Id": "m1",
                        "MetricStat": {
                            "Metric": {
                                "Namespace": "AWS/EC2",
                                "MetricName": "CPUUtilization",
                                "Dimensions": [{
                                    "Name": "InstanceId",
                                    "Value": "i-0e098765432522"
                                }]
```

```
                            },
                            "Period": 60,
                            "Stat": "Average"
                        },
                        "ReturnData": true
                    }]
                },
                "MetricThreshold": 200.0,
                "ComparisonOperator": "LessThanOrEqualTo"
            },
            "Goal": {
                "Interval": {
                    "RollingInterval": {
                        "DurationUnit": "DAY",
                        "Duration": 7
                    }
                },
                "AttainmentGoal": 99.0,
                "WarningThreshold": 50.0
            }
        },
        {
            "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/
SLOName2",
            "Name": "test",
            "EvaluationType": "PeriodBased",
            "BudgetStatus": "BREACHED",
            "Attainment": 97.39583275,
            "TotalBudgetSeconds": 86,
            "BudgetSecondsRemaining": -2154,
            "Sli": {
                "SliMetric": {
                    "MetricDataQueries": [{
                        "Id": "cwMetric",
                        "MetricStat": {
                            "Metric": {
                                "Namespace": "AWS/EC2",
                                "MetricName": "CPUUtilization",
                                "Dimensions": [{
                                    "Name": "InstanceId",
                                    "Value": "i-0e12345678922"
                                }]
                            },
                            "Period": 300,
```

```
                        "Stat": "Average"
                    },
                    "ReturnData": true
                }]
            },
            "MetricThreshold": 5.0,
            "ComparisonOperator": "GreaterThan"
        },
        "Goal": {
            "Interval": {
                "RollingInterval": {
                    "DurationUnit": "DAY",
                    "Duration": 1
                }
            },
            "AttainmentGoal": 99.9,
            "WarningThreshold": 30.0
        }
    }
  ],
  "Errors": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 BatchGetServiceLevelObjectiveBudgetReport。

`create-service-level-objective`

以下代码示例演示了如何使用 `create-service-level-objective`。

Amazon CLI

创建服务级别目标（SLO）

以下 `create-service-level-objective` 示例创建服务级别目标（SLO），有助于您确保关键业务运营符合客户期望。

```
aws application-signals create-service-level-objective \
    --name "SLOName" \
    --description "Description of your SLO" \
    --sli-config file://sli-config.json
```

的内容sli-config.json：

```
{
    "SliMetricConfig": {
        "MetricDataQueries": [
            {
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "Namespace": "AWS/EC2",
                        "MetricName": "CPUUtilization",
                        "Dimensions": [
                            {
                                "Name": "InstanceId",
                                "Value": "i-0e5a1234561522"
                            }
                        ]
                    },
                    "Period": 60,
                    "Stat": "Average"
                },
                "ReturnData": true
            }
        ]
    },
    "MetricThreshold": 200,
    "ComparisonOperator": "LessThanOrEqualTo"
}
```

输出：

```
{
    "Slo": {
    "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName",
    "Name": "SLOName",
    "Description": "Description of your SLO",
    "CreatedTime": "2024-12-27T08:16:09.032000+05:30",
    "LastUpdatedTime": "2024-12-27T08:16:09.032000+05:30",
    "Sli": {
        "SliMetric": {
            "MetricDataQueries": [
                {
                    "Id": "m1",
```

```
                    "MetricStat": {
                        "Metric": {
                            "Namespace": "AWS/EC2",
                            "MetricName": "CPUUtilization",
                            "Dimensions": [
                                {
                                    "Name": "InstanceId",
                                    "Value": "i-0e59876543234522"
                                }
                            ]
                        },
                        "Period": 60,
                        "Stat": "Average"
                    },
                    "ReturnData": true
                }
            ]
        },
        "MetricThreshold": 200.0,
        "ComparisonOperator": "LessThanOrEqualTo"
    },
    "EvaluationType": "PeriodBased",
    "Goal": {
        "Interval": {
            "RollingInterval": {
                "DurationUnit": "DAY",
                "Duration": 7
            }
        },
        "AttainmentGoal": 99.0,
        "WarningThreshold": 50.0
      }
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateServiceLevelObjective。

`delete-service-level-objective`

以下代码示例演示了如何使用 `delete-service-level-objective`。

Amazon CLI

删除指定的服务级别目标。

以下 delete-service-level-objective 示例删除指定的服务级别目标。

```
aws application-signals delete-service-level-objective \
    --id "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteServiceLevelObjective。

get-service-level-objective

以下代码示例演示了如何使用 get-service-level-objective。

Amazon CLI

返回有关在账户中创建的 SLO 的信息

以下 get-service-level-objective 示例返回有关在账户中创建的一个 SLO 的信息。

```
aws application-signals get-service-level-objective \
    --id "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName"
```

输出：

```
{
    "Slo": {
        "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName",
        "Name": "SLOName",
        "Description": "Description of your SLO",
        "CreatedTime": "2024-12-24T22:19:18.624000+05:30",
        "LastUpdatedTime": "2024-12-24T22:19:55.280000+05:30",
        "Sli": {
            "SliMetric": {
                "MetricDataQueries": [{
                    "Id": "m1",
                    "MetricStat": {
```

```
                        "Metric": {
                            "Namespace": "AWS/EC2",
                            "MetricName": "CPUUtilization",
                            "Dimensions": [{
                                "Name": "InstanceId",
                                "Value": "i-0e0987654321522"
                            }]
                        },
                        "Period": 60,
                        "Stat": "Average"
                    },
                    "ReturnData": true
                }]
            },
            "MetricThreshold": 200.0,
            "ComparisonOperator": "LessThanOrEqualTo"
        },
        "EvaluationType": "PeriodBased",
        "Goal": {
            "Interval": {
                "RollingInterval": {
                    "DurationUnit": "DAY",
                    "Duration": 7
                }
            },
            "AttainmentGoal": 99.0,
            "WarningThreshold": 50.0
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetServiceLevelObjective。

`get-service`

以下代码示例演示了如何使用 `get-service`。

Amazon CLI

返回有关 Application Signals 发现的服务的信息

以下 get-service 示例返回有关 Application Signals 发现的服务的信息。

```
aws application-signals get-service \
    --start-time 1732704000 \
    --end-time 1732714500 \
    --key-attributes Environment=lambda:default,Name=hello-world-python,Type=Service
```

输出：

```
{
    "Service": {
        "KeyAttributes": {
            "Environment": "lambda:default",
            "Name": "hello-world-python",
            "Type": "Service"
        },
        "AttributeMaps": [{
            "Lambda.Function.Name": "hello-world-python",
            "PlatformType": "AWS::Lambda"
        }],
        "MetricReferences": [{
            "Namespace": "ApplicationSignals",
            "MetricType": "LATENCY",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Latency"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "FAULT",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Fault"
        }, {
```

```
            "Namespace": "ApplicationSignals",
            "MetricType": "ERROR",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Error"
        }],
        "LogGroupReferences": [{
            "Identifier": "/aws/lambda/hello-world-python",
            "ResourceType": "AWS::Logs::LogGroup",
            "Type": "AWS::Resource"
        }]
    },
    "StartTime": "2024-11-27T10:00:00+00:00",
    "EndTime": "2024-11-27T14:00:01+00:00",
    "LogGroupReferences": [{
        "Identifier": "/aws/lambda/hello-world-python",
        "ResourceType": "AWS::Logs::LogGroup",
        "Type": "AWS::Resource"
    }]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetService。

`list-service-dependencies`

以下代码示例演示了如何使用 `list-service-dependencies`。

Amazon CLI

返回您指定的服务的服务依赖项列表

以下 `list-service-dependencies` 示例返回您指定的服务的服务依赖项列表。

```
aws application-signals list-service-dependencies \
    --start-time 1732021200 \
    --end-time 1732107600 \
```

```
    --key-attributes Environment=api-
gateway:prod, Name=PetAdoptionStatusUpdater,Type=Service
```

输出：

```
{
    "ServiceDependencies": [{
        "OperationName": "PUT /prod",
        "DependencyKeyAttributes": {
            "Environment": "lambda:default",
            "Name": "Services-name",
            "Type": "Service"
        },
        "DependencyOperationName": "Invoke",
        "MetricReferences": [{
            "Namespace": "ApplicationSignals",
            "MetricType": "LATENCY",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "api-gateway:prod"
            }, {
                "Name": "Operation",
                "Value": "PUT /prod"
            }, {
                "Name": "RemoteEnvironment",
                "Value": "lambda:default"
            }, {
                "Name": "RemoteOperation",
                "Value": "Invoke"
            }, {
                "Name": "RemoteService",
                "Value": "Services-name"
            }, {
                "Name": "Service",
                "Value": "PetAdoptionStatusUpdater"
            }],
            "MetricName": "Latency"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "FAULT",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "api-gateway:prod"
```

```
            }, {
                "Name": "Operation",
                "Value": "PUT /prod"
            }, {
                "Name": "RemoteEnvironment",
                "Value": "lambda:default"
            }, {
                "Name": "RemoteOperation",
                "Value": "Invoke"
            }, {
                "Name": "RemoteService",
                "Value": "Services-name"
            }, {
                "Name": "Service",
                "Value": "PetAdoptionStatusUpdater"
            }],
            "MetricName": "Fault"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "ERROR",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "api-gateway:prod"
            }, {
                "Name": "Operation",
                "Value": "PUT /prod"
            }, {
                "Name": "RemoteEnvironment",
                "Value": "lambda:default"
            }, {
                "Name": "RemoteOperation",
                "Value": "Invoke"
            }, {
                "Name": "RemoteService",
                "Value": "Services-name"
            }, {
                "Name": "Service",
                "Value": "PetAdoptionStatusUpdater"
            }],
            "MetricName": "Error"
        }]
    }],
    "StartTime": "2024-11-19T13:00:00+00:00",
    "EndTime": "2024-11-20T13:00:01+00:00"
```

```
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceDependencies。

`list-service-dependents`

以下代码示例演示了如何使用 `list-service-dependents`。

Amazon CLI

返回在提供的时间范围内调用指定服务的依赖项列表

以下 `list-service-dependents` 示例返回在提供的时间范围内调用指定服务的依赖项列表。

```
aws application-signals list-service-dependents \
    --start-time 1732021200 \
    --end-time 1732107600 \
    --key-attributes Environment=generic:default,Name=PetSite,Type=Service
```

输出：

```
{
    "ServiceDependents": [{
        "OperationName": "",
        "DependentKeyAttributes": {
            "Identifier": "pet-api-canary-hao",
            "ResourceType": "AWS::Synthetics::Canary",
            "Type": "AWS::Resource"
        },
        "DependentOperationName": "",
        "MetricReferences": []
    }, {
        "OperationName": "",
        "DependentKeyAttributes": {
            "Identifier": "PetSite",
            "ResourceType": "AWS::Synthetics::Canary",
            "Type": "AWS::Resource"
        },
        "DependentOperationName": "",
```

```
        "MetricReferences": []
    }],
    "StartTime": "2024-12-24T05:00:00+00:00",
    "EndTime": "2024-12-25T06:00:01+00:00"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceDependents。

list-service-level-objectives

以下代码示例演示了如何使用 list-service-level-objectives。

Amazon CLI

返回在此账户中创建的 SLO 列表。

以下 list-service-level-objectives 示例返回在此账户中创建的 SLO 列表。

```
aws application-signals list-service-level-objectives
```

输出：

```
{
    "SloSummaries": [{
        "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/test",
        "Name": "test",
        "CreatedTime": "2024-12-24T22:01:21.116000+05:30"
    }]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceLevelObjectives。

list-service-operations

以下代码示例演示了如何使用 list-service-operations。

Amazon CLI

返回此服务的已由 Application Signals 发现的操作的列表

以下 `list-service-operations` 示例返回此服务的已由 Application Signals 发现的操作的列表。

```
aws application-signals list-service-operations \
    --start-time 1735017423 \
    --end-time 1735103823 \
    --key-attributes Environment=generic:default,Name=payforadoption,Type=Service
```

输出：

```
{
    "ServiceOperations": [{
        "Name": "POST /api",
        "MetricReferences": [{
            "Namespace": "ApplicationSignals",
            "MetricType": "LATENCY",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "generic:default"
            }, {
                "Name": "Operation",
                "Value": "POST /api"
            }, {
                "Name": "Service",
                "Value": "payforadoption"
            }],
            "MetricName": "Latency"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "FAULT",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "generic:default"
            }, {
                "Name": "Operation",
                "Value": "POST /api"
            }, {
                "Name": "Service",
                "Value": "payforadoption"
```

```
            }],
            "MetricName": "Fault"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "ERROR",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "generic:default"
            }, {
                "Name": "Operation",
                "Value": "POST /api"
            }, {
                "Name": "Service",
                "Value": "payforadoption"
            }],
            "MetricName": "Error"
        }]
    }],
    "StartTime": "2024-12-24T05:00:00+00:00",
    "EndTime": "2024-12-25T06:00:01+00:00"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceOperations。

`list-services`

以下代码示例演示了如何使用 `list-services`。

Amazon CLI

返回 Application Signals 已发现的服务列表

以下 `list-services` 示例返回 Application Signals 已发现的服务列表。

```
aws application-signals list-services \
    --start-time 1734918791 \
    --end-time 1734965591
```

输出：

```
{
```

```
    "ServiceSummaries": [{
        "KeyAttributes": {
            "Environment": "lambda:default",
            "Name": "hello-world-python",
            "Type": "Service"
        },
        "AttributeMaps": [{
            "Lambda.Function.Name": "hello-world-python",
            "PlatformType": "AWS::Lambda"
        }],
        "MetricReferences": [{
            "Namespace": "ApplicationSignals",
            "MetricType": "LATENCY",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Latency"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "FAULT",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Fault"
        }, {
            "Namespace": "ApplicationSignals",
            "MetricType": "ERROR",
            "Dimensions": [{
                "Name": "Environment",
                "Value": "lambda:default"
            }, {
                "Name": "Service",
                "Value": "hello-world-python"
            }],
            "MetricName": "Error"
        }]
```

```
    }],
    "StartTime": "2024-11-27T10:00:00+00:00",
    "EndTime": "2024-11-27T14:00:01+00:00"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

显示与 CloudWatch 资源关联的标签

以下 `list-tags-for-resource` 示例与 CloudWatch 资源关联的标签。

```
aws application-signals list-tags-for-resource \
    --resource-arn "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName"
```

输出：

```
{
    "Tags": [{
        "Key": "test",
        "Value": "value"
    }]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`start-discovery`

以下代码示例演示了如何使用 `start-discovery`。

Amazon CLI

使此 Amazon Web Services 账户能够使用 CloudWatch Application Signals

以下 `start-discovery` 示例通过创建 AmazonServiceRoleForCloudWatchApplicationSignals 服务相关角色，使此 Amazon Web Services 账户能够使用 CloudWatch Application Signals。

```
aws application-signals start-discovery
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartDiscovery。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

将一个或多个标签（键值对）分配给指定的 CloudWatch 资源，例如服务级别目标

以下 `tag-resource` 示例将一个或多个标签（键值对）分配给指定的 CloudWatch 资源，例如服务级别目标。

```
aws application-signals tag-resource \
    --resource-arn "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName"
 \
    --tags '{"Key":"test","Value":"value"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从指定资源中移除一个或多个标签

以下 `untag-resource` 示例从指定的资源中移除一个或多个标签。

```
aws application-signals untag-resource \
    --resource-arn "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName"
 \
    --tag-keys "test"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-service-level-objective

以下代码示例演示了如何使用 update-service-level-objective。

Amazon CLI

更新现有服务级别目标（SLO）

以下 update-service-level-objective 示例更新现有服务级别目标（SLO）。

```
aws application-signals update-service-level-objective \
--cli-input-json file://update-slo.json
```

的内容update-slo.json：

```
{
    "id": "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName",
    "goal": {
        "Interval": {
            "RollingInterval": {
                "DurationUnit": "DAY",
                "Duration": 7
            }
        },
        "AttainmentGoal": 90.0,
        "WarningThreshold": 50.0
    }
}
```

输出：

```
{
    "Slo": {
        "Arn": "arn:aws:application-signals:us-east-1:123456789101:slo/SLOName",
        "Name": "SLOName",
        "Description": "Description of your SLO",
        "CreatedTime": "2024-12-24T22:19:18.624000+05:30",
        "LastUpdatedTime": "2024-12-27T08:51:38.278000+05:30",
        "Sli": {
            "SliMetric": {
                "MetricDataQueries": [{
                    "Id": "m1",
                    "MetricStat": {
                        "Metric": {
                            "Namespace": "AWS/EC2",
                            "MetricName": "CPUUtilization",
                            "Dimensions": [{
                                "Name": "InstanceId",
                                "Value": "i-00987654345222"
                            }]
                        },
                        "Period": 60,
                        "Stat": "Average"
                    },
                    "ReturnData": true
                }]
            },
            "MetricThreshold": 200.0,
            "ComparisonOperator": "LessThanOrEqualTo"
        },
        "EvaluationType": "PeriodBased",
        "Goal": {
            "Interval": {
                "RollingInterval": {
                    "DurationUnit": "DAY",
                    "Duration": 7
                }
            },
            "AttainmentGoal": 90.0,
            "WarningThreshold": 50.0
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Application Signals。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateServiceLevelObjective。

使用 Amazon CLI 的 AppRegistry 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 AppRegistry 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-attribute-group`

以下代码示例演示了如何使用 `associate-attribute-group`。

Amazon CLI

关联属性组

以下 `associate-attribute-group` 示例会将您 Amazon 账户中的特定属性组与您 Amazon 账户中的特定应用程序相关联。

```
aws servicecatalog-appregistry associate-attribute-group \
    --application "ExampleApplication" \
    --attribute-group "ExampleAttributeGroup"
```

输出：

```
{
```

```
    "applicationArn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt9l",
    "attributeGroupArn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl"
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的关联和取消关联属性组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAttributeGroup。

`create-application`

以下代码示例演示了如何使用 `create-application`。

Amazon CLI

创建应用程序

以下 `create-application` 示例在您的 Amazon 账户中创建一个新的应用程序。

```
aws servicecatalog-appregistry create-application \
    --name "ExampleApplication"
```

输出：

```
{
    "application": {
        "id": "0ars38r6btoohvpvd9gqrptt9l",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt9l",
        "name": "ExampleApplication",
        "creationTime": "2023-02-28T21:10:10.820000+00:00",
        "lastUpdateTime": "2023-02-28T21:10:10.820000+00:00",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的创建应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApplication。

create-attribute-group

以下代码示例演示了如何使用 create-attribute-group。

Amazon CLI

创建属性组

以下 create-attribute-group 示例在您的 Amazon 账户中创建一个新的属性组。

```
aws servicecatalog-appregistry create-attribute-group \
    --name "ExampleAttributeGroup" \
    --attributes '{"SomeKey1":"SomeValue1","SomeKey2":"SomeValue2"}'
```

输出：

```
{
    "attributeGroup": {
        "id": "01sj5xdwhbw54kejwnt09fnpcl",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl",
        "name": "ExampleAttributeGroup",
        "creationTime": "2023-02-28T20:38:01.389000+00:00",
        "lastUpdateTime": "2023-02-28T20:38:01.389000+00:00",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的创建属性组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAttributeGroup。

delete-application

以下代码示例演示了如何使用 delete-application。

Amazon CLI

删除 应用程序

以下 delete-application 示例在您的 Amazon 账户中删除一个特定应用程序。

```
aws servicecatalog-appregistry delete-application \
```

```
    --application "ExampleApplication3"
```

输出：

```
{
    "application": {
        "id": "055gw7aynr1i5mbv7kjwzx5945",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/055gw7aynr1i5mbv7kjwzx5945",
        "name": "ExampleApplication3",
        "creationTime": "2023-02-28T22:06:28.228000+00:00",
        "lastUpdateTime": "2023-02-28T22:06:28.228000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的删除应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApplication。

delete-attribute-group

以下代码示例演示了如何使用 delete-attribute-group。

Amazon CLI

示例 8：删除属性组

以下 delete-attribute-group 示例在您的 Amazon 账户中删除一个特定属性组。

```
aws servicecatalog-appregistry delete-attribute-group \
    --attribute-group "ExampleAttributeGroup3"
```

输出：

```
{
    "attributeGroup": {
        "id": "011ge6y3emyjijt8dw8jn6r0hv",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/011ge6y3emyjijt8dw8jn6r0hv",
        "name": "ExampleAttributeGroup3",
        "creationTime": "2023-02-28T22:05:35.224000+00:00",
```

```
        "lastUpdateTime": "2023-02-28T22:05:35.224000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的删除属性组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAttributeGroup。

get-application

以下代码示例演示了如何使用 get-application。

Amazon CLI

获取应用程序

以下 get-application 示例检索有关您 Amazon 账户中特定应用程序的元数据信息。

```
aws servicecatalog-appregistry get-application \
    --application "ExampleApplication"
```

输出：

```
{
    "id": "0ars38r6btoohvpvd9gqrptt9l",
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt9l",
    "name": "ExampleApplication",
    "creationTime": "2023-02-28T21:10:10.820000+00:00",
    "lastUpdateTime": "2023-02-28T21:10:10.820000+00:00",
    "associatedResourceCount": 0,
    "tags": {
        "aws:servicecatalog:applicationName": "ExampleApplication"
    },
    "integrations": {
        "resourceGroup": {
            "state": "CREATE_COMPLETE",
            "arn": "arn:aws:resource-groups:us-west-2:813737243517:group/
AWS_AppRegistry_Application-ExampleApplication"
        }
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的使用应用程序详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApplication。

get-attribute-group

以下代码示例演示了如何使用 get-attribute-group。

Amazon CLI

获取属性组

以下 get-attribute-group 示例检索您 Amazon 账户中的特定属性组。

```
aws servicecatalog-appregistry get-attribute-group \
    --attribute-group "ExampleAttributeGroup"
```

输出：

```
{
    "id": "01sj5xdwhbw54kejwnt09fnpcl",
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl",
    "name": "ExampleAttributeGroup",
    "attributes": "{\"SomeKey1\":\"SomeValue1\",\"SomeKey2\":\"SomeValue2\"}",
    "creationTime": "2023-02-28T20:38:01.389000+00:00",
    "lastUpdateTime": "2023-02-28T20:38:01.389000+00:00",
    "tags": {
        "aws:servicecatalog:attributeGroupName": "ExampleAttributeGroup"
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的管理属性组的元数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAttributeGroup。

list-applications

以下代码示例演示了如何使用 list-applications。

Amazon CLI

列出应用程序

以下 `list-applications` 示例检索您 Amazon 账户中所有应用程序的列表。

```
aws servicecatalog-appregistry list-applications
```

输出：

```
{
    "applications": [
        {
            "id": "03axw94pjfj3uan00tcgbrxnkw",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/03axw94pjfj3uan00tcgbrxnkw",
            "name": "ExampleApplication2",
            "creationTime": "2023-02-28T21:59:34.094000+00:00",
            "lastUpdateTime": "2023-02-28T21:59:34.094000+00:00"
        },
        {
            "id": "055gw7aynr1i5mbv7kjwzx5945",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/055gw7aynr1i5mbv7kjwzx5945",
            "name": "ExampleApplication3",
            "creationTime": "2023-02-28T22:06:28.228000+00:00",
            "lastUpdateTime": "2023-02-28T22:06:28.228000+00:00"
        },
        {
            "id": "0ars38r6btoohvpvd9gqrptt9l",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt9l",
            "name": "ExampleApplication",
            "description": "This is an example application",
            "creationTime": "2023-02-28T21:10:10.820000+00:00",
            "lastUpdateTime": "2023-02-28T21:24:19.729000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的查看应用程序详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListApplications。

list-associated-attribute-groups

以下代码示例演示了如何使用 list-associated-attribute-groups。

Amazon CLI

列出关联的属性组

以下 list-associated-attribute-groups 示例检索您 Amazon 账户中与您 Amazon 账户中特定应用程序关联的所有属性组的列表。

```
aws servicecatalog-appregistry list-associated-attribute-groups \
    --application "ExampleApplication"
```

输出：

```
{
    "attributeGroups": [
        "01sj5xdwhbw54kejwnt09fnpcl"
    ]
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的关联和取消关联属性组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssociatedAttributeGroups。

list-attribute-groups-for-application

以下代码示例演示了如何使用 list-attribute-groups-for-application。

Amazon CLI

列出应用程序的属性组

以下 list-attribute-groups-for-application 示例列出您 Amazon 账户中与您 Amazon 账户中特定应用程序关联的所有属性组的详细信息。

```
aws servicecatalog-appregistry list-attribute-groups-for-application \
```

```
    --application "ExampleApplication"
```

输出：

```
{
    "attributeGroupsDetails": [
        {
            "id": "01sj5xdwhbw54kejwnt09fnpcl",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl",
            "name": "ExampleAttributeGroup"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的查看属性组详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttributeGroupsForApplication。

list-attribute-groups

以下代码示例演示了如何使用 list-attribute-groups。

Amazon CLI

列出属性组

以下 list-attribute-groups 示例检索您 Amazon 账户中所有属性组的列表。

```
aws servicecatalog-appregistry list-attribute-groups
```

输出：

```
{
    "attributeGroups": [
        {
            "id": "011ge6y3emyjijt8dw8jn6r0hv",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/011ge6y3emyjijt8dw8jn6r0hv",
            "name": "ExampleAttributeGroup3",
```

```
            "creationTime": "2023-02-28T22:05:35.224000+00:00",
            "lastUpdateTime": "2023-02-28T22:05:35.224000+00:00"
        },
        {
            "id": "01sj5xdwhbw54kejwnt09fnpcl",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl",
            "name": "ExampleAttributeGroup",
            "description": "This is an example attribute group",
            "creationTime": "2023-02-28T20:38:01.389000+00:00",
            "lastUpdateTime": "2023-02-28T21:02:04.559000+00:00"
        },
        {
            "id": "03n1yffgq6d18vwrzxf0c70nm3",
            "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/03n1yffgq6d18vwrzxf0c70nm3",
            "name": "ExampleAttributeGroup2",
            "creationTime": "2023-02-28T21:57:30.687000+00:00",
            "lastUpdateTime": "2023-02-28T21:57:30.687000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的查看属性组详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttributeGroups。

update-application

以下代码示例演示了如何使用 update-application。

Amazon CLI

更新应用程序

以下 update-application 示例更新您 Amazon 账户中的特定应用程序，使其包含说明。

```
aws servicecatalog-appregistry update-application \
    --application "ExampleApplication" \
    --description "This is an example application"
```

输出：

```
{
    "application": {
        "id": "0ars38r6btoohvpvd9gqrptt9l",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt9l",
        "name": "ExampleApplication",
        "description": "This is an example application",
        "creationTime": "2023-02-28T21:10:10.820000+00:00",
        "lastUpdateTime": "2023-02-28T21:24:19.729000+00:00",
        "tags": {
            "aws:servicecatalog:applicationName": "ExampleApplication"
        }
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的编辑应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApplication。

update-attribute-group

以下代码示例演示了如何使用 update-attribute-group。

Amazon CLI

更新属性组

以下 update-attribute-group 示例更新您 Amazon 账户中的特定属性组，使其包含说明。

```
aws servicecatalog-appregistry update-attribute-group \
    --attribute-group "ExampleAttributeGroup" \
    --description "This is an example attribute group"
```

输出：

```
{
    "attributeGroup": {
        "id": "01sj5xdwhbw54kejwnt09fnpcl",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpcl",
        "name": "ExampleAttributeGroup",
```

```
        "description": "This is an example attribute group",
        "creationTime": "2023-02-28T20:38:01.389000+00:00",
        "lastUpdateTime": "2023-02-28T21:02:04.559000+00:00",
        "tags": {
            "aws:servicecatalog:attributeGroupName": "ExampleAttributeGroup"
        }
    }
}
```

有关更多信息，请参阅《Amazon Service Catalog AppRegistry 管理员指南》中的编辑属性组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAttributeGroup。

使用 Amazon CLI 的 Athena 的示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Athena 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-get-named-query`

以下代码示例演示了如何使用 `batch-get-named-query`。

Amazon CLI

返回有关多个查询的信息

以下 `batch-get-named-query` 示例返回有关具有指定 ID 的命名查询的信息。

```
aws athena batch-get-named-query \
```

```
    --named-query-ids a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222 a1b2c3d4-5678-90ab-cdef-EXAMPLE33333
```

输出：

```
{
    "NamedQueries": [
        {
            "Name": "Flights Select Query",
            "Description": "Sample query to get the top 10 airports with the most
 number of departures since 2000",
            "Database": "sampledb",
            "QueryString": "SELECT origin, count(*) AS total_departures\nFROM
\nflights_parquet\nWHERE year >= '2000'\nGROUP BY origin\nORDER BY total_departures
 DESC\nLIMIT 10;",
            "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "WorkGroup": "primary"
        },
        {
            "Name": "Load flights table partitions",
            "Description": "Sample query to load flights table partitions using MSCK
 REPAIR TABLE statement",
            "Database": "sampledb",
            "QueryString": "MSCK REPAIR TABLE flights_parquet;",
            "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "WorkGroup": "primary"
        },
        {
            "Name": "CloudFront Select Query",
            "Description": "Sample query to view requests per operating system
 during a particular time frame",
            "Database": "sampledb",
            "QueryString": "SELECT os, COUNT(*) count FROM cloudfront_logs WHERE
 date BETWEEN date '2014-07-05' AND date '2014-08-05' GROUP BY os;",
            "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "WorkGroup": "primary"
        }
    ],
    "UnprocessedNamedQueryIds": []
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetNamedQuery。

batch-get-query-execution

以下代码示例演示了如何使用 batch-get-query-execution。

Amazon CLI

返回有关一个或多个查询执行的信息

以下 batch-get-query-execution 示例返回具有指定查询 ID 的查询的查询执行信息。

```
aws athena batch-get-query-execution \
    --query-execution-ids a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222
```

输出：

```
{
    "QueryExecutions": [
        {
            "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Query": "create database if not exists webdata",
            "StatementType": "DDL",
            "ResultConfiguration": {
                "OutputLocation": "s3://amzn-s3-demo-bucket/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111.txt"
            },
            "QueryExecutionContext": {},
            "Status": {
                "State": "SUCCEEDED",
                "SubmissionDateTime": 1593470720.592,
                "CompletionDateTime": 1593470720.902
            },
            "Statistics": {
                "EngineExecutionTimeInMillis": 232,
                "DataScannedInBytes": 0,
                "TotalExecutionTimeInMillis": 310,
            "ResultConfiguration": {

                "QueryQueueTimeInMillis": 50,
                "ServiceProcessingTimeInMillis": 28
            },
            "WorkGroup": "AthenaAdmin"
        },
```

```
        {
            "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Query": "select date, location, browser, uri, status from
 cloudfront_logs where method = 'GET' and status = 200 and location like 'SFO%'
 limit 10",
            "StatementType": "DML",
            "ResultConfiguration": {
                "OutputLocation": "s3://amzn-s3-demo-bucket/a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222.csv"
            },
            "QueryExecutionContext": {
                "Database": "mydatabase",
                "Catalog": "awsdatacatalog"
            },
            "Status": {
                "State": "SUCCEEDED",
                "SubmissionDateTime": 1593469842.665,
                "CompletionDateTime": 1593469846.486
            },
            "Statistics": {
                "EngineExecutionTimeInMillis": 3600,
                "DataScannedInBytes": 203089,
                "TotalExecutionTimeInMillis": 3821,
                "QueryQueueTimeInMillis": 267,
                "QueryPlanningTimeInMillis": 1175
            },
            "WorkGroup": "AthenaAdmin"
        }
    ],
    "UnprocessedQueryExecutionIds": []
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetQueryExecution。

`create-data-catalog`

以下代码示例演示了如何使用 `create-data-catalog`。

Amazon CLI

创建数据目录

以下 create-data-catalog 示例创建 dynamo_db_catalog 数据目录。

```
aws athena create-data-catalog \
    --name dynamo_db_catalog \
    --type LAMBDA \
    --description "DynamoDB Catalog" \
    --parameters function=arn:aws:lambda:us-
west-2:111122223333:function:dynamo_db_lambda
```

此命令不生成任何输出。要查看结果，请使用 aws athena get-data-catalog --name dynamo_db_catalog。

有关更多信息，请参阅《Amazon Athena 用户指南》中的注册目录：create-data-catalog。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataCatalog。

create-named-query

以下代码示例演示了如何使用 create-named-query。

Amazon CLI

创建命名查询

以下 create-named-query 示例在 AthenaAdmin 工作组中创建一个已保存的查询，查询 flights_parquet 表中 2016 年 1 月从西雅图飞往纽约的航班，这些航班的起飞和到达时间都延误了十分钟以上。由于表中的机场代码值是包含双引号的字符串（例如，“SEA”），因此使用反斜杠转义，并用单引号扩起。

```
aws athena create-named-query \
    --name "SEA to JFK delayed flights Jan 2016" \
    --description "Both arrival and departure delayed more than 10 minutes." \
    --database sampledb \
    --query-string "SELECT flightdate, carrier, flightnum, origin, dest,
 depdelayminutes, arrdelayminutes FROM sampledb.flights_parquet WHERE yr = 2016 AND
 month = 1 AND origin = '\"SEA\"' AND dest = '\"JFK\"' AND depdelayminutes > 10 AND
 arrdelayminutes > 10" \
    --work-group AthenaAdmin
```

输出：

```
{
```

```
    "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNamedQuery。

create-work-group

以下代码示例演示了如何使用 create-work-group。

Amazon CLI

创建工作组

以下 create-work-group 示例创建一个名为 Data_Analyst_Group 的工作组，具有查询结果的输出位置 s3://amzn-s3-demo-bucket。该命令创建一个覆盖客户端配置设置的工作组，其中包括查询结果的输出位置。该命令还启用 CloudWatch 指标，并向工作组添加三个键值标签对，以将其与其他工作组区分开来。请注意，--configuration 参数在分隔其选项的逗号前没有空格。

```
aws athena create-work-group \
    --name Data_Analyst_Group \
    --configuration ResultConfiguration={OutputLocation="s3://amzn-s3-demo-
bucket"},EnforceWorkGroupConfiguration="true",PublishCloudWatchMetricsEnabled="true"
 \
    --description "Workgroup for data analysts" \
    --tags Key=Division,Value=West Key=Location,Value=Seattle Key=Team,Value="Big
 Data"
```

此命令不生成任何输出。要查看结果，请使用 aws athena get-work-group --work-group Data_Analyst_Group。

有关更多信息，请参阅《Amazon Athena 用户指南》中的管理工作组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWorkGroup。

delete-data-catalog

以下代码示例演示了如何使用 delete-data-catalog。

Amazon CLI

删除数据目录

以下 delete-data-catalog 示例删除 UnusedDataCatalog 数据目录。

```
aws athena delete-data-catalog \
    --name UnusedDataCatalog
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Athena 用户指南》中的删除目录：delete-data-catalog。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDataCatalog。

delete-named-query

以下代码示例演示了如何使用 delete-named-query。

Amazon CLI

删除命名查询

以下 delete-named-query 示例删除具有指定 ID 的命名查询。

```
aws athena delete-named-query \
    --named-query-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNamedQuery。

delete-work-group

以下代码示例演示了如何使用 delete-work-group。

Amazon CLI

删除工作组

以下 delete-work-group 示例删除 TeamB 工作组。

```
aws athena delete-work-group \
    --work-group TeamB
```

此命令不生成任何输出。要确认删除，请使用 `aws athena list-work-groups`。

有关更多信息，请参阅《Amazon Athena 用户指南》中的管理工作组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWorkGroup。

get-data-catalog

以下代码示例演示了如何使用 `get-data-catalog`。

Amazon CLI

返回有关数据目录的信息

以下 `get-data-catalog` 示例返回有关 `dynamo_db_catalog` 数据目录的信息。

```
aws athena get-data-catalog \
    --name dynamo_db_catalog
```

输出：

```
{
    "DataCatalog": {
        "Name": "dynamo_db_catalog",
        "Description": "DynamoDB Catalog",
        "Type": "LAMBDA",
        "Parameters": {
            "catalog": "dynamo_db_catalog",
            "metadata-function": "arn:aws:lambda:us-
west-2:111122223333:function:dynamo_db_lambda",
            "record-function": "arn:aws:lambda:us-
west-2:111122223333:function:dynamo_db_lambda"
        }
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的显示目录详细信息：get-data-catalog。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataCatalog。

get-database

以下代码示例演示了如何使用 get-database。

Amazon CLI

返回有关数据目录中数据库的信息

以下 get-database 示例返回有关 AwsDataCatalog 数据目录中 sampledb 数据库的信息。

```
aws athena get-database \
    --catalog-name AwsDataCatalog \
    --database-name sampledb
```

输出：

```
{
    "Database": {
        "Name": "sampledb",
        "Description": "Sample database",
        "Parameters": {
            "CreatedBy": "Athena",
            "EXTERNAL": "TRUE"
        }
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的显示数据库详细信息：get-database。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDatabase。

get-named-query

以下代码示例演示了如何使用 get-named-query。

Amazon CLI

返回命名查询

以下 get-named-query 示例返回有关具有指定 ID 的查询的信息。

```
aws athena get-named-query \
```

```
    --named-query-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "NamedQuery": {
        "Name": "CloudFront Logs - SFO",
        "Description": "Shows successful GET request data for SFO",
        "Database": "default",
        "QueryString": "select date, location, browser, uri, status from
 cloudfront_logs where method = 'GET' and status = 200 and location like 'SFO%'
 limit 10",
        "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "WorkGroup": "AthenaAdmin"
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetNamedQuery。

get-query-execution

以下代码示例演示了如何使用 get-query-execution。

Amazon CLI

返回有关查询执行的信息

以下 get-query-execution 示例返回有关具有指定查询 ID 的查询的信息。

```
aws athena get-query-execution \
    --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "QueryExecution": {
        "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Query": "select date, location, browser, uri, status from cloudfront_logs
 where method = 'GET
```

```
' and status = 200 and location like 'SFO%' limit 10",
        "StatementType": "DML",
        "ResultConfiguration": {
            "OutputLocation": "s3://amzn-s3-demo-bucket/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111.csv"
        },
        "QueryExecutionContext": {
            "Database": "mydatabase",
            "Catalog": "awsdatacatalog"
        },
        "Status": {
            "State": "SUCCEEDED",
            "SubmissionDateTime": 1593469842.665,
            "CompletionDateTime": 1593469846.486
        },
        "Statistics": {
            "EngineExecutionTimeInMillis": 3600,
            "DataScannedInBytes": 203089,
            "TotalExecutionTimeInMillis": 3821,
            "QueryQueueTimeInMillis": 267,
            "QueryPlanningTimeInMillis": 1175
        },
        "WorkGroup": "AthenaAdmin"
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueryExecution。

get-query-results

以下代码示例演示了如何使用 get-query-results。

Amazon CLI

返回查询结果

以下 get-query-results 示例返回具有指定查询 ID 的查询的结果。

```
aws athena get-query-results \
    --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：


```
{
    "ResultSet": {
        "Rows": [
            {
                "Data": [
                    {
                        "VarCharValue": "date"
                    },
                    {
                        "VarCharValue": "location"
                    },
                    {
                        "VarCharValue": "browser"
                    },
                    {
                        "VarCharValue": "uri"
                    },
                    {
                        "VarCharValue": "status"
                    }
                ]
            },
            {
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
                    },
                    {
                        "VarCharValue": "Safari"
                    },
                    {
                        "VarCharValue": "/test-image-2.jpeg"
                    },
                    {
                        "VarCharValue": "200"
                    }
                ]
            },
            {
```

```
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
                    },
                    {
                        "VarCharValue": "Opera"
                    },
                    {
                        "VarCharValue": "/test-image-2.jpeg"
                    },
                    {
                        "VarCharValue": "200"
                    }
                ]
            },
            {
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
                    },
                    {
                        "VarCharValue": "Firefox"
                    },
                    {
                        "VarCharValue": "/test-image-3.jpeg"
                    },
                    {
                        "VarCharValue": "200"
                    }
                ]
            },
            {
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
```

```
                },
                {
                    "VarCharValue": "Lynx"
                },
                {
                    "VarCharValue": "/test-image-3.jpeg"
                },
                {
                    "VarCharValue": "200"
                }
            ]
        },
        {
            "Data": [
                {
                    "VarCharValue": "2014-07-05"
                },
                {
                    "VarCharValue": "SFO4"
                },
                {
                    "VarCharValue": "IE"
                },
                {
                    "VarCharValue": "/test-image-2.jpeg"
                },
                {
                    "VarCharValue": "200"
                }
            ]
        },
        {
            "Data": [
                {
                    "VarCharValue": "2014-07-05"
                },
                {
                    "VarCharValue": "SFO4"
                },
                {
                    "VarCharValue": "Opera"
                },
                {
                    "VarCharValue": "/test-image-1.jpeg"
```

```
                },
                {
                    "VarCharValue": "200"
                }
            ]
        },
        {
            "Data": [
                {
                    "VarCharValue": "2014-07-05"
                },
                {
                    "VarCharValue": "SFO4"
                },
                {
                    "VarCharValue": "Chrome"
                },
                {
                    "VarCharValue": "/test-image-3.jpeg"
                },
                {
                    "VarCharValue": "200"
                }
            ]
        },
        {
            "Data": [
                {
                    "VarCharValue": "2014-07-05"
                },
                {
                    "VarCharValue": "SFO4"
                },
                {
                    "VarCharValue": "Firefox"
                },
                {
                    "VarCharValue": "/test-image-2.jpeg"
                },
                {
                    "VarCharValue": "200"
                }
            ]
        },
```

```
            {
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
                    },
                    {
                        "VarCharValue": "Chrome"
                    },
                    {
                        "VarCharValue": "/test-image-3.jpeg"
                    },
                    {
                        "VarCharValue": "200"
                    }
                ]
            },
            {
                "Data": [
                    {
                        "VarCharValue": "2014-07-05"
                    },
                    {
                        "VarCharValue": "SFO4"
                    },
                    {
                        "VarCharValue": "IE"
                    },
                    {
                        "VarCharValue": "/test-image-2.jpeg"
                    },
                    {
                        "VarCharValue": "200"
                    }
                ]
            }
        ],
        "ResultSetMetadata": {
            "ColumnInfo": [
                {
                    "CatalogName": "hive",
                    "SchemaName": "",
```

```
                    "TableName": "",
                    "Name": "date",
                    "Label": "date",
                    "Type": "date",
                    "Precision": 0,
                    "Scale": 0,
                    "Nullable": "UNKNOWN",
                    "CaseSensitive": false
                },
                {
                    "CatalogName": "hive",
                    "SchemaName": "",
                    "TableName": "",
                    "Name": "location",
                    "Label": "location",
                    "Type": "varchar",
                    "Precision": 2147483647,
                "Data": [

                    "Scale": 0,
                    "Nullable": "UNKNOWN",
                    "CaseSensitive": true
                },
                {
                    "CatalogName": "hive",
                    "SchemaName": "",
                    "TableName": "",
                    "Name": "browser",
                    "Label": "browser",
                    "Type": "varchar",
                    "Precision": 2147483647,
                    "Scale": 0,
                    "Nullable": "UNKNOWN",
                    "CaseSensitive": true
                },
                {
                    "CatalogName": "hive",
                    "SchemaName": "",
                    "TableName": "",
                    "Name": "uri",
                    "Label": "uri",
                    "Type": "varchar",
                    "Precision": 2147483647,
                    "Scale": 0,
```

```
                    "Nullable": "UNKNOWN",
                    "CaseSensitive": true
                },
                {
                    "CatalogName": "hive",
                    "SchemaName": "",
                    "TableName": "",
                    "Name": "status",
                    "Label": "status",
                    "Type": "integer",
                    "Precision": 10,
                    "Scale": 0,
                    "Nullable": "UNKNOWN",
                    "CaseSensitive": false
                }
            ]
        }
    },
    "UpdateCount": 0
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用查询结果、输出文件和查询历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueryResults。

get-table-metadata

以下代码示例演示了如何使用 get-table-metadata。

Amazon CLI

返回有关表的元数据信息

以下 get-table-metadata 示例从 AwsDataCatalog 数据目录的 sampledb 数据库返回有关 counties 表的元数据信息，包括列名及其数据类型。

```
aws athena get-table-metadata \
    --catalog-name AwsDataCatalog \
    --database-name sampledb \
    --table-name counties
```

输出：

```
{
    "TableMetadata": {
        "Name": "counties",
        "CreateTime": 1593559968.0,
        "LastAccessTime": 0.0,
        "TableType": "EXTERNAL_TABLE",
        "Columns": [
            {
                "Name": "name",
                "Type": "string",
                "Comment": "from deserializer"
            },
            {
                "Name": "boundaryshape",
                "Type": "binary",
                "Comment": "from deserializer"
            },
            {
                "Name": "motto",
                "Type": "string",
                "Comment": "from deserializer"
            },
            {
                "Name": "population",
                "Type": "int",
                "Comment": "from deserializer"
            }
        ],
        "PartitionKeys": [],
        "Parameters": {
            "EXTERNAL": "TRUE",
            "inputformat": "com.esri.json.hadoop.EnclosedJsonInputFormat",
            "location": "s3://amzn-s3-demo-bucket/json",
            "outputformat":
 "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
            "serde.param.serialization.format": "1",
            "serde.serialization.lib": "com.esri.hadoop.hive.serde.JsonSerde",
            "transient_lastDdlTime": "1593559968"
        }
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的显示表详细信息：get-table-metadata。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTableMetadata。

get-work-group

以下代码示例演示了如何使用 get-work-group。

Amazon CLI

返回有关工作组的信息

以下 get-work-group 示例返回有关 AthenaAdmin 工作组的信息。

```
aws athena get-work-group \
    --work-group AthenaAdmin
```

输出：

```
{
    "WorkGroup": {
        "Name": "AthenaAdmin",
        "State": "ENABLED",
        "Configuration": {
            "ResultConfiguration": {
                "OutputLocation": "s3://amzn-s3-demo-bucket/"
            },
            "EnforceWorkGroupConfiguration": false,
            "PublishCloudWatchMetricsEnabled": true,
            "RequesterPaysEnabled": false
        },
        "Description": "Workgroup for Athena administrators",
        "CreationTime": 1573677174.105
    }
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的管理工作组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWorkGroup。

list-data-catalogs

以下代码示例演示了如何使用 list-data-catalogs。

Amazon CLI

列出在 Athena 注册的数据目录

以下 `list-data-catalogs` 示例列出在 Athena 中注册的数据目录。

```
aws athena list-data-catalogs
```

输出：

```
{
    "DataCatalogsSummary": [
        {
            "CatalogName": "AwsDataCatalog",
            "Type": "GLUE"
        },
        {
            "CatalogName": "cw_logs_catalog",
            "Type": "LAMBDA"
        },
        {
            "CatalogName": "cw_metrics_catalog",
            "Type": "LAMBDA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的列出已注册目录：list-data-catalogs。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDataCatalogs。

list-databases

以下代码示例演示了如何使用 `list-databases`。

Amazon CLI

列出数据目录中的数据库

以下 `list-databases` 示例列出 `AwsDataCatalog` 数据目录中的数据库。

```
aws athena list-databases \
    --catalog-name AwsDataCatalog
```

输出：

```
{
    "DatabaseList": [
        {
            "Name": "default"
        },
        {
            "Name": "mydatabase"
        },
        {
            "Name": "newdb"
        },
        {
            "Name": "sampledb",
            "Description": "Sample database",
            "Parameters": {
                "CreatedBy": "Athena",
                "EXTERNAL": "TRUE"
            }
        },
        {
            "Name": "webdata"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的列出目录中的数据库：list-databases。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatabases。

list-named-queries

以下代码示例演示了如何使用 list-named-queries。

Amazon CLI

列出工作组的命名查询

以下 list-named-queries 示例列出 AthenaAdmin 工作组的命名查询。

```
aws athena list-named-queries \
    --work-group AthenaAdmin
```

输出：

```
{
    "NamedQueryIds": [
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"
    ]
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListNamedQueries。

list-query-executions

以下代码示例演示了如何使用 list-query-executions。

Amazon CLI

列出指定工作组中查询的查询 ID

以下 list-query-executions 示例列出 AthenaAdmin 工作组中最多十个查询 ID。

```
aws athena list-query-executions \
    --work-group AthenaAdmin \
    --max-items 10
```

输出：

```
{
    "QueryExecutionIds": [
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11110",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11114",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11115",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11116",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11117",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11118",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11119"
    ],
```

```
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0="
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用查询结果、输出文件和查询历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueryExecutions。

`list-table-metadata`

以下代码示例演示了如何使用 `list-table-metadata`。

Amazon CLI

列出数据目录的指定数据库中表的元数据

以下 `list-table-metadata` 示例返回 `AwsDataCatalog` 数据目录的 `geography` 数据库中最多两个表的元数据信息。

```
aws athena list-table-metadata \
    --catalog-name AwsDataCatalog \
    --database-name geography \
    --max-items 2
```

输出：

```
{
    "TableMetadataList": [
        {
            "Name": "country_codes",
            "CreateTime": 1586553454.0,
            "TableType": "EXTERNAL_TABLE",
            "Columns": [
                {
                    "Name": "country",
                    "Type": "string",
                    "Comment": "geo id"
                },
                {
                    "Name": "alpha-2 code",
                    "Type": "string",
                    "Comment": "geo id2"
                },
```

```
                {
                    "Name": "alpha-3 code",
                    "Type": "string",
                    "Comment": "state name"
                },
                {
                    "Name": "numeric code",
                    "Type": "bigint",
                    "Comment": ""
                },
                {
                    "Name": "latitude",
                    "Type": "bigint",
                    "Comment": "location (latitude)"
                },
                {
                    "Name": "longitude",
                    "Type": "bigint",
                    "Comment": "location (longitude)"
                }
            ],
            "Parameters": {
                "areColumnsQuoted": "false",
                "classification": "csv",
                "columnsOrdered": "true",
                "delimiter": ",",
                "has_encrypted_data": "false",
                "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
                "location": "s3://amzn-s3-demo-bucket/csv/countrycode",
                "outputformat":
"org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
                "serde.param.field.delim": ",",
                "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "skip.header.line.count": "1",
                "typeOfData": "file"
            }
        },
        {
            "Name": "county_populations",
            "CreateTime": 1586553446.0,
            "TableType": "EXTERNAL_TABLE",
            "Columns": [
                {
```

```
                    "Name": "id",
                    "Type": "string",
                    "Comment": "geo id"
                },
                {
                    "Name": "country",

                    "Name": "id2",
                    "Type": "string",
                    "Comment": "geo id2"
                },
                {
                    "Name": "county",
                    "Type": "string",
                    "Comment": "county name"
                },
                {
                    "Name": "state",
                    "Type": "string",
                    "Comment": "state name"
                },
                {
                    "Name": "population estimate 2018",
                    "Type": "string",
                    "Comment": ""
                }
            ],
            "Parameters": {
                "areColumnsQuoted": "false",
                "classification": "csv",
                "columnsOrdered": "true",
                "delimiter": ",",
                "has_encrypted_data": "false",
                "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
                "location": "s3://amzn-s3-demo-bucket/csv/CountyPopulation",
                "outputformat":
"org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
                "serde.param.field.delim": ",",
                "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "skip.header.line.count": "1",
                "typeOfData": "file"
            }
        }
```

```
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的显示数据库中所有表的元数据：list-table-metadata。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTableMetadata。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

示例 1：列出工作组的标签

以下 list-tags-for-resource 示例列出 Data_Analyst_Group 工作组的标签。

```
aws athena list-tags-for-resource \
    --resource-arn arn:aws:athena:us-west-2:111122223333:workgroup/
Data_Analyst_Group
```

输出：

```
{
    "Tags": [
        {
            "Key": "Division",
            "Value": "West"
        },
        {
            "Key": "Team",
            "Value": "Big Data"
        },
        {
            "Key": "Location",
            "Value": "Seattle"
        }
    ]
}
```

示例 2：列出数据目录的标签

以下 `list-tags-for-resource` 示例列出 dynamo_db_catalog 数据目录的标签。

```
aws athena list-tags-for-resource \
    --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/
dynamo_db_catalog
```

输出：

```
{
    "Tags": [
        {
            "Key": "Division",
            "Value": "Mountain"
        },
        {
            "Key": "Organization",
            "Value": "Retail"
        },
        {
            "Key": "Product_Line",
            "Value": "Shoes"
        },
        {
            "Key": "Location",
            "Value": "Denver"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的列出资源标签：list-tags-for-resource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-work-groups`

以下代码示例演示了如何使用 `list-work-groups`。

Amazon CLI

列出工作组

以下 list-work-groups 命令列出当前账户中的工作组。

```
aws athena list-work-groups
```

输出：

```
{
    "WorkGroups": [
        {
            "Name": "Data_Analyst_Group",
            "State": "ENABLED",
            "Description": "",
            "CreationTime": 1578006683.016
        },
        {
            "Name": "AthenaAdmin",
            "State": "ENABLED",
            "Description": "",
            "CreationTime": 1573677174.105
        },
        {
            "Name": "primary",
            "State": "ENABLED",
            "Description": "",
            "CreationTime": 1567465222.723
        }
    ]
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的管理工作组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWorkGroups。

start-query-execution

以下代码示例演示了如何使用 start-query-execution。

Amazon CLI

示例 1：在工作组中对指定数据库和数据目录中的指定表运行查询

以下 start-query-execution 示例使用 AthenaAdmin 工作组对 AwsDataCatalog 数据目录中 cflogsdatabase 的 cloudfront_logs 表运行查询。

```
aws athena start-query-execution \
    --query-string "select date, location, browser, uri, status from cloudfront_logs
 where method = 'GET' and status = 200 and location like 'SFO%' limit 10" \
    --work-group "AthenaAdmin" \
    --query-execution-context Database=cflogsdatabase,Catalog=AwsDataCatalog
```

输出：

```
{
"QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

示例 2：运行查询，使用指定工作组在指定数据目录中创建数据库

以下 start-query-execution 示例使用 AthenaAdmin 工作组在默认数据目录 AwsDataCatalog 中创建数据库 newdb。

```
aws athena start-query-execution \
    --query-string "create database if not exists newdb" \
    --work-group "AthenaAdmin"
```

输出：

```
{
"QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112"
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

示例 3：运行查询，在指定数据库和数据目录中的表上创建视图

以下 start-query-execution 示例在 cflogsdatabase 中的 cloudfront_logs 表上使用 SELECT 语句创建视图 cf10。

```
aws athena start-query-execution \
    --query-string  "CREATE OR REPLACE VIEW cf10 AS SELECT * FROM cloudfront_logs
 limit 10" \
    --query-execution-context Database=cflogsdatabase
```

输出：

```
{
"QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11113"
}
```

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartQueryExecution。

stop-query-execution

以下代码示例演示了如何使用 stop-query-execution。

Amazon CLI

停止正在运行的查询

以下 stop-query-execution 示例停止具有指定查询 ID 的查询。

```
aws athena stop-query-execution \
    --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Athena 用户指南》中的使用 Amazon Athena 运行 SQL 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopQueryExecution。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向资源添加标签

以下 tag-resource 示例向 dynamo_db_catalog 数据目录添加三个标签。

```
aws athena tag-resource \
    --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/
dynamo_db_catalog \
```

```
    --
tags Key=Organization,Value=Retail Key=Division,Value=Mountain Key=Product_Line,Value=Shoes
```

此命令不生成任何输出。要查看结果，请使用 `aws athena list-tags-for-resource --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/dynamo_db_catalog`。

有关更多信息，请参阅《Amazon Athena 用户指南》中的向资源添加标签：tag-resource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源中删除标签

以下 `untag-resource` 示例从 `dynamo_db_catalog` 数据目录资源中移除 `Specialization` 和 `Focus` 键及其相关值。

```
aws athena untag-resource \
    --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/
dynamo_db_catalog \
    --tag-keys Specialization Focus
```

此命令不生成任何输出。要查看结果，请使用 `list-tags-for-resource` 命令。

有关更多信息，请参阅《Amazon Athena 用户指南》中的从资源中删除标签：untag-resource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-data-catalog

以下代码示例演示了如何使用 `update-data-catalog`。

Amazon CLI

更新数据目录

以下 `update-data-catalog` 示例更新 Lambda 函数和 `cw_logs_catalog` 数据目录的说明。

```
aws athena update-data-catalog \
    --name cw_logs_catalog \
    --type LAMBDA \
    --description "New CloudWatch Logs Catalog" \
    --function=arn:aws:lambda:us-west-2:111122223333:function:new_cw_logs_lambda
```

此命令不生成任何输出。要查看结果，请使用 `aws athena get-data-catalog --name cw_logs_catalog`。

有关更多信息，请参阅《Amazon Athena 用户指南》中的更新目录：update-data-catalog。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDataCatalog。

update-work-group

以下代码示例演示了如何使用 `update-work-group`。

Amazon CLI

更新工作组

以下 `update-work-group` 示例禁用 `Data_Analyst_Group` 工作组。用户无法在禁用的工作组中运行或创建查询，但仍可以查看指标、数据使用限制控制、工作组设置、查询历史记录和保存的查询。

```
aws athena update-work-group \
    --work-group Data_Analyst_Group \
    --state DISABLED
```

此命令不生成任何输出。要验证状态的变化，请使用 `aws athena get-work-group --work-group Data_Analyst_Group` 并检查输出中的 `State` 属性。

有关更多信息，请参阅《Amazon Athena 用户指南》中的管理工作组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWorkGroup。

使用 Amazon CLI 的 Auto Scaling 示例

以下代码示例演示了如何通过将 Amazon Command Line Interface与 Auto Scaling 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

attach-instances

以下代码示例演示了如何使用 attach-instances。

Amazon CLI

将实例附加到自动扩缩组

此示例将指定的实例附加到指定的自动扩缩组。

```
aws autoscaling attach-instances \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachInstances。

attach-load-balancer-target-groups

以下代码示例演示了如何使用 attach-load-balancer-target-groups。

Amazon CLI

将目标组附加到自动扩缩组

此示例将指定目标组附加到指定的自动扩缩组。

```
aws autoscaling attach-load-balancer-target-groups \
    --auto-scaling-group-name my-asg \
```

```
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的弹性负载均衡和 Amazon EC2 Auto Scaling。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachLoadBalancerTargetGroups。

attach-load-balancers

以下代码示例演示了如何使用 attach-load-balancers。

Amazon CLI

将经典负载均衡器附加到自动扩缩组

此示例将指定的经典负载均衡器附加到指定的自动扩缩组。

```
aws autoscaling attach-load-balancers \
    --load-balancer-names my-load-balancer \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的弹性负载均衡和 Amazon EC2 Auto Scaling。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachLoadBalancers。

cancel-instance-refresh

以下代码示例演示了如何使用 cancel-instance-refresh。

Amazon CLI

取消实例刷新

以下 cancel-instance-refresh 示例取消指定自动扩缩组正在进行的实例刷新。

```
aws autoscaling cancel-instance-refresh \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的取消实例刷新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelInstanceRefresh。

`complete-lifecycle-action`

以下代码示例演示了如何使用 `complete-lifecycle-action`。

Amazon CLI

完成生命周期操作

此示例通知 Amazon EC2 Auto Scaling 指定的生命周期操作已完成，因此它可以完成启动或终止实例。

```
aws autoscaling complete-lifecycle-action \
    --lifecycle-hook-name my-launch-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-action-result CONTINUE \
    --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteLifecycleAction。

`create-auto-scaling-group`

以下代码示例演示了如何使用 `create-auto-scaling-group`。

Amazon CLI

示例 1：创建自动扩缩组

以下 create-auto-scaling-group 示例在区域内多个可用区中的子网中创建自动扩缩组。实例以指定启动模板的默认版本启动。请注意，大多数其他设置都使用默认值，例如，终止策略和运行状况检查配置。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的自动扩缩组。

示例 2：附加应用程序负载均衡器、网络负载均衡器或网关负载均衡器

此示例为支持预期流量的负载均衡器指定目标组的 ARN。运行状况检查类型指定 ELB，以便在 Elastic Load Balancing 报告实例运行状况不佳时，自动扩缩组将取代它。该命令还定义了以 600 秒为单位的运行状况检查宽限期。宽限期有助于防止新启动的实例过早终止。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/943f017f100becff \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Elastic Load Balancing 和 Amazon EC2 Auto Scaling。

示例 3：指定置放群组，并使用最新版本的启动模板

此示例将实例启动到单个可用区中的置放群组。这对于具有 HPC 工作负载的低延迟群组很有用。此示例还将指定群组的最小大小、最大大小和所需容量。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest' \
    --min-size 1 \
    --max-size 5 \
    --desired-capacity 3 \
    --placement-group my-placement-group \
    --vpc-zone-identifier "subnet-6194ea3b"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 用户指南（适用于 Linux 实例）》中的置放群组。

示例 4：指定单个实例自动扩缩组，并使用特定版本的启动模板

此示例创建一个自动扩缩组，并将其最小和最大容量均设置为 1 以强制运行一个实例。该命令还指定了启动模板的 v1，其中指定了现有 ENI 的 ID。当您使用为 eth0 指定现有 ENI 的启动模板时，必须为自动扩缩组指定与网络接口匹配的可用区，而无需在请求中同时指定子网 ID。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1' \
    --min-size 1 \
    --max-size 1 \
    --availability-zones us-west-2a
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的自动扩缩组。

示例 5：指定不同的终止策略

此示例使用启动配置创建自动扩缩组，并将终止策略设置为首先终止最旧的实例。该命令还将标签应用于该组及其实例，其密钥为 Role，值为 WebServer。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-configuration-name my-lc \
    --min-size 1 \
    --max-size 5 \
    --termination-policies "OldestInstance" \
```

```
    --tags "ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Role,Value=WebServer,PropagateAtLaunch=true" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的使用 Amazon EC2 Auto Scaling 终止策略。

示例 6：指定启动生命周期挂钩

此示例将使用生命周期挂钩创建一个自动扩缩组，该挂钩支持在实例启动时的自定义操作。

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

config.json 文件的内容：

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-launch-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
        "NotificationTargetARN": "arn:aws:sqs:us-west-2:123456789012:my-sqs-queue",
        "RoleARN": "arn:aws:iam::123456789012:role/my-notification-role",
        "NotificationMetadata": "SQS message metadata",
        "HeartbeatTimeout": 4800,
        "DefaultResult": "ABANDON"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": [{
        "ResourceType": "auto-scaling-group",
        "ResourceId": "my-asg",
        "PropagateAtLaunch": true,
        "Value": "test",
        "Key": "environment"
    }]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

示例 7：指定终止生命周期挂钩

此示例将使用生命周期挂钩创建一个自动扩缩组，该挂钩支持在实例终止时的自定义操作。

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

的内容config.json：

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-termination-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
        "HeartbeatTimeout": 120,
        "DefaultResult": "CONTINUE"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "TargetGroupARNs": [
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/73e2d6bc24d8a067"
    ],
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

示例 8：指定自定义终止策略

此示例创建一个自动扩缩组，该组指定自定义 Lambda 函数终止策略，以指示 Amazon EC2 Auto Scaling 哪些实例可以安全地在横向缩减时终止。

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling \
    --min-size 1 \
    --max-size 5 \
    --termination-policies "arn:aws:lambda:us-
west-2:123456789012:function:HelloFunction:prod" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的通过 Lambda 创建自定义终止策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAutoScalingGroup。

create-launch-configuration

以下代码示例演示了如何使用 create-launch-configuration。

Amazon CLI

示例 1：创建启动配置

此示例创建一个简单的启动配置。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的创建启动配置。

示例 2：使用安全组、密钥对和引导脚本创建启动配置

此示例使用用户数据中包含的安全组、密钥对和引导脚本创建启动配置。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
```

```
    --security-groups sg-eb2af88example \
    --key-name my-key-pair \
    --user-data file://myuserdata.txt
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的创建启动配置。

示例 3：使用 IAM 角色创建启动配置

此示例使用 IAM 角色的实例配置文件名称创建启动配置。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --iam-instance-profile my-autoscaling-role
```

此命令不生成任何输出。

有关更多信息，请参阅 Amazon EC2 Auto Scaling 用户指南 中的针对在 Amazon EC2 实例中运行的应用程序的 IAM 角色。

示例 4：创建启用详细监控的启动配置

此示例创建启用 EC2 详细监控的启动配置，该监控在 1 分钟内将 EC2 指标发送到 CloudWatch。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --instance-monitoring Enabled=true
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的配置自动扩缩实例的监控。

示例 5：创建启动竞价型实例的启动配置

此示例创建使用竞价型实例作为唯一购买选项的启动配置。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
```

```
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --spot-price "0.50"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的请求竞价型实例。

示例 6：使用 EC2 实例创建启动配置

此示例基于现有实例的属性创建启动配置。它通过包含 `--placement-tenancy` 和 `--no-associate-public-ip-address` 选项来覆盖置放租赁以及是否设置公有 IP 地址。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc-from-instance \
    --instance-id i-0123a456700123456 \
    --instance-type m5.large \
    --no-associate-public-ip-address \
    --placement-tenancy dedicated
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的使用 EC2 实例创建启动配置。

示例 7：为 Amazon EBS 卷创建具有块设备映射的启动配置

此示例为设备名称为 `/dev/sdh` 和卷大小为 20 的 Amazon EBS gp3 卷创建具有块设备映射的启动配置。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/sdh","Ebs":
{"VolumeSize":20,"VolumeType":"gp3"}}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling API Reference》中的 EBS。

有关引用 JSON 格式参数值的信息，请参阅《Amazon 命令行界面用户指南》中的在 Amazon CLI 中使用字符串的引号。

示例 8：为实例存储卷创建具有块设备映射的启动配置

此示例创建一个启动配置，将 ephemeral1 作为实例存储卷，设备名称为 /dev/sdc。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/sdc","VirtualName":"ephemeral1"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling API Reference》中的 BlockDeviceMapping。

有关引用 JSON 格式参数值的信息，请参阅《Amazon 命令行界面用户指南》中的在 Amazon CLI 中使用字符串的引号。

示例 9：创建启动配置并禁止在启动时附加块设备

此示例创建一个启动配置，用于禁止通过 AMI 的块设备映射指定的块设备（例如，/dev/sdf）。

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/sdf","NoDevice":""}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling API Reference》中的 BlockDeviceMapping。

有关引用 JSON 格式参数值的信息，请参阅《Amazon 命令行界面用户指南》中的在 Amazon CLI 中使用字符串的引号。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLaunchConfiguration。

create-or-update-tags

以下代码示例演示了如何使用 create-or-update-tags。

Amazon CLI

创建或更新自动扩缩组的标签

此示例向指定的自动扩缩组添加两个标签。

```
aws autoscaling create-or-update-tags \
    --tags ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Role,Value=WebServer,PropagateAtLaunch=true ResourceId=my-
asg,ResourceType=auto-scaling-group,Key=Dept,Value=Research,PropagateAtLaunch=true
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的标记自动扩缩组和实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOrUpdateTags。

delete-auto-scaling-group

以下代码示例演示了如何使用 delete-auto-scaling-group。

Amazon CLI

示例 1：删除指定的自动扩缩组

此示例将删除指定的自动扩缩组。

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的删除自动扩缩基础设施。

示例 2：强制删除指定的自动扩缩组

要在不等待自动扩缩组中的实例终止的情况下删除该组，请使用 --force-delete 选项。

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --force-delete
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的删除自动扩缩基础设施。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAutoScalingGroup。

delete-launch-configuration

以下代码示例演示了如何使用 delete-launch-configuration。

Amazon CLI

删除启动配置

此示例删除指定的启动配置。

```
aws autoscaling delete-launch-configuration \
    --launch-configuration-name my-launch-config
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的删除自动扩缩基础设施。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLaunchConfiguration。

delete-lifecycle-hook

以下代码示例演示了如何使用 delete-lifecycle-hook。

Amazon CLI

删除生命周期挂钩

此示例删除指定的生命周期挂钩。

```
aws autoscaling delete-lifecycle-hook \
    --lifecycle-hook-name my-lifecycle-hook \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLifecycleHook。

delete-notification-configuration

以下代码示例演示了如何使用 delete-notification-configuration。

Amazon CLI

删除 Auto Scaling 通知

此示例删除指定自动扩缩组的指定通知。

```
aws autoscaling delete-notification-configuration \
    --auto-scaling-group-name my-asg \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的删除通知配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNotificationConfiguration。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除扩展策略

此示例删除指定的扩展策略。

```
aws autoscaling delete-policy \
    --auto-scaling-group-name my-asg \
    --policy-name alb1000-target-tracking-scaling-policy
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

delete-scheduled-action

以下代码示例演示了如何使用 delete-scheduled-action。

Amazon CLI

从自动扩缩组中删除计划的操作

此示例从指定的自动扩缩组删除指定的计划操作。

```
aws autoscaling delete-scheduled-action \
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-scheduled-action
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScheduledAction。

delete-tags

以下代码示例演示了如何使用 delete-tags。

Amazon CLI

从自动扩缩组中删除标签

此示例从指定的自动扩缩组删除指定的标签。

```
aws autoscaling delete-tags \
    --tags ResourceId=my-asg,ResourceType=auto-scaling-group,Key=Dept,Value=Research
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的标记自动扩缩组和实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTags。

delete-warm-pool

以下代码示例演示了如何使用 delete-warm-pool。

Amazon CLI

示例 1：删除热池

以下示例删除指定自动扩缩组的热池。

```
aws autoscaling delete-warm-pool \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 的热池。

示例 2：强制删除热池

要在不等待其实例终止的情况下删除热池，请使用 `--force-delete` 选项。

```
aws autoscaling delete-warm-pool \
    --auto-scaling-group-name my-asg \
    --force-delete
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 的热池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWarmPool。

`describe-account-limits`

以下代码示例演示了如何使用 `describe-account-limits`。

Amazon CLI

描述您的 Amazon EC2 Auto Scaling 账户限制

此示例描述您的 Amazon 账户的 Amazon EC2 Auto Scaling 限制。

```
aws autoscaling describe-account-limits
```

输出：

```
{
    "NumberOfLaunchConfigurations": 5,
    "MaxNumberOfLaunchConfigurations": 100,
    "NumberOfAutoScalingGroups": 3,
    "MaxNumberOfAutoScalingGroups": 20
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 服务配额。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountLimits。

describe-adjustment-types

以下代码示例演示了如何使用 describe-adjustment-types。

Amazon CLI

描述可用的扩展调整类型

此示例描述可用的调整类型。

```
aws autoscaling describe-adjustment-types
```

输出：

```
{
    "AdjustmentTypes": [
        {
            "AdjustmentType": "ChangeInCapacity"
        },
        {
            "AdjustmentType": "ExactCapacity"
        },
        {
            "AdjustmentType": "PercentChangeInCapacity"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的扩展调整类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAdjustmentTypes。

describe-auto-scaling-groups

以下代码示例演示了如何使用 describe-auto-scaling-groups。

Amazon CLI

示例 1：描述指定的自动扩缩组

此示例将描述指定的自动扩缩组。

```
aws autoscaling describe-auto-scaling-groups \
    --auto-scaling-group-names my-asg
```

输出：

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-1234567890abcde12"
            },
            "MinSize": 0,
            "MaxSize": 1,
            "DesiredCapacity": 1,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "us-west-2a",
                "us-west-2b",
                "us-west-2c"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": 0,
            "Instances": [
                {
                    "InstanceId": "i-06905f55584de02da",
                    "InstanceType": "t2.micro",
                    "AvailabilityZone": "us-west-2a",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService",
                    "ProtectedFromScaleIn": false,
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-1234567890abcde12"
                    }
```

```
                }
            ],
            "CreatedTime": "2023-10-28T02:39:22.152Z",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
            ],
            "NewInstancesProtectedFromScaleIn": false,
            "ServiceLinkedRoleARN":"arn",
            "TrafficSources": []
        }
    ]
}
```

示例 2：描述前 100 个指定的自动扩缩组

此示例将描述指定的自动扩缩组。它允许您指定最多 100 个组名称。

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 100 \
    --auto-scaling-group-names "group1" "group2" "group3" "group4"
```

有关输出示例，请参阅示例 1。

示例 3：描述指定区域中的自动扩缩组

此示例将描述指定区域中的自动扩缩组（最多 75 个组）。

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 75 \
    --region us-east-1
```

有关输出示例，请参阅示例 1。

示例 4：描述指定数量的自动扩缩组

要返回特定数量的自动扩缩组，请使用 `--max-items` 选项。

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 1
```

有关输出示例，请参阅示例 1。

如果输出包含 NextToken 字段，则可描述更多组。要获取其他组，请在后续调用中使用此字段的值和 --starting-token 选项，如下所示。

```
aws autoscaling describe-auto-scaling-groups \
    --starting-token Z3M3LMPEXAMPLE
```

有关输出示例，请参阅示例 1。

示例 5：描述使用启动配置的自动扩缩组

此示例使用 --query 选项描述使用启动配置的自动扩缩组。

```
aws autoscaling describe-auto-scaling-groups \
    --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

输出：

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
        "LaunchConfigurationName": "my-lc",
        "MinSize": 0,
        "MaxSize": 1,
        "DesiredCapacity": 1,
        "DefaultCooldown": 300,
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c"
        ],
        "LoadBalancerNames": [],
        "TargetGroupARNs": [],
        "HealthCheckType": "EC2",
        "HealthCheckGracePeriod": 0,
        "Instances": [
            {
                "InstanceId": "i-088c57934a6449037",
```

```
                "InstanceType": "t2.micro",
                "AvailabilityZone": "us-west-2c",
                "HealthStatus": "Healthy",
                "LifecycleState": "InService",
                "LaunchConfigurationName": "my-lc",
                "ProtectedFromScaleIn": false
            }
        ],
        "CreatedTime": "2023-10-28T02:39:22.152Z",
        "SuspendedProcesses": [],
        "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
        "EnabledMetrics": [],
        "Tags": [],
        "TerminationPolicies": [
            "Default"
        ],
        "NewInstancesProtectedFromScaleIn": false,
        "ServiceLinkedRoleARN":"arn",
        "TrafficSources": []
    }
]
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的筛选 Amazon CLI 输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAutoScalingGroups。

describe-auto-scaling-instances

以下代码示例演示了如何使用 describe-auto-scaling-instances。

Amazon CLI

示例 1：描述一个或多个实例

此示例将描述指定的实例。

```
aws autoscaling describe-auto-scaling-instances \
    --instance-ids i-06905f55584de02da
```

输出：

```
{
    "AutoScalingInstances": [
```

```
        {
            "InstanceId": "i-06905f55584de02da",
            "InstanceType": "t2.micro",
            "AutoScalingGroupName": "my-asg",
            "AvailabilityZone": "us-west-2b",
            "LifecycleState": "InService",
            "HealthStatus": "HEALTHY",
            "ProtectedFromScaleIn": false,
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-1234567890abcde12",
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            }
        }
    ]
}
```

示例 2：描述一个或多个实例

此示例使用 `--max-items` 选项来指定通过此调用返回多少个实例。

```
aws autoscaling describe-auto-scaling-instances \
    --max-items 1
```

如果输出包含 NextToken 字段，可返回更多实例。要获取其他实例，请在后续调用中使用此字段的值和 `--starting-token` 选项，如下所示。

```
aws autoscaling describe-auto-scaling-instances \
    --starting-token Z3M3LMPEXAMPLE
```

有关输出示例，请参阅示例 1。

示例 3：描述使用启动配置的实例

此示例使用 `--query` 选项描述使用启动配置的实例。

```
aws autoscaling describe-auto-scaling-instances \
    --query 'AutoScalingInstances[?LaunchConfigurationName!=`null`]'
```

输出：

```
[
```

```
    {
        "InstanceId": "i-088c57934a6449037",
        "InstanceType": "t2.micro",
        "AutoScalingGroupName": "my-asg",
        "AvailabilityZone": "us-west-2c",
        "LifecycleState": "InService",
        "HealthStatus": "HEALTHY",
        "LaunchConfigurationName": "my-lc",
        "ProtectedFromScaleIn": false
    }
]
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的筛选 Amazon CLI 输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAutoScalingInstances。

describe-auto-scaling-notification-types

以下代码示例演示了如何使用 describe-auto-scaling-notification-types。

Amazon CLI

描述可用的通知类型

此示例描述可用的通知类型。

```
aws autoscaling describe-auto-scaling-notification-types
```

输出：

```
{
    "AutoScalingNotificationTypes": [
        "autoscaling:EC2_INSTANCE_LAUNCH",
        "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
        "autoscaling:EC2_INSTANCE_TERMINATE",
        "autoscaling:EC2_INSTANCE_TERMINATE_ERROR",
        "autoscaling:TEST_NOTIFICATION"
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的在自动扩缩组扩展时获取 Amazon SNS 通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAutoScalingNotificationTypes。

describe-instance-refreshes

以下代码示例演示了如何使用 describe-instance-refreshes。

Amazon CLI

描述实例刷新

以下 describe-instance-refreshes 示例返回指定自动扩缩组的所有实例刷新请求的说明，包括状态消息和（如果有）状态原因。

```
aws autoscaling describe-instance-refreshes \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "InstanceRefreshes": [
        {
            "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b",
            "AutoScalingGroupName": "my-asg",
            "Status": "InProgress",
            "StatusReason": "Waiting for instances to warm up before continuing. For
 example: 0e69cc3f05f825f4f is warming up.",
            "EndTime": "2023-03-23T16:42:55Z",
            "PercentageComplete": 0,
            "InstancesToUpdate": 0,
    "Preferences": {
                "MinHealthyPercentage": 100,
                "InstanceWarmup": 300,
                "CheckpointPercentages": [
                    50
                ],
                "CheckpointDelay": 3600,
                "SkipMatching": false,
                "AutoRollback": true,
                "ScaleInProtectedInstances": "Ignore",
                "StandbyInstances": "Ignore"
            }
```

```
        },
        {
            "InstanceRefreshId": "dd7728d0-5bc4-4575-96a3-1b2c52bf8bb1",
            "AutoScalingGroupName": "my-asg",
            "Status": "Successful",
            "EndTime": "2022-06-02T16:53:37Z",
            "PercentageComplete": 100,
            "InstancesToUpdate": 0,
    "Preferences": {
                "MinHealthyPercentage": 90,
                "InstanceWarmup": 300,
                "SkipMatching": true,
                "AutoRollback": true,
                "ScaleInProtectedInstances": "Ignore",
                "StandbyInstances": "Ignore"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的检查实例刷新的状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceRefreshes。

`describe-launch-configurations`

以下代码示例演示了如何使用 `describe-launch-configurations`。

Amazon CLI

示例 1：描述指定的启动配置

此示例描述指定的启动配置。

```
aws autoscaling describe-launch-configurations \
    --launch-configuration-names my-launch-config
```

输出：

```
{
    "LaunchConfigurations": [
        {
```

```
            "LaunchConfigurationName": "my-launch-config",
            "LaunchConfigurationARN": "arn:aws:autoscaling:us-
west-2:123456789012:launchConfiguration:98d3b196-4cf9-4e88-8ca1-8547c24ced8b:launchConfigura
my-launch-config",
            "ImageId": "ami-0528a5175983e7f28",
            "KeyName": "my-key-pair-uswest2",
            "SecurityGroups": [
                "sg-05eaec502fcdadc2e"
            ],
            "ClassicLinkVPCSecurityGroups": [],
            "UserData": "",
            "InstanceType": "t2.micro",
            "KernelId": "",
            "RamdiskId": "",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/xvda",
                    "Ebs": {
                        "SnapshotId": "snap-06c1606ba5ca274b1",
                        "VolumeSize": 8,
                        "VolumeType": "gp2",
                        "DeleteOnTermination": true,
                        "Encrypted": false
                    }
                }
            ],
            "InstanceMonitoring": {
                "Enabled": true
            },
            "CreatedTime": "2020-10-28T02:39:22.321Z",
            "EbsOptimized": false,
            "AssociatePublicIpAddress": true,
            "MetadataOptions": {
                "HttpTokens": "required",
                "HttpPutResponseHopLimit": 1,
                "HttpEndpoint": "disabled"
            }
        }
    ]
}
```

示例 2：描述指定数量的启动配置

要返回特定数量的启动配置，请使用 `--max-items` 选项。

```
aws autoscaling describe-launch-configurations \
    --max-items 1
```

如果输出包含 NextToken 字段，则可返回更多启动配置。要获取其他启动配置，请在后续调用中使用此字段的值和 --starting-token 选项，如下所示。

```
aws autoscaling describe-launch-configurations \
    --starting-token Z3M3LMPEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLaunchConfigurations。

describe-lifecycle-hook-types

以下代码示例演示了如何使用 describe-lifecycle-hook-types。

Amazon CLI

描述可用的生命周期挂钩类型

此示例描述可用的生命周期挂钩类型。

```
aws autoscaling describe-lifecycle-hook-types
```

输出：

```
{
    "LifecycleHookTypes": [
        "autoscaling:EC2_INSTANCE_LAUNCHING",
        "autoscaling:EC2_INSTANCE_TERMINATING"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLifecycleHookTypes。

describe-lifecycle-hooks

以下代码示例演示了如何使用 describe-lifecycle-hooks。

Amazon CLI

描述您的生命周期挂钩

此示例描述指定自动扩缩组的生命周期挂钩。

```
aws autoscaling describe-lifecycle-hooks \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "LifecycleHooks": [
        {
            "GlobalTimeout": 3000,
            "HeartbeatTimeout": 30,
            "AutoScalingGroupName": "my-asg",
            "LifecycleHookName": "my-launch-hook",
            "DefaultResult": "ABANDON",
            "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING"
        },
        {
            "GlobalTimeout": 6000,
            "HeartbeatTimeout": 60,
            "AutoScalingGroupName": "my-asg",
            "LifecycleHookName": "my-termination-hook",
            "DefaultResult": "CONTINUE",
            "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLifecycleHooks。

describe-load-balancer-target-groups

以下代码示例演示了如何使用 describe-load-balancer-target-groups。

Amazon CLI

描述自动扩缩组的负载均衡器目标组

此示例描述附加到指定自动扩缩组的负载均衡器目标组。

```
aws autoscaling describe-load-balancer-target-groups \
```

```
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "LoadBalancerTargetGroups": [
        {
            "LoadBalancerTargetGroupARN": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
            "State": "Added"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancerTargetGroups。

describe-load-balancers

以下代码示例演示了如何使用 describe-load-balancers。

Amazon CLI

描述自动扩缩组的经典负载均衡器

此示例描述指定自动扩缩组的经典负载均衡器。

```
aws autoscaling describe-load-balancers \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "LoadBalancers": [
        {
            "State": "Added",
            "LoadBalancerName": "my-load-balancer"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancers。

describe-metric-collection-types

以下代码示例演示了如何使用 describe-metric-collection-types。

Amazon CLI

描述可用的指标收集类型

此示例描述可用的指标收集类型。

```
aws autoscaling describe-metric-collection-types
```

输出：

```
{
    "Metrics": [
        {
            "Metric": "GroupMinSize"
        },
        {
            "Metric": "GroupMaxSize"
        },
        {
            "Metric": "GroupDesiredCapacity"
        },
        {
            "Metric": "GroupInServiceInstances"
        },
        {
            "Metric": "GroupInServiceCapacity"
        },
        {
            "Metric": "GroupPendingInstances"
        },
        {
            "Metric": "GroupPendingCapacity"
        },
        {
            "Metric": "GroupTerminatingInstances"
        },
        {
            "Metric": "GroupTerminatingCapacity"
        },
        {
```

```
            "Metric": "GroupStandbyInstances"
        },
        {
            "Metric": "GroupStandbyCapacity"
        },
        {
            "Metric": "GroupTotalInstances"
        },
        {
            "Metric": "GroupTotalCapacity"
        }
    ],
    "Granularities": [
        {
            "Granularity": "1Minute"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的自动扩缩组指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMetricCollectionTypes。

describe-notification-configurations

以下代码示例演示了如何使用 describe-notification-configurations。

Amazon CLI

示例 1：描述指定组的通知配置

此示例描述指定自动扩缩组的通知配置。

```
aws autoscaling describe-notification-configurations \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "NotificationConfigurations": [
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
```

```
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"
        },
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的在自动扩缩组扩展时获取 Amazon SNS 通知。

示例 1：描述指定数量的通知配置

要返回特定数量的通知配置，请使用 `max-items` 参数。

```
aws autoscaling describe-notification-configurations \
    --auto-scaling-group-name my-auto-scaling-group \
    --max-items 1
```

输出：

```
{
    "NotificationConfigurations": [
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"
        },
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"
        }
    ]
}
```

如果输出包含 `NextToken` 字段，可返回更多通知配置。要获取其他通知配置，请在后续调用中使用此字段的值和 `starting-token` 选项，如下所示。

```
aws autoscaling describe-notification-configurations \
```

```
    --auto-scaling-group-name my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的在自动扩缩组扩展时获取 Amazon SNS 通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNotificationConfigurations。

describe-policies

以下代码示例演示了如何使用 describe-policies。

Amazon CLI

示例 1：描述指定组的扩展策略

此示例描述指定自动扩缩组的扩展策略。

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "ScalingPolicies": [
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "alb1000-target-tracking-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:3065d9c8-9969-4bec-
bb6a-3fbe5550fde6:autoScalingGroupName/my-asg:policyName/alb1000-target-tracking-
scaling-policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196"
                },
```

```
                {
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-f96f899d-b8e7-4d09-
a010-c1aaa35da296",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-f96f899d-b8e7-4d09-a010-
c1aaa35da296"
                }
            ],
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ALBRequestCountPerTarget",
                    "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-
alb-target-group/943f017f100becff"
                },
                "TargetValue": 1000.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        },
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "cpu40-target-tracking-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:5fd26f71-39d4-4690-82a9-
b8515c45cdde:autoScalingGroupName/my-asg:policyName/cpu40-target-tracking-scaling-
policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-139f9789-37b9-42ad-bea5-b5b147d7f473",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-139f9789-37b9-42ad-bea5-
b5b147d7f473"
                },
                {
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-bd681c67-
fc18-4c56-8468-fb8e413009c9",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-bd681c67-fc18-4c56-8468-
fb8e413009c9"
                }
            ],
```

```
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ASGAverageCPUUtilization"
                },
                "TargetValue": 40.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的动态扩展。

示例 2：描述指定名称的扩展策略

要返回特定的扩展策略，请使用 `--policy-names` 选项。

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg \
    --policy-names cpu40-target-tracking-scaling-policy
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的动态扩展。

示例 3：描述多种扩展策略

要返回特定数量的策略，请使用 `--max-items` 选项。

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg \
    --max-items 1
```

有关输出示例，请参阅示例 1。

如果输出包含 `NextToken` 字段，则请在后续调用中使用此字段的值和 `--starting-token` 选项获取其他策略。

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg --starting-
token Z3M3LMPEXAMPLE
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的动态扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePolicies。

describe-scaling-activities

以下代码示例演示了如何使用 describe-scaling-activities。

Amazon CLI

示例 1：描述指定组的扩展活动

此示例描述指定自动扩缩组的扩展活动。

```
aws autoscaling describe-scaling-activities \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",
            "Description": "Launching a new EC2 instance: i-0d44425630326060f",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2020-10-30T19:35:51Z a user request update of
 AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the desired
 capacity from 0 to 16.  At 2020-10-30T19:36:07Z an instance was started in response
 to a difference between desired and actual capacity, increasing the capacity from 0
 to 16.",
            "StartTime": "2020-10-30T19:36:09.766Z",
            "EndTime": "2020-10-30T19:36:41Z",
            "StatusCode": "Successful",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的验证自动扩缩组的扩展活动。

示例 2：描述已删除组的扩展活动

要在删除自动扩缩组后描述扩展活动，请添加 `--include-deleted-groups` 选项。

```
aws autoscaling describe-scaling-activities \
    --auto-scaling-group-name my-asg \
    --include-deleted-groups
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "e1f5de0e-f93e-1417-34ac-092a76fba220",
            "Description": "Launching a new EC2 instance.  Status Reason: Your Spot
 request price of 0.001 is lower than the minimum required Spot request fulfillment
 price of 0.0031. Launching EC2 instance failed.",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2021-01-13T20:47:24Z a user request update of
 AutoScalingGroup constraints to min: 1, max: 5, desired: 3 changing the desired
 capacity from 0 to 3.  At 2021-01-13T20:47:27Z an instance was started in response
 to a difference between desired and actual capacity, increasing the capacity from 0
 to 3.",
            "StartTime": "2021-01-13T20:47:30.094Z",
            "EndTime": "2021-01-13T20:47:30Z",
            "StatusCode": "Failed",
            "StatusMessage": "Your Spot request price of 0.001 is lower than the
 minimum required Spot request fulfillment price of 0.0031. Launching EC2 instance
 failed.",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}",
            "AutoScalingGroupState": "Deleted",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的排除 Amazon EC2 Auto Scaling 问题。

示例 3：描述指定数量的扩展活动

要返回特定数量的活动，请使用 --max-items 选项。

```
aws autoscaling describe-scaling-activities \
    --max-items 1
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",
            "Description": "Launching a new EC2 instance: i-0d44425630326060f",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2020-10-30T19:35:51Z a user request update of
 AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the desired
 capacity from 0 to 16.  At 2020-10-30T19:36:07Z an instance was started in response
 to a difference between desired and actual capacity, increasing the capacity from 0
 to 16.",
            "StartTime": "2020-10-30T19:36:09.766Z",
            "EndTime": "2020-10-30T19:36:41Z",
            "StatusCode": "Successful",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}"
        }
    ]
}
```

如果输出包含 NextToken 字段，可返回更多活动。要获取其他活动，请在后续调用中使用此字段的值和 --starting-token 选项，如下所示。

```
aws autoscaling describe-scaling-activities \
    --starting-token Z3M3LMPEXAMPLE
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的验证自动扩缩组的扩展活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingActivities。

describe-scaling-process-types

以下代码示例演示了如何使用 describe-scaling-process-types。

Amazon CLI

描述可用的流程类型

此示例描述可用的流程类型。

```
aws autoscaling describe-scaling-process-types
```

输出：

```
{
    "Processes": [
        {
            "ProcessName": "AZRebalance"
        },
        {
            "ProcessName": "AddToLoadBalancer"
        },
        {
            "ProcessName": "AlarmNotification"
        },
        {
            "ProcessName": "HealthCheck"
        },
        {
            "ProcessName": "InstanceRefresh"
        },
        {
            "ProcessName": "Launch"
        },
        {
            "ProcessName": "ReplaceUnhealthy"
        },
        {
            "ProcessName": "ScheduledActions"
        },
        {
            "ProcessName": "Terminate"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的暂停和恢复扩展流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingProcessTypes。

describe-scheduled-actions

以下代码示例演示了如何使用 describe-scheduled-actions。

Amazon CLI

示例 1：描述所有计划操作

此示例描述所有计划操作。

```
aws autoscaling describe-scheduled-actions
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 2：描述指定组的计划操作

要描述特定自动扩缩组的计划操作，请使用 --auto-scaling-group-name 选项。

```
aws autoscaling describe-scheduled-actions \
```

```
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 3：描述指定的计划操作

要描述特定的计划操作，请使用 `--scheduled-action-names` 选项。

```
aws autoscaling describe-scheduled-actions \
    --scheduled-action-names my-recurring-action
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
```

```
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 4：描述具有指定开始时间的计划操作

要描述在特定时间开始的计划操作，请使用 `--start-time` 选项。

```
aws autoscaling describe-scheduled-actions \
    --start-time "2023-12-01T04:00:00Z"
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 5：描述在指定时间结束的计划操作

要描述在特定时间结束的计划操作，请使用 `--end-time` 选项。

```
aws autoscaling describe-scheduled-actions \
    --end-time "2023-12-01T04:00:00Z"
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 6：描述指定数量的计划操作

要返回特定数量的计划操作，请使用 `--max-items` 选项。

```
aws autoscaling describe-scheduled-actions \
    --auto-scaling-group-name my-asg \
    --max-items 1
```

输出：

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
```

```
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

如果输出包含 NextToken 字段，可返回更多计划操作。要获取其他计划操作，请在后续调用中使用此字段的值和 --starting-token 选项，如下所示。

```
aws autoscaling describe-scheduled-actions \
    --auto-scaling-group-name my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledActions。

describe-tags

以下代码示例演示了如何使用 describe-tags。

Amazon CLI

描述所有标签

此示例描述所有标签。

```
aws autoscaling describe-tags
```

输出：

```
{
```

```
    "Tags": [
        {
            "ResourceType": "auto-scaling-group",
            "ResourceId": "my-asg",
            "PropagateAtLaunch": true,
            "Value": "Research",
            "Key": "Dept"
        },
        {
            "ResourceType": "auto-scaling-group",
            "ResourceId": "my-asg",
            "PropagateAtLaunch": true,
            "Value": "WebServer",
            "Key": "Role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的标记自动扩缩组和实例。

示例 2：描述指定组的标签

要描述特定自动扩缩组的标签，请使用 `--filters` 选项。

```
aws autoscaling describe-tags --filters Name=auto-scaling-group,Values=my-asg
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的标记自动扩缩组和实例。

示例 3：描述指定数量的标签

要返回特定数量的标签，请使用 `--max-items` 选项。

```
aws autoscaling describe-tags \
    --max-items 1
```

如果输出包含 `NextToken` 字段，则可返回更多标签。要获取其他标签，请在后续调用中使用此字段的值和 `--starting-token` 选项，如下所示。

```
aws autoscaling describe-tags \
    --filters Name=auto-scaling-group,Values=my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的标记自动扩缩组和实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

describe-termination-policy-types

以下代码示例演示了如何使用 describe-termination-policy-types。

Amazon CLI

描述可用的终止策略类型

此示例描述可用的终止策略类型。

```
aws autoscaling describe-termination-policy-types
```

输出：

```
{
    "TerminationPolicyTypes": [
        "AllocationStrategy",
        "ClosestToNextInstanceHour",
        "Default",
        "NewestInstance",
        "OldestInstance",
        "OldestLaunchConfiguration",
        "OldestLaunchTemplate"
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的控制在横向缩减期间终止哪些自动扩缩实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTerminationPolicyTypes。

describe-warm-pool

以下代码示例演示了如何使用 describe-warm-pool。

Amazon CLI

描述热池

此示例将描述指定自动扩缩组的热池。

```
aws autoscaling describe-warm-pool \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "WarmPoolConfiguration": {
        "MinSize": 2,
        "PoolState": "Stopped"
    },
    "Instances": [
        {
            "InstanceId": "i-070a5bbc7e7f40dc5",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2c",
            "LifecycleState": "Warmed:Pending",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-00a731f6e9fa48610",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "6"
            }
        },
        {
            "InstanceId": "i-0b52f061814d3bd2d",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2b",
            "LifecycleState": "Warmed:Pending",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-00a731f6e9fa48610",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "6"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 的热池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeWarmPool。

detach-instances

以下代码示例演示了如何使用 detach-instances。

Amazon CLI

将实例与自动扩缩组分离

此示例将指定实例与指定的自动扩缩组分离。

```
aws autoscaling detach-instances \
    --instance-ids i-030017cfa84b20135 \
    --auto-scaling-group-name my-asg \
    --should-decrement-desired-capacity
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "5091cb52-547a-47ce-a236-c9ccbc2cb2c9",
            "AutoScalingGroupName": "my-asg",
            "Description": "Detaching EC2 instance: i-030017cfa84b20135",
            "Cause": "At 2020-10-31T17:35:04Z instance i-030017cfa84b20135 was
 detached in response to a user request, shrinking the capacity from 2 to 1.",
            "StartTime": "2020-04-12T15:02:16.179Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachInstances。

detach-load-balancer-target-groups

以下代码示例演示了如何使用 detach-load-balancer-target-groups。

Amazon CLI

将负载均衡器目标组与自动扩缩组分离

此示例将指定的负载均衡器目标组与指定的自动扩缩组分离。

```
aws autoscaling detach-load-balancer-target-groups \
    --auto-scaling-group-name my-asg \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

此命令不生成任何输出

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的将负载均衡器挂接到自动扩缩组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachLoadBalancerTargetGroups。

detach-load-balancers

以下代码示例演示了如何使用 detach-load-balancers。

Amazon CLI

将经典负载均衡器与自动扩缩组分离

此示例将指定的经典负载均衡器与指定的自动扩缩组分离。

```
aws autoscaling detach-load-balancers \
    --load-balancer-names my-load-balancer \
    --auto-scaling-group-name my-asg
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的将负载均衡器挂接到自动扩缩组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachLoadBalancers。

disable-metrics-collection

以下代码示例演示了如何使用 disable-metrics-collection。

Amazon CLI

禁用自动扩缩组指标收集

此示例禁用指定自动扩缩组的 GroupDesiredCapacity 指标的收集。

```
aws autoscaling disable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --metrics GroupDesiredCapacity
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的监控自动扩缩组和实例的 CloudWatch 指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableMetricsCollection。

enable-metrics-collection

以下代码示例演示了如何使用 enable-metrics-collection。

Amazon CLI

示例 1：启用自动扩缩组的指标收集

此示例启用指定自动扩缩组的数据收集。

```
aws autoscaling enable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --granularity "1Minute"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的监控自动扩缩组和实例的 CloudWatch 指标。

示例 2：收集自动扩缩组指定指标的相关数据

要收集特定指标的相关数据，请使用 --metrics 选项。

```
aws autoscaling enable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --metrics GroupDesiredCapacity --granularity "1Minute"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的监控自动扩缩组和实例的 CloudWatch 指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableMetricsCollection。

enter-standby

以下代码示例演示了如何使用 enter-standby。

Amazon CLI

将实例移入备用模式

此示例将指定实例置于备用模式。这对于更新当前正在使用的实例或对其进行问题排查非常有用。

```
aws autoscaling enter-standby \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg \
    --should-decrement-desired-capacity
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "ffa056b4-6ed3-41ba-ae7c-249dfae6eba1",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance to Standby: i-061c63c5eb45f0416",
            "Cause": "At 2020-10-31T20:31:00Z instance i-061c63c5eb45f0416 was moved
 to standby in response to a user request, shrinking the capacity from 1 to 0.",
            "StartTime": "2020-10-31T20:31:00.949Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 实例生命周期。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnterStandby。

execute-policy

以下代码示例演示了如何使用 execute-policy。

Amazon CLI

执行扩展策略

此示例将对指定的自动扩缩组执行名为 my-step-scale-out-policy 的扩展策略。

```
aws autoscaling execute-policy \
    --auto-scaling-group-name my-asg \
    --policy-name  my-step-scale-out-policy \
    --metric-value 95 \
    --breach-threshold 80
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的步进和简单扩展策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExecutePolicy。

exit-standby

以下代码示例演示了如何使用 exit-standby。

Amazon CLI

将实例移出备用模式

此示例将指定的实例移出备用模式。

```
aws autoscaling exit-standby \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg
```

输出：

```
{
```

```
    "Activities": [
        {
            "ActivityId": "142928e1-a2dc-453a-9b24-b85ad6735928",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance out of Standby:
 i-061c63c5eb45f0416",
            "Cause": "At 2020-10-31T20:32:50Z instance i-061c63c5eb45f0416 was moved
 out of standby in response to a user request, increasing the capacity from 0 to
 1.",
            "StartTime": "2020-10-31T20:32:50.222Z",
            "StatusCode": "PreInService",
            "Progress": 30,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的临时从自动扩缩组中移除实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExitStandby。

put-lifecycle-hook

以下代码示例演示了如何使用 put-lifecycle-hook。

Amazon CLI

示例 1：创建生命周期挂钩

此示例创建一个生命周期挂钩，该挂钩将在任何新启动的实例上调用，超时时间为 4800 秒。这对于在用户数据脚本完成之前保持实例处于等待状态，或使用 EventBridge 调用 Amazon Lambda 函数很有用。

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --heartbeat-timeout 4800
```

此命令不生成任何输出。如果已存在同名的生命周期挂钩，则该挂钩将被新的生命周期挂钩覆盖。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

示例 2：发送 Amazon SNS 电子邮件以通知您实例状态转换

此示例创建一个包含 Amazon SNS 主题和 IAM 角色的生命周期挂钩，用于在实例启动时接收通知。

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --notification-target-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \
    --role-arn arn:aws:iam::123456789012:role/my-auto-scaling-role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

示例 3：向 Amazon SQS 队列发布消息

此示例创建一个生命周期挂钩，该挂钩将包含元数据的消息发布到指定的 Amazon SQS 队列。

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --notification-target-arn arn:aws:sqs:us-west-2:123456789012:my-sqs-queue \
    --role-arn arn:aws:iam::123456789012:role/my-notification-role \
    --notification-metadata "SQS message metadata"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLifecycleHook。

put-notification-configuration

以下代码示例演示了如何使用 put-notification-configuration。

Amazon CLI

添加通知

此示例将指定的通知添加到指定的自动扩缩组。

```
aws autoscaling put-notification-configuration \
    --auto-scaling-group-name my-asg \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \
    --notification-type autoscaling:TEST_NOTIFICATION
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的在自动扩缩组扩展时获取 Amazon SNS 通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutNotificationConfiguration。

put-scaling-policy

以下代码示例演示了如何使用 put-scaling-policy。

Amazon CLI

向自动扩缩组添加目标跟踪扩展策略

以下 put-scaling-policy 示例将目标跟踪扩展策略应用到指定的自动扩缩组。输出包含代表您创建的两个 CloudWatch 警报的 ARN 和名称。如果已存在同名扩缩策略，则该扩展策略将被新扩展策略覆盖。

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
  --policy-name alb1000-target-tracking-scaling-policy \
  --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

的内容config.json：

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
          "PredefinedMetricType": "ALBRequestCountPerTarget",
          "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-target-
group/943f017f100becff"
```

```
    }
}
```

输出：

```
{
    "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:228f02c2-
c665-4bfd-aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/alb1000-target-
tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
my-asg-AlarmHigh-fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
my-asg-AlarmLow-61a39305-ed0c-47af-bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
 }
```

有关更多示例，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon 命令行界面（Amazon CLI）的扩展策略示例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutScalingPolicy。

put-scheduled-update-group-action

以下代码示例演示了如何使用 put-scheduled-update-group-action。

Amazon CLI

示例 1：向自动扩缩组添加计划操作

此示例将指定的计划操作添加到指定的自动扩缩组。

```
aws autoscaling put-scheduled-update-group-action \
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-scheduled-action \
```

```
    --start-time "2023-05-12T08:00:00Z" \
    --min-size 2 \
    --max-size 6 \
    --desired-capacity 4
```

此命令不生成任何输出。如果已存在同名的计划操作，则该计划操作将被新计划操作覆盖。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

示例 2：指定周期性计划

此示例创建一个计划操作，以按计划在每年一月、六月和十二月的第一天 00:30 执行的周期性计划进行扩展。

```
aws autoscaling put-scheduled-update-group-action \
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-recurring-action \
    --recurrence "30 0 1 1,6,12 *" \
    --min-size 2 \
    --max-size 6 \
    --desired-capacity 4
```

此命令不生成任何输出。如果已存在同名的计划操作，则该计划操作将被新计划操作覆盖。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的计划扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutScheduledUpdateGroupAction。

put-warm-pool

以下代码示例演示了如何使用 put-warm-pool。

Amazon CLI

创建热池

以下示例为指定的自动扩缩组创建热池。

```
aws autoscaling put-warm-pool \
    --auto-scaling-group-name my-asg \
    --min-size 2
```

此命令不生成任何输出。如果已经存在热池，则会对其进行更新。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 的热池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutWarmPool。

record-lifecycle-action-heartbeat

以下代码示例演示了如何使用 record-lifecycle-action-heartbeat。

Amazon CLI

记录生命周期操作心跳

此示例记录生命周期操作心跳，以使实例保持待处理状态。

```
aws autoscaling record-lifecycle-action-heartbeat \
    --lifecycle-hook-name my-launch-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Amazon EC2 Auto Scaling 生命周期挂钩。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RecordLifecycleActionHeartbeat。

resume-processes

以下代码示例演示了如何使用 resume-processes。

Amazon CLI

恢复暂停的流程

此示例恢复指定自动扩缩组的指定暂停扩展流程。

```
aws autoscaling resume-processes \
    --auto-scaling-group-name my-asg \
    --scaling-processes AlarmNotification
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的暂停和恢复扩展流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResumeProcesses。

rollback-instance-refresh

以下代码示例演示了如何使用 rollback-instance-refresh。

Amazon CLI

回滚实例刷新

以下 rollback-instance-refresh 示例回滚指定自动扩缩组正在进行的实例刷新。

```
aws autoscaling rollback-instance-refresh \
    --auto-scaling-group-name my-asg
```

输出：

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的通过回滚撤消更改。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RollbackInstanceRefresh。

set-desired-capacity

以下代码示例演示了如何使用 set-desired-capacity。

Amazon CLI

为自动扩缩组设置所需容量

此示例为指定的自动扩缩组设置所需容量。

```
aws autoscaling set-desired-capacity \
    --auto-scaling-group-name my-asg \
    --desired-capacity 2 \
    --honor-cooldown
```

如果成功，该命令将返回到提示符。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetDesiredCapacity。

set-instance-health

以下代码示例演示了如何使用 set-instance-health。

Amazon CLI

设置实例的运行状况

此示例将指定实例的运行状况设置为 Unhealthy。

```
aws autoscaling set-instance-health \
    --instance-id i-061c63c5eb45f0416 \
    --health-status Unhealthy
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetInstanceHealth。

set-instance-protection

以下代码示例演示了如何使用 set-instance-protection。

Amazon CLI

示例 1：启用实例的实例保护设置

此示例启用对指定实例的实例保护。

```
aws autoscaling set-instance-protection \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg --protected-from-scale-in
```

此命令不生成任何输出。

示例 2：禁用实例的实例保护设置

此示例禁用对指定实例的实例保护。

```
aws autoscaling set-instance-protection \
```

```
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg \
    --no-protected-from-scale-in
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetInstanceProtection。

start-instance-refresh

以下代码示例演示了如何使用 start-instance-refresh。

Amazon CLI

示例 1：使用命令行参数启动实例刷新

以下 start-instance-refresh 示例使用命令行参数启动实例刷新。可选的 preferences 参数指定 InstanceWarmup 为 60 秒，MinHealthyPercentage 为 50 %。

```
aws autoscaling start-instance-refresh \
    --auto-scaling-group-name my-asg \
    --preferences '{"InstanceWarmup": 60, "MinHealthyPercentage": 50}'
```

输出：

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的启动实例刷新。

示例 2：使用 JSON 文件启动实例刷新

以下 start-instance-refresh 示例使用 JSON 文件启动实例刷新。您可以指定自动扩缩组，并在 JSON 文件中定义所需的配置和首选项，如以下示例所示。

```
aws autoscaling start-instance-refresh \
    --cli-input-json file://config.json
```

的内容config.json：

```
{
```

```
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
        "LaunchTemplate": {
            "LaunchTemplateId": "lt-068f72b729example",
            "Version": "$Default"
        }
    },
    "Preferences": {
        "InstanceWarmup": 60,
        "MinHealthyPercentage": 50,
        "AutoRollback": true,
        "ScaleInProtectedInstances": Ignore,
        "StandbyInstances": Terminate
    }
}
```

输出：

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的启动实例刷新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartInstanceRefresh。

suspend-processes

以下代码示例演示了如何使用 suspend-processes。

Amazon CLI

暂停自动扩缩流程

此示例暂停指定自动扩缩组的指定扩展流程。

```
aws autoscaling suspend-processes \
    --auto-scaling-group-name my-asg \
    --scaling-processes AlarmNotification
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的暂停和恢复扩展流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SuspendProcesses。

terminate-instance-in-auto-scaling-group

以下代码示例演示了如何使用 terminate-instance-in-auto-scaling-group。

Amazon CLI

终止自动扩缩组中的实例

此示例在不更新指定自动扩缩组大小的情况下终止该组中的指定实例。Amazon EC2 Auto Scaling 会在指定实例终止后启动替代实例。

```
aws autoscaling terminate-instance-in-auto-scaling-group \
    --instance-id i-061c63c5eb45f0416 \
    --no-should-decrement-desired-capacity
```

输出：

```
{
    "Activities": [
        {
            "ActivityId": "8c35d601-793c-400c-fcd0-f64a27530df7",
            "AutoScalingGroupName": "my-asg",
            "Description": "Terminating EC2 instance: i-061c63c5eb45f0416",
            "Cause": "",
            "StartTime": "2020-10-31T20:34:25.680Z",
            "StatusCode": "InProgress",
            "Progress": 0,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateInstanceInAutoScalingGroup。

update-auto-scaling-group

以下代码示例演示了如何使用 update-auto-scaling-group。

Amazon CLI

示例 1：更新自动扩缩组的大小限制

该示例更新指定的自动扩缩组，该组的最小大小为 1，最大大小为 10。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --min-size 2 \
    --max-size 10
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的设置自动扩缩组的容量限制。

示例 2：添加 Elastic Load Balancing 运行状况检查并指定要使用的可用区和子网

此示例更新指定的自动扩缩组以添加 Elastic Load Balancing 运行状况检查。此命令还会使用多个可用区中的子网 ID 列表更新 `--vpc-zone-identifier` 的值。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的 Elastic Load Balancing 和 Amazon EC2 Auto Scaling。

示例 3：更新置放群组和终止策略

此示例更新要使用的置放群组和终止策略。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --placement-group my-placement-group \
    --termination-policies "OldestInstance"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的自动扩缩组。

示例 4：使用最新版本的启动模板

此示例将指定的自动扩缩组更新为使用最新版本的指定启动模板。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的启动模板。

示例 5：使用特定版本的启动模板

此示例将指定的自动扩缩组更新为使用特定版本的启动模板，而不是最新或默认版本。

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的启动模板。

示例 6：定义混合实例策略并启用容量再平衡

此示例将指定的自动扩缩组更新为使用混合实例策略并启用容量再平衡。此结构允许您指定具有竞价和按需容量的组，并针对不同的架构使用不同的启动模板。

```
aws autoscaling update-auto-scaling-group \
    --cli-input-json file://~/config.json
```

的内容config.json：

```
{
    "AutoScalingGroupName": "my-asg",
    "CapacityRebalance": true,
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template-for-x86",
                "Version": "$Latest"
            },
```

```
            "Overrides": [
                {
                    "InstanceType": "c6g.large",
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateName": "my-launch-template-for-arm",
                        "Version": "$Latest"
                    }
                },
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的具有多个实例类型和购买选项的自动扩缩组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAutoScalingGroup。

使用 Amazon CLI 的 Auto Scaling Plans 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Auto Scaling Plans 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

create-scaling-plan

以下代码示例演示了如何使用 create-scaling-plan。

Amazon CLI

创建扩展计划

以下 create-scaling-plan 示例使用已创建的 JSON 文件（名为 config.json）创建一个名为 my-scaling-plan 的扩展计划。扩展计划的结构包括名为 my-asg 的自动扩缩组的扩展指令。它指定 TagFilters 属性作为应用程序源，并启用预测式扩展和动态扩展。

```
aws autoscaling-plans create-scaling-plan \
    --scaling-plan-name my-scaling-plan \
    --cli-input-json file://~/config.json
```

config.json 文件的内容：

```
{
    "ApplicationSource": {
        "TagFilters": [
            {
                "Key": "purpose",
                "Values": [
                    "my-application"
                ]
            }
        ]
    },
    "ScalingInstructions": [
        {
            "ServiceNamespace": "autoscaling",
            "ResourceId": "autoScalingGroup/my-asg",
            "ScalableDimension": "autoscaling:autoScalingGroup:DesiredCapacity",
            "ScheduledActionBufferTime": 300,
            "PredictiveScalingMaxCapacityBehavior":
 "SetForecastCapacityToMaxCapacity",
            "PredictiveScalingMode": "ForecastAndScale",
```

```
            "PredefinedLoadMetricSpecification": {
                "PredefinedLoadMetricType": "ASGTotalCPUUtilization"
            },
            "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
            "MinCapacity": 1,
            "MaxCapacity": 4,
            "TargetTrackingConfigurations": [
                {
                    "PredefinedScalingMetricSpecification": {
                        "PredefinedScalingMetricType": "ASGAverageCPUUtilization"
                    },
                    "TargetValue": 50
                }
            ]
        }
    ]
}
```

输出：

```
{
"ScalingPlanVersion": 1
}
```

有关更多信息，请参阅 Amazon Auto Scaling 用户指南。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateScalingPlan。

delete-scaling-plan

以下代码示例演示了如何使用 delete-scaling-plan。

Amazon CLI

删除扩展计划

以下 delete-scaling-plan 示例删除指定扩展计划。

```
aws autoscaling-plans delete-scaling-plan \
    --scaling-plan-name my-scaling-plan \
    --scaling-plan-version 1
```

此命令不生成任何输出。

有关更多信息，请参阅 Amazon Auto Scaling 用户指南。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScalingPlan。

describe-scaling-plan-resources

以下代码示例演示了如何使用 describe-scaling-plan-resources。

Amazon CLI

描述扩展计划的可扩展资源

以下 describe-scaling-plan-resources 示例显示与指定扩展计划关联的单个可扩展资源（自动扩缩组）的详细信息。

```
aws autoscaling-plans describe-scaling-plan-resources \
    --scaling-plan-name my-scaling-plan \
    --scaling-plan-version 1
```

输出：

```
{
    "ScalingPlanResources": [
        {
            "ScalableDimension": "autoscaling:autoScalingGroup:DesiredCapacity",
            "ScalingPlanVersion": 1,
            "ResourceId": "autoScalingGroup/my-asg",
            "ScalingStatusCode": "Active",
            "ScalingStatusMessage": "Target tracking scaling policies have been
 applied to the resource.",
            "ScalingPolicies": [
                {
                    "PolicyName": "AutoScaling-my-asg-b1ab65ae-4be3-4634-bd64-
c7471662b251",
                    "PolicyType": "TargetTrackingScaling",
                    "TargetTrackingConfiguration": {
                        "PredefinedScalingMetricSpecification": {
                            "PredefinedScalingMetricType":
 "ALBRequestCountPerTarget",
                            "ResourceLabel": "app/my-alb/f37c06a68c1748aa/
targetgroup/my-target-group/6d4ea56ca2d6a18d"
                        },
                        "TargetValue": 40.0
```

```
                    }
                }
            ],
            "ServiceNamespace": "autoscaling",
            "ScalingPlanName": "my-scaling-plan"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Auto Scaling 用户指南》中的什么是 Amazon Auto Scaling？。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingPlanResources。

describe-scaling-plans

以下代码示例演示了如何使用 describe-scaling-plans。

Amazon CLI

描述扩展计划

以下 describe-scaling-plans 示例显示指定扩展计划的详细信息。

```
aws autoscaling-plans describe-scaling-plans \
    --scaling-plan-names scaling-plan-with-asg-and-ddb
```

输出：

```
{
    "ScalingPlans": [
        {
            "LastMutatingRequestTime": 1565388443.963,
            "ScalingPlanVersion": 1,
            "CreationTime": 1565388443.963,
            "ScalingInstructions": [
                {
                    "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
                    "ScalableDimension":
 "autoscaling:autoScalingGroup:DesiredCapacity",
                    "TargetTrackingConfigurations": [
                        {
                            "PredefinedScalingMetricSpecification": {
```

```
                                "PredefinedScalingMetricType":
"ASGAverageCPUUtilization"
                            },
                            "TargetValue": 50.0,
                            "EstimatedInstanceWarmup": 300,
                            "DisableScaleIn": false
                        }
                    ],
                    "ResourceId": "autoScalingGroup/my-asg",
                    "DisableDynamicScaling": false,
                    "MinCapacity": 1,
                    "ServiceNamespace": "autoscaling",
                    "MaxCapacity": 10
                },
                {
                    "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
                    "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
                    "TargetTrackingConfigurations": [
                        {
                            "PredefinedScalingMetricSpecification": {
                                "PredefinedScalingMetricType":
"DynamoDBReadCapacityUtilization"
                            },
                            "TargetValue": 50.0,
                            "ScaleInCooldown": 60,
                            "DisableScaleIn": false,
                            "ScaleOutCooldown": 60
                        }
                    ],
                    "ResourceId": "table/my-table",
                    "DisableDynamicScaling": false,
                    "MinCapacity": 5,
                    "ServiceNamespace": "dynamodb",
                    "MaxCapacity": 10000
                },
                {
                    "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
                    "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
                    "TargetTrackingConfigurations": [
                        {
                            "PredefinedScalingMetricSpecification": {
                                "PredefinedScalingMetricType":
"DynamoDBWriteCapacityUtilization"
                            },
```

```
                            "TargetValue": 50.0,
                            "ScaleInCooldown": 60,
                            "DisableScaleIn": false,
                            "ScaleOutCooldown": 60
                        }
                    ],
                    "ResourceId": "table/my-table",
                    "DisableDynamicScaling": false,
                    "MinCapacity": 5,
                    "ServiceNamespace": "dynamodb",
                    "MaxCapacity": 10000
                }
            ],
            "ApplicationSource": {
                "TagFilters": [
                    {
                        "Values": [
                            "my-application-id"
                        ],
                        "Key": "application"
                    }
                ]
            },
            "StatusStartTime": 1565388455.836,
            "ScalingPlanName": "scaling-plan-with-asg-and-ddb",
            "StatusMessage": "Scaling plan has been created and applied to all
 resources.",
            "StatusCode": "Active"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Auto Scaling 用户指南》中的什么是 Amazon Auto Scaling？。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScalingPlans。

get-scaling-plan-resource-forecast-data

以下代码示例演示了如何使用 get-scaling-plan-resource-forecast-data。

Amazon CLI

检索负载预测数据

此示例检索与指定扩展计划关联的可扩展资源（自动扩缩组）的负载预测数据。

```
aws autoscaling-plans get-scaling-plan-resource-forecast-data \
    --scaling-plan-name my-scaling-plan \
    --scaling-plan-version 1 \
    --service-namespace "autoscaling" \
    --resource-id autoScalingGroup/my-asg \
    --scalable-dimension "autoscaling:autoScalingGroup:DesiredCapacity" \
    --forecast-data-type "LoadForecast" \
    --start-time "2019-08-30T00:00:00Z" \
    --end-time "2019-09-06T00:00:00Z"
```

输出：

```
{
    "Datapoints": [...]
}
```

有关更多信息，请参阅《Amazon Auto Scaling 用户指南》中的什么是 Amazon Auto Scaling。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetScalingPlanResourceForecastData。

update-scaling-plan

以下代码示例演示了如何使用 update-scaling-plan。

Amazon CLI

更新扩展计划

以下 update-scaling-plan 示例修改指定扩展计划中自动扩缩组的扩展指标。

```
aws autoscaling-plans update-scaling-plan \
    --scaling-plan-name my-scaling-plan \
    --scaling-plan-version 1 \
    --scaling-instructions
 '{"ScalableDimension":"autoscaling:autoScalingGroup:DesiredCapacity","ResourceId":"autoScal
my-asg","ServiceNamespace":"autoscaling","TargetTrackingConfigurations":
[{"PredefinedScalingMetricSpecification":
 {"PredefinedScalingMetricType":"ALBRequestCountPerTarget","ResourceLabel":"app/my-
```

```
alb/f37c06a68c1748aa/targetgroup/my-target-
group/6d4ea56ca2d6a18d"},"TargetValue":40.0}],"MinCapacity": 1,"MaxCapacity": 10}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Auto Scaling 用户指南》中的什么是 Amazon Auto Scaling？。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateScalingPlan。

Amazon Backup使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Backup 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-backup-plan`

以下代码示例演示了如何使用 `create-backup-plan`。

Amazon CLI

创建备份计划

以下 `create-backup-plan` 示例创建留存期为 35 天的指定备份计划。

```
aws backup create-backup-plan \
--backup-plan "{\"BackupPlanName\":\"Example-Backup-Plan\",\"Rules\":[{\"RuleName\":
\"DailyBackups\",\"ScheduleExpression\":\"cron(0 5 ? * * *)\",\"StartWindowMinutes
\":480,\"TargetBackupVaultName\":\"Default\",\"Lifecycle\":{\"DeleteAfterDays
\":35}}]}"
```

输出：

```
{
    "BackupPlanId": "1fa3895c-a7f5-484a-a371-2dd6a1a9f729",
    "BackupPlanArn": "arn:aws:backup:us-west-2:123456789012:backup-plan:1fa3895c-
a7f5-484a-a371-2dd6a1a9f729",
    "CreationDate": 1568928754.747,
    "VersionId": "ZjQ2ZTI5YWQtZDg5Yi00MzYzLWJmZTAtMDI1MzhlMDhjYjEz"
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBackupPlan。

create-backup-vault

以下代码示例演示了如何使用 create-backup-vault。

Amazon CLI

创建备份保管库

以下 create-backup-vault 示例创建具有指定名称的备份保管库。

```
aws backup create-backup-vault
    --backup-vault-name sample-vault
```

此命令不生成任何输出。输出：

```
{
    "BackupVaultName": "sample-vault",
    "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-vault:sample-
vault",
    "CreationDate": 1568928338.385
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份保管库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBackupVault。

get-backup-plan-from-template

以下代码示例演示了如何使用 get-backup-plan-from-template。

Amazon CLI

从模板中获取现有备份计划

以下 get-backup-plan-from-template 示例从模板中获取现有备份计划，指定了留存期为 35 天的每日备份。

```
aws backup get-backup-plan-from-template \
    --backup-plan-template-id "87c0c1ef-254d-4180-8fef-2e76a2c38aaa"
```

输出：

```
{
    "BackupPlanDocument": {
        "Rules": [
            {
                "RuleName": "DailyBackups",
                "ScheduleExpression": "cron(0 5 ? * * *)",
                "StartWindowMinutes": 480,
                "Lifecycle": {
                    "DeleteAfterDays": 35
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBackupPlanFromTemplate。

get-backup-plan

以下代码示例演示了如何使用 get-backup-plan。

Amazon CLI

获取备份计划的详细信息

以下 get-backup-plan 示例显示指定备份计划的详细信息。

```
aws backup get-backup-plan \
```

```
    --backup-plan-id "fcbf5d8f-bd77-4f3a-9c97-f24fb3d373a5"
```

输出：

```
{
    "BackupPlan": {
        "BackupPlanName": "Example-Backup-Plan",
        "Rules": [
            {
                "RuleName": "DailyBackups",
                "TargetBackupVaultName": "Default",
                "ScheduleExpression": "cron(0 5 ? * * *)",
                "StartWindowMinutes": 480,
                "CompletionWindowMinutes": 10080,
                "Lifecycle": {
                    "DeleteAfterDays": 35
                },
                "RuleId": "70e0ccdc-e9df-4e83-82ad-c1e5a9471cc3"
            }
        ]
    },
    "BackupPlanId": "fcbf5d8f-bd77-4f3a-9c97-f24fb3d373a5",
    "BackupPlanArn": "arn:aws:backup:us-west-2:123456789012:backup-plan:fcbf5d8f-
bd77-4f3a-9c97-f24fb3d373a5",
    "VersionId": "NjQ2ZTZkODktMGVhNy00MmQ0LWE4YjktZTkwNTQ3OTkyYTcw",
    "CreationDate": 1568926091.57
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBackupPlan。

list-backup-jobs

以下代码示例演示了如何使用 list-backup-jobs。

Amazon CLI

示例 1：列出所有备份任务

以下 list-backup-jobs 示例返回有关您 Amazon 账户中备份任务的元数据。

```
aws backup list-backup-jobs
```

输出：

```
{
    "BackupJobs": [
        {
            "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "BackupVaultName": "Default",
            "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-
vault:Default",
            "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-12345678901234567",
            "CreationDate": 1600721892.929,
            "State": "CREATED",
            "PercentDone": "0.0",
            "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/
AWSBackupDefaultServiceRole",
            "StartBy": 1600725492.929,
            "ResourceType": "EC2"
        },
        {
            "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "BackupVaultName": "Default",
            "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-
vault:Default",
            "RecoveryPointArn": "arn:aws:backup:us-west-2:123456789012:recovery-
point:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "ResourceArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-
system/fs-12345678",
            "CreationDate": 1600721724.77,
            "CompletionDate": 1600721744.488,
            "State": "COMPLETED",
            "PercentDone": "100.0",
            "BackupSizeInBytes": 71,
            "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/
AWSBackupDefaultServiceRole",
            "StartBy": 1600725324.77,
            "ResourceType": "EFS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份。

示例 2：列出已完成的备份任务

以下 `list-backup-jobs` 示例返回有关您 Amazon 账户中已完成的备份任务的元数据。

```
aws backup list-backup-jobs \
    --by-state COMPLETED
```

输出：

```
{
    "BackupJobs": [
        {
            "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "BackupVaultName": "Default",
            "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-
vault:Default",
            "RecoveryPointArn": "arn:aws:backup:us-west-2:123456789012:recovery-
point:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "ResourceArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-
system/fs-12345678",
            "CreationDate": 1600721724.77,
            "CompletionDate": 1600721744.488,
            "State": "COMPLETED",
            "PercentDone": "100.0",
            "BackupSizeInBytes": 71,
            "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/
AWSBackupDefaultServiceRole",
            "StartBy": 1600725324.77,
            "ResourceType": "EFS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Backup 开发人员指南》中的创建备份。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBackupJobs。

Amazon Batch使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Batch 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`cancel-job`

以下代码示例演示了如何使用 `cancel-job`。

Amazon CLI

取消作业

此示例取消具有指定作业 ID 的作业。

命令:

```
aws batch cancel-job --job-id bcf0b186-a532-4122-842e-2ccab8d54efb --
reason "Cancelling job."
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelJob。

`create-compute-environment`

以下代码示例演示了如何使用 `create-compute-environment`。

Amazon CLI

创建具有按需型实例的托管计算环境

此示例创建一个托管计算环境，具有按需启动的特定 C4 实例类型。计算环境称为 C4OnDemand。

命令:

```
aws batch create-compute-environment --cli-input-json file://<path_to_json_file>/
C4OnDemand.json
```

JSON 文件格式：

```
{
  "computeEnvironmentName": "C4OnDemand",
  "type": "MANAGED",
  "state": "ENABLED",
  "computeResources": {
    "type": "EC2",
    "minvCpus": 0,
    "maxvCpus": 128,
    "desiredvCpus": 48,
    "instanceTypes": [
      "c4.large",
      "c4.xlarge",
      "c4.2xlarge",
      "c4.4xlarge",
      "c4.8xlarge"
    ],
    "subnets": [
      "subnet-220c0e0a",
      "subnet-1a95556d",
      "subnet-978f6dce"
    ],
    "securityGroupIds": [
      "sg-cf5093b2"
    ],
    "ec2KeyPair": "id_rsa",
    "instanceRole": "ecsInstanceRole",
    "tags": {
      "Name": "Batch Instance - C4OnDemand"
    }
  },
  "serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole"
}
```

输出：

```
{
    "computeEnvironmentName": "C4OnDemand",
```

```
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/C4OnDemand"
}
```

创建具有竞价型实例托管计算环境

此示例创建一个具有 M4 实例类型的托管计算环境，在竞价出价等于或低于该实例类型按需价格的 20% 时启动。计算环境称为 M4Spot。

命令:

```
aws batch create-compute-environment --cli-input-json file://<path_to_json_file>/
M4Spot.json
```

JSON 文件格式：

```
{
  "computeEnvironmentName": "M4Spot",
  "type": "MANAGED",
  "state": "ENABLED",
  "computeResources": {
    "type": "SPOT",
    "spotIamFleetRole": "arn:aws:iam::012345678910:role/aws-ec2-spot-fleet-role",
    "minvCpus": 0,
    "maxvCpus": 128,
    "desiredvCpus": 4,
    "instanceTypes": [
      "m4"
    ],
    "bidPercentage": 20,
    "subnets": [
      "subnet-220c0e0a",
      "subnet-1a95556d",
      "subnet-978f6dce"
    ],
    "securityGroupIds": [
      "sg-cf5093b2"
    ],
    "ec2KeyPair": "id_rsa",
    "instanceRole": "ecsInstanceRole",
    "tags": {
      "Name": "Batch Instance - M4Spot"
    }
```

```
  },
  "serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole"
}
```

输出：

```
{
    "computeEnvironmentName": "M4Spot",
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/M4Spot"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateComputeEnvironment。

`create-job-queue`

以下代码示例演示了如何使用 `create-job-queue`。

Amazon CLI

创建包含单一计算环境的低优先级作业队列

此示例创建一个名为 LowPriority 且使用 M4Spot 计算环境的作业队列。

命令:

```
aws batch create-job-queue --cli-input-json file://<path_to_json_file>/
LowPriority.json
```

JSON 文件格式：

```
{
  "jobQueueName": "LowPriority",
  "state": "ENABLED",
  "priority": 10,
  "computeEnvironmentOrder": [
    {
      "order": 1,
      "computeEnvironment": "M4Spot"
    }
  ]
```

```
}
```

输出：

```
{
    "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/LowPriority",
    "jobQueueName": "LowPriority"
}
```

创建包含两个计算环境的高优先级作业队列

此示例创建一个名为 HighPriority 的作业队列，该队列使用顺序为 1 的 C4OnDemand 计算环境和顺序为 2 的 M4Spot 计算环境。调度器将首先尝试在 C4OnDemand 计算环境中放置作业。

命令:

```
aws batch create-job-queue --cli-input-json file://<path_to_json_file>/
HighPriority.json
```

JSON 文件格式：

```
{
  "jobQueueName": "HighPriority",
  "state": "ENABLED",
  "priority": 1,
  "computeEnvironmentOrder": [
    {
      "order": 1,
      "computeEnvironment": "C4OnDemand"
    },
    {
      "order": 2,
      "computeEnvironment": "M4Spot"
    }
  ]
}
```

输出：

```
{
    "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/HighPriority",
```

```
    "jobQueueName": "HighPriority"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJobQueue。

delete-compute-environment

以下代码示例演示了如何使用 delete-compute-environment。

Amazon CLI

删除计算环境

此示例删除 P2OnDemand 计算环境。

命令:

```
aws batch delete-compute-environment --compute-environment P2OnDemand
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteComputeEnvironment。

delete-job-queue

以下代码示例演示了如何使用 delete-job-queue。

Amazon CLI

删除作业队列

此示例删除 GPGPU 作业队列。

命令:

```
aws batch delete-job-queue --job-queue GPGPU
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteJobQueue。

deregister-job-definition

以下代码示例演示了如何使用 deregister-job-definition。

Amazon CLI

注销作业定义

此示例注销名为 sleep10 的作业定义。

命令:

```
aws batch deregister-job-definition --job-definition sleep10
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterJobDefinition。

describe-compute-environments

以下代码示例演示了如何使用 describe-compute-environments。

Amazon CLI

描述计算环境

此示例描述 P2OnDemand 计算环境。

命令:

```
aws batch describe-compute-environments --compute-environments P2OnDemand
```

输出：

```
{
    "computeEnvironments": [
        {
            "status": "VALID",
            "serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole",
            "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/P2OnDemand",
            "computeResources": {
                "subnets": [
                    "subnet-220c0e0a",
                    "subnet-1a95556d",
                    "subnet-978f6dce"
```

```
                ],
                "tags": {
                    "Name": "Batch Instance - P2OnDemand"
                },
                "desiredvCpus": 48,
                "minvCpus": 0,
                "instanceTypes": [
                    "p2"
                ],
                "securityGroupIds": [
                    "sg-cf5093b2"
                ],
                "instanceRole": "ecsInstanceRole",
                "maxvCpus": 128,
                "type": "EC2",
                "ec2KeyPair": "id_rsa"
            },
            "statusReason": "ComputeEnvironment Healthy",
            "ecsClusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/
P2OnDemand_Batch_2c06f29d-d1fe-3a49-879d-42394c86effc",
            "state": "ENABLED",
            "computeEnvironmentName": "P2OnDemand",
            "type": "MANAGED"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeComputeEnvironments。

describe-job-definitions

以下代码示例演示了如何使用 describe-job-definitions。

Amazon CLI

描述活动作业定义

此示例描述您的所有活动作业定义。

命令:

```
aws batch describe-job-definitions --status ACTIVE
```

输出：

```
{
    "jobDefinitions": [
        {
            "status": "ACTIVE",
            "jobDefinitionArn": "arn:aws:batch:us-east-1:012345678910:job-
definition/sleep60:1",
            "containerProperties": {
                "mountPoints": [],
                "parameters": {},
                "image": "busybox",
                "environment": {},
                "vcpus": 1,
                "command": [
                    "sleep",
                    "60"
                ],
                "volumes": [],
                "memory": 128,
                "ulimits": []
            },
            "type": "container",
            "jobDefinitionName": "sleep60",
            "revision": 1
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJobDefinitions。

`describe-job-queues`

以下代码示例演示了如何使用 `describe-job-queues`。

Amazon CLI

描述作业队列

此示例描述 HighPriority 作业队列。

命令:

```
aws batch describe-job-queues --job-queues HighPriority
```

输出：

```
{
    "jobQueues": [
        {
            "status": "VALID",
            "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/
HighPriority",
            "computeEnvironmentOrder": [
                {
                    "computeEnvironment": "arn:aws:batch:us-
east-1:012345678910:compute-environment/C4OnDemand",
                    "order": 1
                }
            ],
            "statusReason": "JobQueue Healthy",
            "priority": 1,
            "state": "ENABLED",
            "jobQueueName": "HighPriority"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJobQueues。

describe-jobs

以下代码示例演示了如何使用 describe-jobs。

Amazon CLI

描述任务

以下 describe-jobs 示例描述具有指定作业 ID 的作业。

```
aws batch describe-jobs \
    --jobs bcf0b186-a532-4122-842e-2ccab8d54efb
```

输出：

```
{
    "jobs": [
        {
            "status": "SUBMITTED",
            "container": {
                "mountPoints": [],
                "image": "busybox",
                "environment": [],
                "vcpus": 1,
                "command": [
                    "sleep",
                    "60"
                ],
                "volumes": [],
                "memory": 128,
                "ulimits": []
            },
            "parameters": {},
            "jobDefinition": "arn:aws:batch:us-east-1:012345678910:job-definition/
sleep60:1",
            "jobQueue": "arn:aws:batch:us-east-1:012345678910:job-queue/
HighPriority",
            "jobId": "bcf0b186-a532-4122-842e-2ccab8d54efb",
            "dependsOn": [],
            "jobName": "example",
            "createdAt": 1480483387803
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJobs。

`list-jobs`

以下代码示例演示了如何使用 `list-jobs`。

Amazon CLI

列出正在运行的作业

此示例列出 HighPriority 作业队列中正在进行的作业。

命令:

```
aws batch list-jobs --job-queue HighPriority
```

输出：

```
{
    "jobSummaryList": [
        {
            "jobName": "example",
            "jobId": "e66ff5fd-a1ff-4640-b1a2-0b0a142f49bb"
        }
    ]
}
```

列出已提交的作业

此示例列出 HighPriority 作业队列中处于 SUBMITTED 作业状态的作业。

命令:

```
aws batch list-jobs --job-queue HighPriority --job-status SUBMITTED
```

输出：

```
{
    "jobSummaryList": [
        {
            "jobName": "example",
            "jobId": "68f0c163-fbd4-44e6-9fd1-25b14a434786"
        }
    ]
}
```

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

register-job-definition

以下代码示例演示了如何使用 register-job-definition。

Amazon CLI

注册作业定义

此示例为一个简单容器作业注册作业定义。

命令:

```
aws batch register-job-definition --job-definition-name sleep30 --type container --
container-properties '{ "image": "busybox", "vcpus": 1, "memory": 128, "command":
 [ "sleep", "30"]}'
```

输出：

```
{
    "jobDefinitionArn": "arn:aws:batch:us-east-1:012345678910:job-definition/
sleep30:1",
    "jobDefinitionName": "sleep30",
    "revision": 1
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterJobDefinition。

submit-job

以下代码示例演示了如何使用 submit-job。

Amazon CLI

提交作业

此示例向 HighPriority 作业队列提交一个名为 example 的简单容器作业。

命令:

```
aws batch submit-job --job-name example --job-queue HighPriority  --job-
definition sleep60
```

输出：

```
{
    "jobName": "example",
    "jobId": "876da822-4198-45f2-a252-6cea32512ea8"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SubmitJob。

terminate-job

以下代码示例演示了如何使用 terminate-job。

Amazon CLI

终止作业

此示例中止具有指定作业 ID 的作业。

命令:

```
aws batch terminate-job --job-id 61e743ed-35e4-48da-b2de-5c8333821c84 --
reason "Terminating job."
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateJob。

update-compute-environment

以下代码示例演示了如何使用 update-compute-environment。

Amazon CLI

更新计算环境

此示例禁用 P2OnDemand 计算环境，以便将其删除。

命令:

```
aws batch update-compute-environment --compute-environment P2OnDemand --
state DISABLED
```

输出：

```
{
    "computeEnvironmentName": "P2OnDemand",
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/P2OnDemand"
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateComputeEnvironment。

update-job-queue

以下代码示例演示了如何使用 update-job-queue。

Amazon CLI

更新作业队列

此示例禁用作业队列，以便将其删除。

命令:

```
aws batch update-job-queue --job-queue GPGPU --state DISABLED
```

输出：

```
{
    "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/GPGPU",
    "jobQueueName": "GPGPU"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJobQueue。

Amazon Budgets使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Budgets 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

create-budget

以下代码示例演示了如何使用 create-budget。

Amazon CLI

创建成本和使用量预算

以下 create-budget 命令创建成本和使用量预算。

```
aws budgets create-budget \
    --account-id 111122223333 \
    --budget file://budget.json \
    --notifications-with-subscribers file://notifications-with-subscribers.json
```

的内容budget.json：

```
{
    "BudgetLimit": {
        "Amount": "100",
        "Unit": "USD"
    },
    "BudgetName": "Example Tag Budget",
    "BudgetType": "COST",
    "CostFilters": {
        "TagKeyValue": [
            "user:Key$value1",
            "user:Key$value2"
        ]
    },
    "CostTypes": {
        "IncludeCredit": true,
        "IncludeDiscount": true,
        "IncludeOtherSubscription": true,
        "IncludeRecurring": true,
        "IncludeRefund": true,
        "IncludeSubscription": true,
        "IncludeSupport": true,
```

```
        "IncludeTax": true,
        "IncludeUpfront": true,
        "UseBlended": false
    },
    "TimePeriod": {
        "Start": 1477958399,
        "End": 3706473600
    },
    "TimeUnit": "MONTHLY"
}
```

的内容notifications-with-subscribers.json：

```
[
    {
        "Notification": {
            "ComparisonOperator": "GREATER_THAN",
            "NotificationType": "ACTUAL",
            "Threshold": 80,
            "ThresholdType": "PERCENTAGE"
        },
        "Subscribers": [
            {
                "Address": "example@example.com",
                "SubscriptionType": "EMAIL"
            }
        ]
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBudget。

create-notification

以下代码示例演示了如何使用 create-notification。

Amazon CLI

为指定的成本和使用量预算创建通知

此示例为指定的成本和使用量预算创建通知。

命令:

```
aws budgets create-notification --account-id 111122223333 --budget-name "Example
 Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdT
 --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNotification。

create-subscriber

以下代码示例演示了如何使用 create-subscriber。

Amazon CLI

为与成本和使用量预算关联的通知创建订阅用户

此示例为指定通知创建订阅用户。

命令:

```
aws budgets create-subscriber --account-id 111122223333 --budget-name "Example
 Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdT
 --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscriber。

delete-budget

以下代码示例演示了如何使用 delete-budget。

Amazon CLI

删除成本和使用量预算

此示例删除指定的成本和使用量预算。

命令:

```
aws budgets delete-budget --account-id 111122223333 --budget-name "Example Budget"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBudget。

delete-notification

以下代码示例演示了如何使用 delete-notification。

Amazon CLI

从预算中删除通知

此示例删除指定预算中的指定通知。

命令:

```
aws budgets delete-notification --account-id 111122223333 --budget-name "Example
 Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdT
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNotification。

delete-subscriber

以下代码示例演示了如何使用 delete-subscriber。

Amazon CLI

从通知中删除订阅用户

此示例删除指定通知中的指定订阅用户。

命令:

```
aws budgets delete-subscriber --account-id 111122223333 --budget-name "Example
 Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdT
 --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubscriber。

describe-budget

以下代码示例演示了如何使用 describe-budget。

Amazon CLI

检索与账户关联的预算

此示例检索指定的成本和使用量预算。

命令:

```
aws budgets describe-budget --account-id 111122223333 --budget-name "Example Budget"
```

输出：

```
{
   "Budget": {
       "CalculatedSpend": {
           "ForecastedSpend": {
               "Amount": "2641.548000000000229192664846777915954589843 75",
               "Unit": "USD"
           },
           "ActualSpend": {
               "Amount": "604.456000000000017280399333685636520385742 1875",
               "Unit": "USD"
           }
       },
       "BudgetType": "COST",
       "BudgetLimit": {
           "Amount": "100",
           "Unit": "USD"
       },
       "BudgetName": "Example Budget",
       "CostTypes": {
           "IncludeOtherSubscription": true,
           "IncludeUpfront": true,
           "IncludeRefund": true,
           "UseBlended": false,
           "IncludeDiscount": true,
           "UseAmortized": false,
           "IncludeTax": true,
           "IncludeCredit": true,
           "IncludeSupport": true,
           "IncludeRecurring": true,
           "IncludeSubscription": true
       },
```

```
        "TimeUnit": "MONTHLY",
        "TimePeriod": {
            "Start": 1477958399.0,
            "End": 3706473600.0
        },
        "CostFilters": {
            "AZ": [
                "us-east-1"
            ]
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBudget。

describe-budgets

以下代码示例演示了如何使用 describe-budgets。

Amazon CLI

检索与账户关联的预算

此示例检索账户的成本和使用量预算。

命令:

```
aws budgets describe-budgets --account-id 111122223333 --max-results 20
```

输出：

```
{
    "Budgets": [
        {
            "CalculatedSpend": {
                "ForecastedSpend": {
                    "Amount": "2641.548000000000229192664846777915954589843 75",
                    "Unit": "USD"
                },
                "ActualSpend": {
                    "Amount": "604.456000000000017280399333685636520385742 1875",
                    "Unit": "USD"
                }
```

```
            },
            "BudgetType": "COST",
            "BudgetLimit": {
                "Amount": "100",
                "Unit": "USD"
            },
            "BudgetName": "Example Budget",
            "CostTypes": {
                "IncludeOtherSubscription": true,
                "IncludeUpfront": true,
                "IncludeRefund": true,
                "UseBlended": false,
                "IncludeDiscount": true,
                "UseAmortized": false,
                "IncludeTax": true,
                "IncludeCredit": true,
                "IncludeSupport": true,
                "IncludeRecurring": true,
                "IncludeSubscription": true
            },
            "TimeUnit": "MONTHLY",
            "TimePeriod": {
                "Start": 1477958399.0,
                "End": 3706473600.0
            },
            "CostFilters": {
                "AZ": [
                    "us-east-1"
                ]
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBudgets。

describe-notifications-for-budget

以下代码示例演示了如何使用 describe-notifications-for-budget。

Amazon CLI

检索预算的通知

此示例检索成本和使用量预算的通知。

命令:

```
aws budgets describe-notifications-for-budget --account-id 111122223333 --budget-
name "Example Budget" --max-results 5
```

输出：

```
{
   "Notifications": [
       {
           "Threshold": 80.0,
           "ComparisonOperator": "GREATER_THAN",
           "NotificationType": "ACTUAL"
       }
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNotificationsForBudget。

describe-subscribers-for-notification

以下代码示例演示了如何使用 describe-subscribers-for-notification。

Amazon CLI

检索预算通知的订阅用户

此示例检索成本和使用量预算通知的订阅用户。

命令:

```
aws budgets describe-subscribers-for-notification --
account-id 111122223333 --budget-name "Example Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,Threshold1
 --max-results 5
```

输出：

```
{
```

```
    "Subscribers": [
        {
            "SubscriptionType": "EMAIL",
            "Address": "example2@example.com"
        },
        {
            "SubscriptionType": "EMAIL",
            "Address": "example@example.com"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSubscribersForNotification。

update-budget

以下代码示例演示了如何使用 update-budget。

Amazon CLI

替换成本和使用量预算的预算

此示例将成本和使用量预算替换为新预算。

命令:

```
aws budgets update-budget --account-id 111122223333 --new-budget file://new-budget.json
```

new-budget.json：

```
{
    "BudgetLimit": {
       "Amount": "100",
       "Unit": "USD"
    },
    "BudgetName": "Example Budget",
    "BudgetType": "COST",
    "CostFilters": {
       "AZ" : [ "us-east-1" ]
```

```
    },
    "CostTypes": {
       "IncludeCredit": false,
       "IncludeDiscount": true,
       "IncludeOtherSubscription": true,
       "IncludeRecurring": true,
       "IncludeRefund": true,
       "IncludeSubscription": true,
       "IncludeSupport": true,
       "IncludeTax": true,
       "IncludeUpfront": true,
       "UseBlended": false,
       "UseAmortized": true
    },
    "TimePeriod": {
       "Start": 1477958399,
       "End": 3706473600
    },
    "TimeUnit": "MONTHLY"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBudget。

update-notification

以下代码示例演示了如何使用 update-notification。

Amazon CLI

替换成本和使用量预算的通知

此示例将成本和使用量预算的 80% 通知替换为 90% 通知。

命令:

```
aws budgets update-notification --account-id 111122223333 --budget-name "Example
 Budget" --old-notification
  NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENT
 --new-notification
  NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=90,ThresholdType=PERCENT
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateNotification。

update-subscriber

以下代码示例演示了如何使用 update-subscriber。

Amazon CLI

替换成本和使用量预算的订阅用户

此示例替换成本和使用量预算的订阅用户。

命令:

```
aws budgets update-subscriber --account-id 111122223333 --budget-name "Example
 Budget" --
notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdT
 --old-subscriber SubscriptionType=EMAIL,Address=example@example.com --new-
subscriber SubscriptionType=EMAIL,Address=example2@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscriber。

使用 Amazon CLI 的 Amazon Chime 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Chime 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

associate-phone-number-with-user

以下代码示例演示了如何使用 associate-phone-number-with-user。

Amazon CLI

将电话号码与用户关联

以下 associate-phone-number-with-user 示例将指定电话号码与用户关联。

```
aws chime associate-phone-number-with-user \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --user-id 1ab2345c-67de-8901-f23g-45h678901j2k \
    --e164-phone-number "+12065550100"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociatePhoneNumberWithUser。

associate-signin-delegate-groups-with-account

以下代码示例演示了如何使用 associate-signin-delegate-groups-with-account。

Amazon CLI

关联登录委托组

以下 associate-signin-delegate-groups-with-account 示例将指定的登录委托组与指定的 Amazon Chime 账户关联。

```
aws chime associate-signin-delegate-groups-with-account \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --signin-delegate-groups GroupName=my_users
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户访问和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateSigninDelegateGroupsWithAccount。

batch-create-room-membership

以下代码示例演示了如何使用 batch-create-room-membership。

Amazon CLI

创建多个聊天室成员资格

以下 batch-create-room-membership 示例将多个用户作为聊天室成员添加到聊天室。它还会为用户分配管理员和成员角色。

```
aws chime batch-create-room-membership \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \
    --membership-item-list "MemberId=1ab2345c-67de-8901-
f23g-45h678901j2k,Role=Administrator" "MemberId=2ab2345c-67de-8901-
f23g-45h678901j2k,Role=Member"
```

输出：

```
{
    "ResponseMetadata": {
        "RequestId": "169ba401-d886-475f-8b3f-e01eac6fadfb",
        "HTTPStatusCode": 201,
        "HTTPHeaders": {
            "x-amzn-requestid": "169ba401-d886-475f-8b3f-e01eac6fadfb",
            "content-type": "application/json",
            "content-length": "13",
            "date": "Mon, 02 Dec 2019 22:46:58 GMT",
            "connection": "keep-alive"
        },
        "RetryAttempts": 0
    },
    "Errors": []
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchCreateRoomMembership。

batch-delete-phone-number

以下代码示例演示了如何使用 batch-delete-phone-number。

Amazon CLI

删除多个电话号码

以下 batch-delete-phone-number 示例删除所有指定的电话号码。

```
aws chime batch-delete-phone-number \
    --phone-number-ids "%2B12065550100" "%2B12065550101"
```

此命令不生成任何输出。输出：

```
{
    "PhoneNumberErrors": []
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeletePhoneNumber。

batch-suspend-user

以下代码示例演示了如何使用 batch-suspend-user。

Amazon CLI

暂停多个用户

以下 batch-suspend-user 示例暂停指定 Amazon Chime 账户中列出的用户。

```
aws chime batch-suspend-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-id-list "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE" "a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE" "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"
```

输出：

```
{
    "UserErrors": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchSuspendUser。

batch-unsuspend-user

以下代码示例演示了如何使用 batch-unsuspend-user。

Amazon CLI

取消暂停多个用户

以下 batch-unsuspend-user 示例移除指定 Amazon Chime 账户中所列用户之前的任何暂停。

```
aws chime batch-unsuspend-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-id-list "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE" "a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE" "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"
```

输出：

```
{
    "UserErrors": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUnsuspendUser。

batch-update-phone-number

以下代码示例演示了如何使用 batch-update-phone-number。

Amazon CLI

同时更新多个电话号码产品类型

以下 batch-update-phone-number 示例更新所有指定电话号码的产品类型。

```
aws chime batch-update-phone-number \
    --update-phone-number-request-items PhoneNumberId=
%2B12065550100,ProductType=BusinessCalling PhoneNumberId=
%2B12065550101,ProductType=BusinessCalling
```

输出：

```
{
    "PhoneNumberErrors": []
}
```

同时更新多个电话号码呼叫名称

以下 batch-update-phone-number 示例更新所有指定电话号码的呼叫名称。

```
aws chime batch-update-phone-number \
    --update-phone-number-request-items PhoneNumberId=
%2B14013143874,CallingName=phonenumber1 PhoneNumberId=
%2B14013144061,CallingName=phonenumber2
```

输出：

```
{
    "PhoneNumberErrors": []
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdatePhoneNumber。

batch-update-user

以下代码示例演示了如何使用 batch-update-user。

Amazon CLI

使用单个命令更新多个用户

以下 batch-update-user 示例更新指定 Amazon Chime 账户中列出的每位用户的 LicenseType。

```
aws chime batch-update-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
    --update-user-request-items "UserId=a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE,LicenseType=Basic" "UserId=a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE,LicenseType=Basic"
```

输出：

```
{
    "UserErrors": []
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateUser。

create-account

以下代码示例演示了如何使用 create-account。

Amazon CLI

创建账户

以下 create-account 示例在管理员的 Amazon 账户中创建 Amazon Chime 账户。

```
aws chime create-account \
    --name MyChimeAccount
```

输出：

```
{
    "Account": {
        "AwsAccountId": "111122223333",
        "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "Name": "MyChimeAccount",
        "AccountType": "Team",
        "CreatedTimestamp": "2019-01-04T17:11:22.003Z",
        "DefaultLicense": "Pro",
        "SupportedLicenses": [
            "Basic",
            "Pro"
        ],
        "SigninDelegateGroups": [
            {
                "GroupName": "myGroup"
            },
        ]
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccount。

create-bot

以下代码示例演示了如何使用 create-bot。

Amazon CLI

创建 Amazon Chime 机器人

以下 create-bot 示例为指定的 Amazon Chime Enterprise 账户创建一个机器人。

```
aws chime create-bot \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --display-name "myBot" \
    --domain "example.com"
```

输出：

```
{
    "Bot": {
        "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "DisplayName": "myBot (Bot)",
        "BotType": "ChatBot",
        "Disabled": false,
        "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
        "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
        "BotEmail": "myBot-chimebot@example.com",
        "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
    }
}
```

有关更多信息，请参阅《Amazon Chime 开发人员指南》中的将聊天机器人与 Amazon Chime 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBot。

create-phone-number-order

以下代码示例演示了如何使用 create-phone-number-order。

Amazon CLI

创建电话号码订单

以下 create-phone-number-order 示例为指定的电话号码创建电话号码订单。

```
aws chime create-phone-number-order \
    --product-type VoiceConnector \
    --e164-phone-numbers "+12065550100" "+12065550101" "+12065550102"
```

输出：

```
{
    "PhoneNumberOrder": {
        "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
        "ProductType": "VoiceConnector",
        "Status": "Processing",
        "OrderedPhoneNumbers": [
            {
                "E164PhoneNumber": "+12065550100",
                "Status": "Processing"
            },
            {
               "E164PhoneNumber": "+12065550101",
               "Status": "Processing"
            },
            {
              "E164PhoneNumber": "+12065550102",
              "Status": "Processing"
            }
        ],
        "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
        "UpdatedTimestamp": "2019-08-09T21:35:22.408Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePhoneNumberOrder。

create-room-membership

以下代码示例演示了如何使用 create-room-membership。

Amazon CLI

创建聊天室成员资格

以下 create-room-membership 示例将指定用户作为聊天室成员添加到聊天室。

```
aws chime create-room-membership \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \
    --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

输出：

```
{
    "RoomMembership": {
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Member": {
            "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",
            "MemberType": "User",
            "Email": "janed@example.com",
            "FullName": "Jane Doe",
            "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"
        },
        "Role": "Member",
        "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",
        "UpdatedTimestamp": "2019-12-02T22:36:41.969Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoomMembership。

create-room

以下代码示例演示了如何使用 create-room。

Amazon CLI

创建聊天室

以下 create-room 示例为指定的 Amazon Chime 账户创建一个聊天室。

```
aws chime create-room \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --name chatRoom
```

输出：

```
{
    "Room": {
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Name": "chatRoom",
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
        "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
        "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
        "UpdatedTimestamp": "2019-12-02T22:29:31.549Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoom。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

为共享设备创建用户配置文件

以下 create-user 示例为指定的电子邮件地址创建共享设备配置文件。

```
aws chime create-user \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --email roomdevice@example.com \
    --user-type SharedDevice
```

输出：

```
{
    "User": {
        "UserId": "1ab2345c-67de-8901-f23g-45h678901j2k",
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
        "PrimaryEmail": "roomdevice@example.com",
        "DisplayName": "Room Device",
        "LicenseType": "Pro",
        "UserType": "SharedDevice",
```

```
        "UserRegistrationStatus": "Registered",
        "RegisteredOn": "2020-01-15T22:38:09.806Z",
        "AlexaForBusinessMetadata": {
            "IsAlexaForBusinessEnabled": false
        }
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的设置准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

delete-account

以下代码示例演示了如何使用 delete-account。

Amazon CLI

删除账户

以下 delete-account 示例删除指定账户。

```
aws chime delete-account --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的删除您的账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccount。

delete-phone-number

以下代码示例演示了如何使用 delete-phone-number。

Amazon CLI

删除电话号码

以下 delete-phone-number 示例将指定的电话号码移入删除队列。

```
aws chime delete-phone-number \
```

```
    --phone-number-id "+12065550100"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePhoneNumber。

delete-room-membership

以下代码示例演示了如何使用 delete-room-membership。

Amazon CLI

移除聊天室用户的成员资格

以下 delete-room-membership 示例从指定聊天室中移除指定成员。

```
aws chime delete-room-membership \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \
    --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoomMembership。

delete-room

以下代码示例演示了如何使用 delete-room。

Amazon CLI

删除聊天室

以下 delete-room 示例删除指定聊天室，并删除聊天室成员资格。

```
aws chime delete-room \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoom。

disassociate-phone-number-from-user

以下代码示例演示了如何使用 disassociate-phone-number-from-user。

Amazon CLI

取消电话号码与用户的关联

以下 disassociate-phone-number-from-user 示例取消电话号码与指定用户的关联。

```
aws chime disassociate-phone-number-from-user \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --user-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociatePhoneNumberFromUser。

disassociate-signin-delegate-groups-from-account

以下代码示例演示了如何使用 disassociate-signin-delegate-groups-from-account。

Amazon CLI

取消关联登录委托组

以下 disassociate-signin-delegate-groups-from-account 示例取消指定登录委托组与指定 Amazon Chime 账户的关联。

```
aws chime disassociate-signin-delegate-groups-from-account \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --group-names "my_users"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户访问和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateSigninDelegateGroupsFromAccount。

get-account-settings

以下代码示例演示了如何使用 get-account-settings。

Amazon CLI

检索账户的设置

以下 get-account-settings 示例检索指定账户的账户设置。

```
aws chime get-account-settings --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "AccountSettings": {
        "DisableRemoteControl": false,
        "EnableDialOut": false
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理您的 Amazon Chime 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccountSettings。

get-account

以下代码示例演示了如何使用 get-account。

Amazon CLI

检索账户的详细信息

以下 get-account 示例检索指定 Amazon Chime 账户的详细信息。

```
aws chime get-account \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "Account": {
        "AwsAccountId": "111122223333",
        "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "Name": "EnterpriseDirectory",
        "AccountType": "EnterpriseDirectory",
        "CreatedTimestamp": "2018-12-20T18:38:02.181Z",
        "DefaultLicense": "Pro",
        "SupportedLicenses": [
            "Basic",
            "Pro"
        ],
        "SigninDelegateGroups": [
            {
                "GroupName": "myGroup"
            },
        ]
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理您的 Amazon Chime 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccount。

get-bot

以下代码示例演示了如何使用 get-bot。

Amazon CLI

检索有关机器人的详细信息

以下 get-bot 示例显示指定机器人的详细信息。

```
aws chime get-bot \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k
```

输出：

```
{
    "Bot": {
        "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "DisplayName": "myBot (Bot)",
        "BotType": "ChatBot",
        "Disabled": false,
        "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
        "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
        "BotEmail": "myBot-chimebot@example.com",
        "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
    }
}
```

有关更多信息，请参阅《Amazon Chime 开发人员指南》中的更新聊天机器人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBot。

get-global-settings

以下代码示例演示了如何使用 get-global-settings。

Amazon CLI

获取全局设置

以下 get-global-settings 示例检索用于存储与管理员 Amazon 账户关联的 Amazon Chime Business Calling 和 Amazon Chime Voice Connector 的调用详细记录的 S3 存储桶名称。

```
aws chime get-global-settings
```

输出：

```
{
    "BusinessCalling": {
        "CdrBucket": "s3bucket"
    },
    "VoiceConnector": {
        "CdrBucket": "s3bucket"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理全局设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGlobalSettings。

get-phone-number-order

以下代码示例演示了如何使用 get-phone-number-order。

Amazon CLI

获取电话号码订单的详细信息

以下 get-phone-number-order 示例显示指定电话号码订单的详细信息。

```
aws chime get-phone-number-order \
    --phone-number-order-id abc12345-de67-89f0-123g-h45i678j9012
```

输出：

```
{
    "PhoneNumberOrder": {
        "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
        "ProductType": "VoiceConnector",
        "Status": "Partial",
        "OrderedPhoneNumbers": [
            {
              "E164PhoneNumber": "+12065550100",
              "Status": "Acquired"
            },
            {
                "E164PhoneNumber": "+12065550101",
                "Status": "Acquired"
            },
            {
                "E164PhoneNumber": "+12065550102",
                "Status": "Failed"
            }
        ],
        "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
        "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPhoneNumberOrder。

get-phone-number-settings

以下代码示例演示了如何使用 get-phone-number-settings。

Amazon CLI

检索出站呼叫名称

以下 get-phone-number-settings 示例检索呼叫用户 Amazon 账户的默认出站呼叫名称。

```
aws chime get-phone-number-settings
```

此命令不生成任何输出。输出：

```
{
    "CallingName": "myName",
    "CallingNameUpdatedTimestamp": "2019-10-28T18:56:42.911Z"
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPhoneNumberSettings。

get-phone-number

以下代码示例演示了如何使用 get-phone-number。

Amazon CLI

获取电话号码的详细信息

以下 get-phone-number 示例显示指定电话号码的详细信息。

```
aws chime get-phone-number \
```

```
    --phone-number-id +12065550100
```

输出：

```
{
    "PhoneNumber": {
        "PhoneNumberId": "%2B12065550100",
        "E164PhoneNumber": "+12065550100",
        "Type": "Local",
        "ProductType": "VoiceConnector",
        "Status": "Unassigned",
        "Capabilities": {
            "InboundCall": true,
            "OutboundCall": true,
            "InboundSMS": true,
            "OutboundSMS": true,
            "InboundMMS": true,
            "OutboundMMS": true
        },
       "Associations": [
            {
                "Value": "abcdef1ghij2klmno3pqr4",
                "Name": "VoiceConnectorId",
                "AssociatedTimestamp": "2019-10-28T18:40:37.453Z"
            }
        ],
        "CallingNameStatus": "UpdateInProgress",
        "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
        "UpdatedTimestamp": "2019-08-09T21:35:31.745Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPhoneNumber。

get-room

以下代码示例演示了如何使用 get-room。

Amazon CLI

获取有关聊天室的详细信息

以下 get-room 示例显示有关指定聊天室的详细信息。

```
aws chime get-room \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

输出：

```
{
    "Room": {
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Name": "chatRoom",
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
        "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
        "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
        "UpdatedTimestamp": "2019-12-02T22:29:31.549Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRoom。

get-user-settings

以下代码示例演示了如何使用 get-user-settings。

Amazon CLI

检索用户设置

以下 get-user-settings 示例显示指定的用户设置。

```
aws chime get-user-settings \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --user-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

输出：

```
{
    "UserSettings": {
```

```
        "Telephony": {
            "InboundCalling": true,
            "OutboundCalling": true,
            "SMS": true
        }
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUserSettings。

get-user

以下代码示例演示了如何使用 get-user。

Amazon CLI

获取有关用户的详细信息

以下 get-user 示例检索指定用户的详细信息。

```
aws chime get-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

输出：

```
{
    "User": {
        "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
        "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "PrimaryEmail": "marthar@example.com",
        "DisplayName": "Martha Rivera",
        "LicenseType": "Pro",
        "UserRegistrationStatus": "Registered",
        "RegisteredOn": "2018-12-20T18:45:25.231Z",
        "InvitedOn": "2018-12-20T18:45:25.231Z",
        "AlexaForBusinessMetadata": {
            "IsAlexaForBusinessEnabled": False,
            "AlexaForBusinessRoomArn": "null"
        },
```

```
        "PersonalPIN": "XXXXXXXXXX"
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUser。

`invite-users`

以下代码示例演示了如何使用 `invite-users`。

Amazon CLI

邀请用户加入 Amazon Chime

以下 `invite-users` 示例发送一封电子邮件，邀请用户加入指定的 Amazon Chime 账户。

```
aws chime invite-users \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-email-list "alejandror@example.com" "janed@example.com"
```

输出：

```
{
    "Invites": [
        {
            "InviteId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "Status": "Pending",
            "EmailAddress": "alejandror@example.com",
            "EmailStatus": "Sent"
        }
        {
            "InviteId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "Status": "Pending",
            "EmailAddress": "janed@example.com",
            "EmailStatus": "Sent"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的邀请和暂停用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InviteUsers。

`list-accounts`

以下代码示例演示了如何使用 `list-accounts`。

Amazon CLI

获取账户列表

以下 `list-accounts` 示例检索管理员 Amazon 账户中的 Amazon Chime 账户列表。

```
aws chime list-accounts
```

输出：

```
{
    "Accounts": [
        {
            "AwsAccountId": "111122223333",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "Name": "First Chime Account",
            "AccountType": "EnterpriseDirectory",
            "CreatedTimestamp": "2018-12-20T18:38:02.181Z",
            "DefaultLicense": "Pro",
            "SupportedLicenses": [
                "Basic",
                "Pro"
            ],
            "SigninDelegateGroups": [
                {
                    "GroupName": "myGroup"
                },
            ]
        },
        {
            "AwsAccountId": "111122223333",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "Name": "Second Chime Account",
            "AccountType": "Team",
            "CreatedTimestamp": "2018-09-04T21:44:22.292Z",
            "DefaultLicense": "Pro",
```

```
            "SupportedLicenses": [
                "Basic",
                "Pro"
            ],
            "SigninDelegateGroups": [
                {
                    "GroupName": "myGroup"
                },
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理您的 Amazon Chime 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccounts。

list-bots

以下代码示例演示了如何使用 list-bots。

Amazon CLI

检索机器人列表

以下 list-bots 示例列出与指定 Amazon Chime Enterprise 账户关联的机器人。

```
aws chime list-bots \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45
```

输出：

```
{
    "Bot": {
        "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "DisplayName": "myBot (Bot)",
        "BotType": "ChatBot",
        "Disabled": false,
        "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
        "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
        "BotEmail": "myBot-chimebot@example.com",
```

```
        "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
    }
}
```

有关更多信息，请参阅《Amazon Chime 开发人员指南》中的使用 Amazon Chime 聊天机器人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBots。

`list-phone-number-orders`

以下代码示例演示了如何使用 `list-phone-number-orders`。

Amazon CLI

列出电话号码订单

以下 `list-phone-number-orders` 示例列出与 Amazon Chime 管理员账户关联的电话号码订单。

```
aws chime list-phone-number-orders
```

输出：

```
{
    "PhoneNumberOrders": [
        {
            "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
            "ProductType": "VoiceConnector",
            "Status": "Partial",
            "OrderedPhoneNumbers": [
                {
                    "E164PhoneNumber": "+12065550100",
                    "Status": "Acquired"
                },
                {
                    "E164PhoneNumber": "+12065550101",
                    "Status": "Acquired"
                },
                {
                    "E164PhoneNumber": "+12065550102",
                    "Status": "Failed"
                }
```

```
            ],
            "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
            "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
        }
        {
            "PhoneNumberOrderId": "cba54321-ed76-09f5-321g-h54i876j2109",
            "ProductType": "BusinessCalling",
            "Status": "Partial",
            "OrderedPhoneNumbers": [
                {
                    "E164PhoneNumber": "+12065550103",
                    "Status": "Acquired"
                },
                {
                    "E164PhoneNumber": "+12065550104",
                    "Status": "Acquired"
                },
                {
                    "E164PhoneNumber": "+12065550105",
                    "Status": "Failed"
                }
            ],
            "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
            "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPhoneNumberOrders。

`list-phone-numbers`

以下代码示例演示了如何使用 `list-phone-numbers`。

Amazon CLI

列出 Amazon Chime 账户的电话号码

以下 `list-phone-numbers` 示例列出与管理员的 Amazon Chime 账户关联的电话号码。

```
aws chime list-phone-numbers
```

此命令不生成任何输出。输出：

```
{
    "PhoneNumbers": [
        {
            "PhoneNumberId": "%2B12065550100",
            "E164PhoneNumber": "+12065550100",
            "Type": "Local",
            "ProductType": "VoiceConnector",
            "Status": "Assigned",
            "Capabilities": {
                "InboundCall": true,
                "OutboundCall": true,
                "InboundSMS": true,
                "OutboundSMS": true,
                "InboundMMS": true,
                "OutboundMMS": true
            },
            "Associations": [
                {
                    "Value": "abcdef1ghij2klmno3pqr4",
                    "Name": "VoiceConnectorId",
                    "AssociatedTimestamp": "2019-10-28T18:40:37.453Z"
                }
            ],
            "CallingNameStatus": "UpdateInProgress",
            "CreatedTimestamp": "2019-08-12T22:10:20.521Z",
            "UpdatedTimestamp": "2019-10-28T18:42:07.964Z"
        },
        {
            "PhoneNumberId": "%2B12065550101",
            "E164PhoneNumber": "+12065550101",
            "Type": "Local",
            "ProductType": "VoiceConnector",
            "Status": "Assigned",
            "Capabilities": {
                "InboundCall": true,
                "OutboundCall": true,
                "InboundSMS": true,
                "OutboundSMS": true,
                "InboundMMS": true,
                "OutboundMMS": true
            },
            "Associations": [
```

```
                {
                    "Value": "abcdef1ghij2klmno3pqr4",
                    "Name": "VoiceConnectorId",
                    "AssociatedTimestamp": "2019-10-28T18:40:37.511Z"
                }
            ],
            "CallingNameStatus": "UpdateInProgress",
            "CreatedTimestamp": "2019-08-12T22:10:20.521Z",
            "UpdatedTimestamp": "2019-10-28T18:42:07.960Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPhoneNumbers。

`list-room-memberships`

以下代码示例演示了如何使用 `list-room-memberships`。

Amazon CLI

列出聊天室成员资格

以下 `list-room-memberships` 示例显示指定聊天室的成员资格详细信息列表。

```
aws chime list-room-memberships \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

输出：

```
{
    "RoomMemberships": [
        {
            "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
            "Member": {
                "MemberId": "2ab2345c-67de-8901-f23g-45h678901j2k",
                "MemberType": "User",
                "Email": "zhangw@example.com",
                "FullName": "Zhang Wei",
                "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"
```

```
            },
            "Role": "Member",
            "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",
            "UpdatedTimestamp": "2019-12-02T22:46:58.532Z"
        },
        {
            "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
            "Member": {
                "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",
                "MemberType": "User",
                "Email": "janed@example.com",
                "FullName": "Jane Doe",
                "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"
            },
            "Role": "Administrator",
            "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",
            "UpdatedTimestamp": "2019-12-02T22:46:58.532Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoomMemberships。

`list-rooms`

以下代码示例演示了如何使用 `list-rooms`。

Amazon CLI

列出聊天室

以下 `list-rooms` 示例显示指定账户中的聊天室列表。列表仅筛选指定成员所属的聊天室。

```
aws chime list-rooms \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

输出：

```
{
    "Room": {
```

```
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Name": "teamRoom",
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
        "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
        "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
        "UpdatedTimestamp": "2019-12-02T22:33:19.310Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRooms。

list-users

以下代码示例演示了如何使用 `list-users`。

Amazon CLI

列出账户中的用户

以下 `list-users` 示例列出指定 Amazon Chime 账户的用户。

```
aws chime list-users --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "Users": [
        {
            "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "PrimaryEmail": "mariag@example.com",
            "DisplayName": "Maria Garcia",
            "LicenseType": "Pro",
            "UserType": "PrivateUser",
            "UserRegistrationStatus": "Registered",
            "RegisteredOn": "2018-12-20T18:45:25.231Z"
            "AlexaForBusinessMetadata": {
                "IsAlexaForBusinessEnabled": false
            }
        },
        {
```

```
            "UserId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "PrimaryEmail": "richardr@example.com",
            "DisplayName": "Richard Roe",
            "LicenseType": "Pro",
            "UserType": "PrivateUser",
            "UserRegistrationStatus": "Registered",
            "RegisteredOn": "2018-12-20T18:45:45.415Z"
            "AlexaForBusinessMetadata": {
                "IsAlexaForBusinessEnabled": false
            }
        },
        {
            "UserId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "PrimaryEmail": "saanvis@example.com",
            "DisplayName": "Saanvi Sarkar",
            "LicenseType": "Basic",
            "UserType": "PrivateUser",
            "UserRegistrationStatus": "Registered",
            "RegisteredOn": "2018-12-20T18:46:57.747Z"
            "AlexaForBusinessMetadata": {
                "IsAlexaForBusinessEnabled": false
            }
        },
        {
            "UserId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE",
            "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "PrimaryEmail": "wxiulan@example.com",
            "DisplayName": "Wang Xiulan",
            "LicenseType": "Basic",
            "UserType": "PrivateUser",
            "UserRegistrationStatus": "Registered",
            "RegisteredOn": "2018-12-20T18:47:15.390Z"
            "AlexaForBusinessMetadata": {
                "IsAlexaForBusinessEnabled": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

logout-user

以下代码示例演示了如何使用 logout-user。

Amazon CLI

注销用户

以下 logout-user 示例注销指定用户。

```
aws chime logout-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 LogoutUser。

regenerate-security-token

以下代码示例演示了如何使用 regenerate-security-token。

Amazon CLI

重新生成安全令牌

以下 regenerate-security-token 示例为指定机器人重新生成安全令牌。

```
aws chime regenerate-security-token \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k
```

输出：

```
{
    "Bot": {
        "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "DisplayName": "myBot (Bot)",
        "BotType": "ChatBot",
        "Disabled": false,
```

```
        "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
        "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
        "BotEmail": "myBot-chimebot@example.com",
        "SecurityToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY"
    }
}
```

有关更多信息，请参阅《Amazon Chime 开发人员指南》中的验证聊天机器人请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegenerateSecurityToken。

reset-personal-pin

以下代码示例演示了如何使用 reset-personal-pin。

Amazon CLI

重置用户的个人会议 PIN

以下 reset-personal-pin 示例重置指定用户的个人会议 PIN。

```
aws chime reset-personal-pin \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
    --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

输出：

```
{
    "User": {
        "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
        "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "PrimaryEmail": "mateo@example.com",
        "DisplayName": "Mateo Jackson",
        "LicenseType": "Pro",
        "UserType": "PrivateUser",
        "UserRegistrationStatus": "Registered",
        "RegisteredOn": "2018-12-20T18:45:25.231Z",
        "AlexaForBusinessMetadata": {
            "IsAlexaForBusinessEnabled": False,
            "AlexaForBusinessRoomArn": "null"
        },
        "PersonalPIN": "XXXXXXXXXX"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的更改个人会议 PIN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetPersonalPin。

`restore-phone-number`

以下代码示例演示了如何使用 `restore-phone-number`。

Amazon CLI

恢复电话号码

以下 `restore-phone-number` 示例从删除队列恢复指定电话号码。

```
aws chime restore-phone-number \
    --phone-number-id "+12065550100"
```

输出：

```
{
    "PhoneNumber": {
        "PhoneNumberId": "%2B12065550100",
        "E164PhoneNumber": "+12065550100",
        "Type": "Local",
        "ProductType": "BusinessCalling",
        "Status": "Unassigned",
        "Capabilities": {
            "InboundCall": true,
            "OutboundCall": true,
            "InboundSMS": true,
            "OutboundSMS": true,
            "InboundMMS": true,
            "OutboundMMS": true
        },
        "Associations": [],
        "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
        "UpdatedTimestamp": "2019-08-12T22:06:36.355Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestorePhoneNumber。

search-available-phone-numbers

以下代码示例演示了如何使用 search-available-phone-numbers。

Amazon CLI

搜索可用的电话号码

以下 search-available-phone-numbers 示例按区号搜索可用的电话号码。

```
aws chime search-available-phone-numbers \
    --area-code "206"
```

输出：

```
{
    "E164PhoneNumbers": [
        "+12065550100",
        "+12065550101",
        "+12065550102",
        "+12065550103",
        "+12065550104",
        "+12065550105",
        "+12065550106",
        "+12065550107",
        "+12065550108",
        "+12065550109",
    ]
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchAvailablePhoneNumbers。

update-account-settings

以下代码示例演示了如何使用 update-account-settings。

Amazon CLI

更新您的账户设置

以下 update-account-settings 示例禁用指定 Amazon Chime 账户的共享屏幕远程控制。

```
aws chime update-account-settings \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --account-settings DisableRemoteControl=true
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccountSettings。

update-account

以下代码示例演示了如何使用 update-account。

Amazon CLI

更新账户

以下 update-account 示例更新指定的账户名称。

```
aws chime update-account \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --name MyAccountName
```

输出：

```
{
    "Account": {
        "AwsAccountId": "111122223333",
        "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "Name": "MyAccountName",
        "AccountType": "Team",
        "CreatedTimestamp": "2018-09-04T21:44:22.292Z",
        "DefaultLicense": "Pro",
        "SupportedLicenses": [
            "Basic",
            "Pro"
        ],
```

```
        "SigninDelegateGroups": [
            {
                "GroupName": "myGroup"
            },
        ]
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的重命名账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccount。

update-bot

以下代码示例演示了如何使用 update-bot。

Amazon CLI

更新机器人

以下 update-bot 示例更新指定机器人的状态，使其停止运行。

```
aws chime update-bot \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k \
    --disabled
```

输出：

```
{
    "Bot": {
        "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
        "DisplayName": "myBot (Bot)",
        "BotType": "ChatBot",
        "Disabled": true,
        "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
        "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
        "BotEmail": "myBot-chimebot@example.com",
        "SecurityToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY"
    }
}
```

有关更多信息，请参阅《Amazon Chime 开发人员指南》中的更新聊天机器人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBot。

update-global-settings

以下代码示例演示了如何使用 update-global-settings。

Amazon CLI

更新全局设置

以下 update-global-settings 示例更新用于存储与管理员 Amazon 账户关联的 Amazon Chime Business Calling 和 Amazon Chime Voice Connector 的呼叫详细记录的 S3 存储桶。

```
aws chime update-global-settings \
    --business-calling CdrBucket="s3bucket" \
    --voice-connector CdrBucket="s3bucket"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理全局设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGlobalSettings。

update-phone-number-settings

以下代码示例演示了如何使用 update-phone-number-settings。

Amazon CLI

更新出站呼叫名称

以下 update-phone-number-settings 示例更新呼叫管理员 Amazon 账户的默认出站呼叫名称。

```
aws chime update-phone-number-settings \
    --calling-name "myName"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePhoneNumberSettings。

update-phone-number

以下代码示例演示了如何使用 update-phone-number。

Amazon CLI

示例 1：更新电话号码的产品类型

以下 update-phone-number 示例更新指定电话号码的产品类型。

```
aws chime update-phone-number \
    --phone-number-id "+12065550100" \
    --product-type "BusinessCalling"
```

输出：

```
{
    "PhoneNumber": {
        "PhoneNumberId": "%2B12065550100",
        "E164PhoneNumber": "+12065550100",
        "Type": "Local",
        "ProductType": "BusinessCalling",
        "Status": "Unassigned",
        "Capabilities": {
            "InboundCall": true,
            "OutboundCall": true,
            "InboundSMS": true,
            "OutboundSMS": true,
            "InboundMMS": true,
            "OutboundMMS": true
        },
        "Associations": [],
        "CallingName": "phonenumber1",
        "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
        "UpdatedTimestamp": "2019-08-12T21:44:07.591Z"
    }
}
```

示例 2：更新电话号码的出站呼叫名称

以下 update-phone-number 示例更新指定电话号码的出站呼叫名称。

aws chime update-phone-number --phone-number-id "+12065550100" --calling-name "phonenumber2"

输出：

```
{
    "PhoneNumber": {
        "PhoneNumberId": "%2B12065550100",
        "E164PhoneNumber": "+12065550100",
        "Type": "Local",
        "ProductType": "BusinessCalling",
        "Status": "Unassigned",
        "Capabilities": {
            "InboundCall": true,
            "OutboundCall": true,
            "InboundSMS": true,
            "OutboundSMS": true,
            "InboundMMS": true,
            "OutboundMMS": true
        },
        "Associations": [],
        "CallingName": "phonenumber2",
        "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
        "UpdatedTimestamp": "2019-08-12T21:44:07.591Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 管理指南》中的使用电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePhoneNumber。

update-room-membership

以下代码示例演示了如何使用 update-room-membership。

Amazon CLI

更新聊天室成员资格

以下 update-room-membership 示例会将指定聊天室成员的角色修改为 Administrator。

```
aws chime update-room-membership \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
```

```
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \
    --member-id 1ab2345c-67de-8901-f23g-45h678901j2k \
    --role Administrator
```

输出：

```
{
    "RoomMembership": {
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Member": {
            "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",
            "MemberType": "User",
            "Email": "sofiamartinez@example.com",
            "FullName": "Sofia Martinez",
            "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"
        },
        "Role": "Administrator",
        "InvitedBy": "arn:aws:iam::111122223333:user/admin",
        "UpdatedTimestamp": "2019-12-02T22:40:22.931Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoomMembership。

update-room

以下代码示例演示了如何使用 update-room。

Amazon CLI

更新聊天室

以下 update-room 示例修改指定聊天室的名称。

```
aws chime update-room \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \
    --name teamRoom
```

输出：

```
{
    "Room": {
        "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
        "Name": "teamRoom",
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
        "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
        "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
        "UpdatedTimestamp": "2019-12-02T22:33:19.310Z"
    }
}
```

有关更多信息，请参阅《Amazon Chime 用户指南》中的创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoom。

update-user-settings

以下代码示例演示了如何使用 update-user-settings。

Amazon CLI

更新用户设置

以下 update-user-settings 示例可以让指定用户进行入站和出站呼叫，并收发 SMS 消息。

```
aws chime update-user-settings \
    --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
    --user-id 1ab2345c-67de-8901-f23g-45h678901j2k \
    --user-settings "Telephony={InboundCalling=true,OutboundCalling=true,SMS=true}"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Chime 管理指南》中的管理用户电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserSettings。

update-user

以下代码示例演示了如何使用 update-user。

Amazon CLI

更新用户详细信息

此示例更新指定用户的指定详细信息。

命令:

```
aws chime update-user \
    --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE \
    --license-type "Basic"
```

输出：

```
{
    "User": {
        "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUser。

使用 Amazon CLI 的 Cloud Control API 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Cloud Control API 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-resource`

以下代码示例演示了如何使用 `create-resource`。

Amazon CLI

创建资源

以下 create-resource 示例创建一个名为 ResourceExample 的 Amazon::Kinesis::Stream 资源，其留存期为 168 小时，分片计数为 3。

```
aws cloudcontrol create-resource \
    --type-name AWS::Kinesis::Stream \
    --desired-state "{\"Name\": \"ResourceExample\",\"RetentionPeriodHours\":168,
 \"ShardCount\":3}"
```

输出：

```
{
    "ProgressEvent": {
        "EventTime": 1632506656.706,
        "TypeName": "AWS::Kinesis::Stream",
        "OperationStatus": "IN_PROGRESS",
        "Operation": "CREATE",
        "Identifier": "ResourceExample",
        "RequestToken": "20999d87-e304-4725-ad84-832dcbfd7fc5"
    }
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的创建资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResource。

delete-resource

以下代码示例演示了如何使用 delete-resource。

Amazon CLI

删除资源

以下 delete-resource 示例从您的 Amazon 账户中删除标识为 ResourceExample 的 Amazon::Kinesis::Stream 资源。

```
aws cloudcontrol delete-resource \
```

```
    --type-name AWS::Kinesis::Stream \
    --identifier ResourceExample
```

输出：

```
{
    "ProgressEvent": {
        "TypeName": "AWS::Kinesis::Stream",
        "Identifier": "ResourceExample",
        "RequestToken": "e48f26ff-d0f9-4ab8-a878-120db1edf111",
        "Operation": "DELETE",
        "OperationStatus": "IN_PROGRESS",
        "EventTime": 1632950300.14
    }
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的删除资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResource。

get-resource-request-status

以下代码示例演示了如何使用 get-resource-request-status。

Amazon CLI

获取资源请求的状态信息

以下 get-resource-request-status 示例返回指定资源请求的状态信息。

```
aws cloudcontrol get-resource-request-status \
    --request-token "e1a6b86e-46bd-41ac-bfba-001234567890"
```

输出：

```
{
    "ProgressEvent": {
        "TypeName": "AWS::Kinesis::Stream",
        "Identifier": "Demo",
        "RequestToken": "e1a6b86e-46bd-41ac-bfba-001234567890",
        "Operation": "CREATE",
```

```
        "OperationStatus": "FAILED",
        "EventTime": 1632950268.481,
        "StatusMessage": "Resource of type 'AWS::Kinesis::Stream' with identifier
 'Demo' already exists.",
        "ErrorCode": "AlreadyExists"
    }
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的管理资源操作请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceRequestStatus。

`get-resource`

以下代码示例演示了如何使用 `get-resource`。

Amazon CLI

获取资源的当前状态

以下 `get-resource` 示例返回名为 ResourceExample 的 Amazon::Kinesis::Stream 资源的当前状态。

```
aws cloudcontrol get-resource \
    --type-name AWS::Kinesis::Stream \
    --identifier ResourceExample
```

输出：

```
{
    "TypeName": "AWS::Kinesis::Stream",
    "ResourceDescription": {
        "Identifier": "ResourceExample",
        "Properties": "{\"Arn\":\"arn:aws:kinesis:us-west-2:099908667365:stream/
ResourceExample\",\"RetentionPeriodHours\":168,\"Name\":\"ResourceExample\",
\"ShardCount\":3}"
    }
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的读取资源的当前状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResource。

list-resource-requests

以下代码示例演示了如何使用 list-resource-requests。

Amazon CLI

列出活动资源操作请求

以下 list-resource-requests 示例列出您 Amazon 账户中已失败的“CREATE”和“UPDATE”操作的资源请求。

```
aws cloudcontrol list-resource-requests \
    --resource-request-status-filter Operations=CREATE,OperationStatuses=FAILED
```

输出：

```
{
    "ResourceRequestStatusSummaries": [
        {
            "TypeName": "AWS::Kinesis::Stream",
            "Identifier": "Demo",
            "RequestToken": "e1a6b86e-46bd-41ac-bfba-633abcdfdbd7",
            "Operation": "CREATE",
            "OperationStatus": "FAILED",
            "EventTime": 1632950268.481,
            "StatusMessage": "Resource of type 'AWS::Kinesis::Stream' with
 identifier 'Demo' already exists.",
            "ErrorCode": "AlreadyExists"
        }
    ]
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的管理资源操作请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceRequests。

list-resources

以下代码示例演示了如何使用 list-resources。

Amazon CLI

列出给定类型的资源

以下 `list-resources` 示例列出在您 Amazon 账户中预置的 Amazon::Kinesis::Stream 资源。

```
aws cloudcontrol list-resources \
    --type-name AWS::Kinesis::Stream
```

输出：

```
{
    "TypeName": "AWS::Kinesis::Stream",
    "ResourceDescriptions": [
        {
            "Identifier": "MyKinesisStream",
            "Properties": "{\"Name\":\"MyKinesisStream\"}"
        },
        {
            "Identifier": "AnotherStream",
            "Properties": "{\"Name\":\"AnotherStream\"}"
        }
    ]
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的发现资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResources。

update-resource

以下代码示例演示了如何使用 `update-resource`。

Amazon CLI

更新现有资源的属性

以下 `update-resource` 示例会将名为 ExampleLogGroup 的 Amazon::Logs::LogGroup 资源的留存策略更新为 90 天。

```
aws cloudcontrol update-resource \
    --type-name AWS::Logs::LogGroup \
    --identifier ExampleLogGroup \
    --patch-document "[{\"op\":\"replace\",\"path\":\"/RetentionInDays\",\"value
\":90}]"
```

输出：

```
{
    "ProgressEvent": {
        "EventTime": "2021-08-09T18:17:15.219Z",
        "TypeName": "AWS::Logs::LogGroup",
        "OperationStatus": "IN_PROGRESS",
        "Operation": "UPDATE",
        "Identifier": "ExampleLogGroup",
        "RequestToken": "5f40c577-3534-4b20-9599-0b0123456789"
    }
}
```

有关更多信息，请参阅《Cloud Control API 用户指南》中的更新资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResource。

Amazon Cloud Map使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Cloud Map 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-http-namespace`

以下代码示例演示了如何使用 `create-http-namespace`。

Amazon CLI

创建 HTTP 命名空间

以下 create-http-namespace 示例创建 HTTP 命名空间 example.com。

```
aws servicediscovery create-http-namespace \
    --name example.com \
    --creator-request-id example-request-id
```

输出：

```
{
    "OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd"
}
```

要确认操作成功，可以运行 get-operation。有关更多信息，请参阅 get-operation。

有关创建命名空间的更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Creating an Amazon Cloud Map namespace to group application services。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateHttpNamespace。

create-private-dns-namespace

以下代码示例演示了如何使用 create-private-dns-namespace。

Amazon CLI

创建私有 DNS 命名空间

以下 create-private-dns-namespace 示例创建一个私有 DNS 命名空间。

```
aws servicediscovery create-private-dns-namespace \
    --name example.com \
    --vpc vpc-1c56417b
```

输出：

```
{
    "OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd"
}
```

要确认操作成功，可以运行 get-operation。有关更多信息，请参阅 get-operation。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的创建命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePrivateDnsNamespace。

`create-public-dns-namespace`

以下代码示例演示了如何使用 `create-public-dns-namespace`。

Amazon CLI

创建公有 DNS 命名空间

以下 `create-public-dns-namespace` 示例创建公有 DNS 命名空间 `example.com`。

```
aws servicediscovery create-public-dns-namespace \
    --name example-public-dns.com \
    --creator-request-id example-public-request-id \
    --properties DnsProperties={SOA={TTL=60}}
```

输出：

```
{
    "OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd"
}
```

要确认操作成功，可以运行 `get-operation`。

有关创建命名空间的更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Creating an Amazon Cloud Map namespace to group application services。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreatePublicDnsNamespace。

`create-service`

以下代码示例演示了如何使用 `create-service`。

Amazon CLI

示例 1：使用命名空间 ID 创建服务

以下 `create-service` 示例创建一个服务。

```
aws servicediscovery create-service \
    --name myservice \
    --namespace-id  ns-ylexjili4cdxy3xm \
    --dns-config "RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=60}]"
```

输出：

```
{
    "Service": {
        "Id": "srv-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Name": "myservice",
        "NamespaceId": "ns-abcd1234xmpl5678",
        "DnsConfig": {
            "NamespaceId": "ns-abcd1234xmpl5678",
            "RoutingPolicy": "MULTIVALUE",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 60
                }
            ]
        },
        "Type": "DNS_HTTP",
        "CreateDate": "2025-08-18T13:45:31.023000-05:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678",
        "CreatedByAccount": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的为应用程序组件创建 Amazon Cloud Map 服务。

示例 2：使用命名空间 ARN 创建服务

以下 `create-service` 示例使用命名空间 ARN（而不是命名空间 ID）来创建服务。在共享命名空间中创建服务时，必须指定命名空间 ARN。

```
aws servicediscovery create-service \
    --name myservice-arn \
```

```
    --namespace-id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678 \
    --dns-config "RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=60}]"
```

输出：

```
{
    "Service": {
        "Id": "srv-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Name": "myservice-arn",
        "NamespaceId": "ns-abcd1234xmpl5678",
        "DnsConfig": {
            "NamespaceId": "ns-abcd1234xmpl5678",
            "RoutingPolicy": "MULTIVALUE",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 60
                }
            ]
        },
        "Type": "DNS_HTTP",
        "CreateDate": "2025-08-18T13:45:31.023000-05:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678",
        "CreatedByAccount": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateService。

`delete-namespace`

以下代码示例演示了如何使用 `delete-namespace`。

Amazon CLI

示例 1：删除命名空间

以下 delete-namespace 示例删除一个命名空间。

```
aws servicediscovery delete-namespace \
    --id ns-abcd1234xmpl5678
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

要确认操作成功，可以运行 get-operation。有关更多信息，请参阅 get-operation。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的删除 Amazon Cloud Map 命名空间。

示例 2：使用命名空间 ARN 删除命名空间

以下 delete-namespace 示例使用命名空间 ARN 删除命名空间。

```
aws servicediscovery delete-namespace \
    --id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的删除 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNamespace。

delete-service-attributes

以下代码示例演示了如何使用 delete-service-attributes。

Amazon CLI

示例 1：删除服务属性

以下 delete-service-attributes 示例删除具有与指定服务关联的键 Port 的服务属性。

```
aws servicediscovery delete-service-attributes \
    --service-id srv-abcd1234xmpl5678 \
    --attributes Port
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map services。

示例 2：使用 ARN 删除服务属性

以下 delete-service-attributes 示例使用服务 ARN 删除服务属性。对于与您的账户共享的命名空间，要删除与这些命名空间中创建的服务关联的属性，必须指定 ARN。

```
aws servicediscovery delete-service-attributes \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678 \
    --attributes Port
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的 Amazon Cloud Map 服务和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteServiceAttributes。

delete-service

以下代码示例演示了如何使用 delete-service。

Amazon CLI

示例 1：删除服务

以下 delete-service 示例删除一项服务。

```
aws servicediscovery delete-service \
    --id srv-abcd1234xmpl5678
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的删除 Amazon Cloud Map 服务。

示例 2：使用 ARN 删除服务

以下 delete-service 示例使用服务 ARN 删除服务。

```
aws servicediscovery delete-service \
    --id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的删除 Amazon Cloud Map 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteService。

deregister-instance

以下代码示例演示了如何使用 deregister-instance。

Amazon CLI

示例 1：取消注册服务实例

以下 deregister-instance 示例取消注册一个服务实例。

```
aws servicediscovery deregister-instance \
    --service-id srv-p5zdwlg5uvvzjita \
    --instance-id myservice-53
```

输出：

```
{
    "OperationId": "4yejorelbukcjzpnr6tlmrghsjwpngf4-k98rnaiq"
}
```

要确认操作成功，可以运行 get-operation。有关更多信息，请参阅 get-operation。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的取消注册服务实例。

示例 2：使用共享命名空间的服务 ARN 取消注册服务实例

以下 deregister-instance 示例使用服务 ARN（而不是服务 ID）取消注册服务实例。要从与您的账户共享的命名空间中创建的服务取消注册实例，必须指定 ARN。

```
aws servicediscovery deregister-instance \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita \
    --instance-id web-server-01
```

输出：

```
{
    "OperationId": "gv4g5meo7ndmkqjrhpn39wk42xmpl"
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间和取消注册 Amazon Cloud Map 服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterInstance。

discover-instances-revision

以下代码示例演示了如何使用 discover-instances-revision。

Amazon CLI

示例 1：发现实例的修订

以下 discover-instances-revision 示例发现不断增加的实例修订版。

```
aws servicediscovery discover-instances-revision \
    --namespace-name example.com \
    --service-name myservice
```

输出：

```
{
    "InstancesRevision": 123456
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map service instances。

示例 2：从特定所有者账户中发现实例的修订

以下 discover-instances-revision 示例从特定所有者账户中发现实例的修订。对于与您的账户共享的命名空间中的实例，必须使用 owner-account 参数。

```
aws servicediscovery discover-instances-revision \
    --namespace-name shared-namespace \
    --service-name shared-service \
    --owner-account 123456789111
```

输出：

```
{
    "InstancesRevision": 1234567890
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间和 Amazon Cloud Map 服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DiscoverInstancesRevision。

discover-instances

以下代码示例演示了如何使用 discover-instances。

Amazon CLI

示例 1：发现已注册实例

以下 discover-instances 示例查找一个已注册实例。

```
aws servicediscovery discover-instances \
    --namespace-name example.com \
    --service-name myservice \
    --max-results 10 \
    --health-status ALL
```

输出：

```
{
    "Instances": [
        {
            "InstanceId": "myservice-53",
            "NamespaceName": "example.com",
            "ServiceName": "myservice",
            "HealthStatus": "UNKNOWN",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "172.2.1.3",
                "AWS_INSTANCE_PORT": "808"
            }
        }
    ],
    "InstancesRevision": 85648075627387284
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map service instances。

示例 2：从特定所有者账户中发现实例

以下 discover-instances 示例从特定所有者账户中发现已注册实例。要发现与您的账户共享的命名空间中的实例，必须使用此参数。

```
aws servicediscovery discover-instances \
    --namespace-name shared-namespace \
    --service-name shared-service \
    --owner-account 123456789111
```

输出：

```
{
    "Instances": [
        {
            "InstanceId": "shared-instance-1234",
            "NamespaceName": "shared-namespace",
            "ServiceName": "shared-service",
            "HealthStatus": "HEALTHY",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "203.0.113.75",
```

```
                "AWS_INSTANCE_PORT": "80"
            }
        }
    ],
    "InstancesRevision": 1234567890
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间和 Amazon Cloud Map 服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DiscoverInstances。

get-instance

以下代码示例演示了如何使用 get-instance。

Amazon CLI

示例 1：获取实例的详细信息

以下 get-instance 示例获取服务的属性。

```
aws servicediscovery get-instance \
    --service-id srv-e4anhexample0004
    --instance-id i-abcd1234
```

输出：

```
{
    "ResourceOwner": "123456789012",
    "Instance": {
        "Id": "arn:aws:servicediscovery:us-west-2:111122223333;:service/srv-
e4anhexample0004",
        "Attributes": {
            "AWS_INSTANCE_IPV4": "192.0.2.44",
            "AWS_INSTANCE_PORT": "80",
            "color": "green",
            "region": "us-west-2",
            "stage": "beta"
        },
        "CreatedByAccount": "123456789012"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map service instances。

示例 2：使用共享命名空间的服务 ARN 获取实例的详细信息

以下 `get-instance` 示例使用服务 ARN（而不是服务 ID）获取实例的属性。对于与您的账户共享的命名空间，在获取与这些命名空间关联的实例的详细信息时，必须指定 ARN。此示例中返回的实例是由在账户 123456789012 拥有的命名空间中的账户 123456789111 注册的。

```
aws servicediscovery get-instance \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita \
    --instance-id web-server-01
```

输出：

```
{
    "ResourceOwner": "123456789012",
    "Instance": {
        "Id": "web-server-01",
        "Attributes": {
            "AWS_INSTANCE_IPV4": "203.0.113.15",
            "AWS_INSTANCE_PORT": "80"
        },
        "CreatedByAccount": "123456789111"
    }
}
```

有关跨账户命名空间共享的更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstance。

get-instances-health-status

以下代码示例演示了如何使用 `get-instances-health-status`。

Amazon CLI

示例 1：获取与服务关联的实例的运行状况

以下 get-instances-health-status 示例获取与指定服务关联的实例的运行状况。

```
aws servicediscovery get-instances-health-status \
    --service-id srv-e4anhexample0004
```

输出：

```
{
    "Status": {
        "i-abcd1234": "HEALTHY",
        "i-abcd1235": "UNHEALTHY"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map service instances。

示例 2：使用共享命名空间的服务 ARN 获取实例的运行状况

以下 get-instances-health-status 示例使用服务 ARN（而不是服务 ID）获取实例的运行状况。对于与请求者的账户共享的命名空间，在获取与这些命名空间关联的实例的运行状况时，必须指定 ARN。

```
aws servicediscovery get-instances-health-status \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita
```

输出：

```
{
    "Status": {
        "web-server-01": "HEALTHY",
        "web-server-02": "UNHEALTHY"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的 Amazon Cloud Map 服务实例和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetInstancesHealthStatus。

get-namespace

以下代码示例演示了如何使用 get-namespace。

Amazon CLI

示例 1：获取命名空间的详细信息

以下 get-namespace 示例检索有关指定命名空间的信息。

```
aws servicediscovery get-namespace \
    --id ns-abcd1234xmpl5678
```

输出：

```
{
    "Namespace": {
        "Id": "ns-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Name": "example-http.com",
        "Type": "HTTP",
        "Description": "Example.com AWS Cloud Map HTTP Namespace",
        "Properties": {
            "DnsProperties": {},
            "HttpProperties": {
                "HttpName": "example-http.com"
            }
        },
        "CreateDate": "2024-02-23T13:35:21.874000-06:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

示例 2：使用 ARN 获取命名空间的详细信息

以下 get-namespace 示例使用命名空间 ARN 检索有关指定命名空间的信息。要检索与您的账户共享的命名空间的详细信息，必须指定 ARN。

```
aws servicediscovery get-namespace \
    --id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678
```

输出：

```
{
    "Namespace": {
        "Id": "ns-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Name": "example-http.com",
        "Type": "HTTP",
        "Description": "Example.com AWS Cloud Map HTTP Namespace",
        "Properties": {
            "DnsProperties": {},
            "HttpProperties": {
                "HttpName": "example-http.com"
            }
        },
        "CreateDate": "2024-02-23T13:35:21.874000-06:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetNamespace。

get-operation

以下代码示例演示了如何使用 get-operation。

Amazon CLI

示例 1：获取操作结果

以下 get-operation 示例获取命名空间创建操作的结果。

```
aws servicediscovery get-operation \
```

```
    --operation-id abcd1234xmpl5678abcd1234xmpl5678-abcd1234
```

输出：

```
{
    "Operation": {
        "Id": "abcd1234xmpl5678abcd1234xmpl5678-abcd1234",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": "2025-01-13T13:35:21.874000-06:00",
        "UpdateDate": "2025-01-13T13:36:02.469000-06:00",
        "Targets": {
            "NAMESPACE": "ns-abcd1234xmpl5678"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Creating an Amazon Cloud Map namespace to group application services。

示例 2：从特定所有者账户获取操作

以下 `get-operation` 示例获取与特定命名空间所有者账户关联的操作的结果。要获取与您的账户共享的命名空间关联的操作的结果，必须使用此参数。

```
aws servicediscovery get-operation \
    --operation-id abcd1234xmpl5678abcd1234xmpl5678-abcd1234 \
    --owner-account 123456789111
```

输出：

```
{
    "Operation": {
        "Id": "abcd1234xmpl5678abcd1234xmpl5678-abcd1234",
        "OwnerAccount": "123456789111",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": "2025-01-13T13:35:21.874000-06:00",
        "UpdateDate": "2025-01-13T13:36:02.469000-06:00",
        "Targets": {
            "NAMESPACE": "ns-abcd1234xmpl5678"
        }
```

```
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOperation。

get-service-attributes

以下代码示例演示了如何使用 get-service-attributes。

Amazon CLI

示例 1：获取服务的属性

以下 get-service-attributes 示例获取服务的属性。

```
aws servicediscovery get-service-attributes \
    --service-id srv-abcd1234xmpl5678
```

输出：

```
{
    "ServiceAttributes": {
        "ServiceArn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Attributes": {
            "Port": "80"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map services。

示例 2：使用 ARN 获取服务的属性

以下 get-service-attributes 示例使用服务 ARN 获取服务的属性。对于与您的账户共享的命名空间，要获取在该命名空间中创建的服务的属性，必须指定 ARN。

```
aws servicediscovery get-service-attributes \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678
```

输出：

```
{
    "ServiceAttributes": {
        "ServiceArn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Attributes": {
            "Port": "80"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的 Amazon Cloud Map 服务和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetServiceAttributes。

get-service

以下代码示例演示了如何使用 get-service。

Amazon CLI

示例 1：获取服务的设置

以下 get-service 示例获取指定服务的设置。

```
aws servicediscovery get-service \
    --id srv-abcd1234xmpl5678
```

输出：

```
{
    "Service": {
        "Id": "srv-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
```

```
        "ResourceOwner": "123456789012",
        "Name": "test-service",
        "NamespaceId": "ns-abcd1234xmpl5678",
        "DnsConfig": {},
        "Type": "HTTP",
        "CreateDate": "2025-08-18T13:53:02.775000-05:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678",
        "CreatedByAccount": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map services。

示例 2：使用 ARN 获取服务的设置

以下 `get-service` 示例使用服务 ARN 获取指定服务的设置。对于与您的账户共享的命名空间，要检索在该命名空间中创建的服务的信息，必须指定 ARN。调用者账户 123456789111 已在账户 123456789012 共享的命名空间中创建了服务。

```
aws servicediscovery get-service \
    --id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678
```

输出：

```
{
    "Service": {
        "Id": "srv-abcd1234xmpl5678",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678",
        "ResourceOwner": "123456789012",
        "Name": "test-service",
        "NamespaceId": "ns-abcd1234xmpl5678",
        "DnsConfig": {},
        "Type": "HTTP",
        "CreateDate": "2025-08-18T13:53:02.775000-05:00",
        "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678",
        "CreatedByAccount": "123456789111"
    }
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的为应用程序组件创建 Amazon Cloud Map 服务和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetService。

list-instances

以下代码示例演示了如何使用 list-instances。

Amazon CLI

示例 1：列出服务实例

以下 list-instances 示例列出服务实例。

```
aws servicediscovery list-instances \
    --service-id srv-qzpwvt2tfqcegapy
```

输出：

```
{
    "Instances": [
        {
            "Id": "i-06bdabbae60f65a4e",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "172.2.1.3",
                "AWS_INSTANCE_PORT": "808"
            },
            "CreatedByAccount": "123456789012"
        }
    ],
    "ResourceOwner": "123456789012"
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的列出 Amazon Cloud Map 服务实例。

示例 2：使用服务 ARN 列出服务实例

以下 list-instances 示例使用服务 ARN（而不是服务 ID）列出服务实例。对于与您的账户共享的命名空间，在列出与这些命名空间关联的实例时，必须指定 ARN。

```
aws servicediscovery list-instances \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
p5zdwlg5uvvzjita
```

输出：

```
{
    "ResourceOwner": "123456789012",
    "Instances": [
        {
            "Id": "web-server-01",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "203.0.113.15",
                "AWS_INSTANCE_PORT": "80"
            },
            "CreatedByAccount": "123456789012"
        },
        {
            "Id": "web-server-02",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "203.0.113.16",
                "AWS_INSTANCE_PORT": "80"
            },
            "CreatedByAccount": "123456789012"
        }
    ]
}
```

有关跨账户命名空间共享的更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间和列出 Amazon Cloud Map 服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstances。

`list-namespaces`

以下代码示例演示了如何使用 `list-namespaces`。

Amazon CLI

示例 1：列出命名空间

以下 `list-namespaces` 示例列出命名空间。

```
aws servicediscovery list-namespaces
```

输出：

```
{
    "Namespaces": [
        {
            "Id": "ns-abcd1234xmpl5678",
            "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678",
            "ResourceOwner": "123456789012",
            "Name": "local",
            "Type": "DNS_PRIVATE",
            "Properties": {
                "DnsProperties": {
                    "HostedZoneId": "Z06752353VBUDTC32S84S",
                    "SOA": {}
                },
                "HttpProperties": {
                    "HttpName": "local"
                 }
            },
            "CreateDate": "2023-07-17T13:37:27.872000-05:00"
        },
        {
            "Id": "ns-abcd1234xmpl9012",
            "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl9012",
            "ResourceOwner": "123456789012",
            "Name": "My-second-namespace",
            "Type": "HTTP",
            "Description": "My second namespace",
            "Properties": {
                "DnsProperties": {
                    "SOA": {}
                },
                "HttpProperties": {
                    "HttpName": "My-second-namespace"
                }
            },
            "CreateDate": "2023-11-14T10:35:47.840000-06:00"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的列出 Amazon Cloud Map 命名空间。

示例 2：列出其他账户共享的命名空间

以下 `list-namespaces` 示例使用 RESOURCE_OWNER 筛选条件列出通过其他 Amazon 账户与调用者账户共享的命名空间。

```
aws servicediscovery list-namespaces \
    --filters Name=RESOURCE_OWNER,Values=OTHER_ACCOUNTS,Condition=EQ
```

输出：

```
{
    "Namespaces": [
        {
            "Id": "ns-abcd1234xmpl5678",
            "Arn": "arn:aws:servicediscovery:us-west-2:123456789111:namespace/ns-
abcd1234xmpl5678",
            "ResourceOwner": "123456789111",
            "Name": "shared-namespace",
            "Type": "HTTP",
            "Description": "Namespace shared from another account",
            "Properties": {
                "DnsProperties": {
                    "SOA": {}
                },
                "HttpProperties": {
                    "HttpName": "shared-namespace"
                }
            },
            "CreateDate": "2025-01-13T13:35:21.874000-06:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListNamespaces。

list-operations

以下代码示例演示了如何使用 list-operations。

Amazon CLI

列出符合指定条件的操作

以下 list-operations 示例列出了状态为 PENDING 或 SUCCESS 的操作。

```
aws servicediscovery list-operations \
    --service-id srv-e4anhexample0004 \
    --filters Name=STATUS,Condition=IN,Values=PENDING,SUCCESS
```

输出：

```
{
    "Operations": [
        {
            "Id": "76yy8ovhpdz0plmjzbsnqgnrqvpv2qdt-kexample",
            "Status": "SUCCESS"
        },
        {
            "Id": "prysnyzpji3u2ciy45nke83x2zanl7yk-dexample",
            "Status": "SUCCESS"
        },
        {
            "Id": "ko4ekftir7kzlbechsh7xvcdgcpk66gh-7example",
            "Status": "PENDING"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 What is Amazon Cloud Map?

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOperations。

list-services

以下代码示例演示了如何使用 list-services。

Amazon CLI

示例 1：列出服务

以下 `list-services` 示例列出服务。

```
aws servicediscovery list-services
```

输出：

```
{
    "Services": [
        {
            "Id": "srv-p5zdwlg5uvvzjita",
            "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
p5zdwlg5uvvzjita",
            "Name": "myservice",
            "DnsConfig": {
                "RoutingPolicy": "MULTIVALUE",
                "DnsRecords": [
                    {
                        "Type": "A",
                        "TTL": 60
                    }
                ]
            },
            "CreateDate": 1587081768.334
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的列出命名空间中的 Amazon Cloud Map 服务。

示例 2：列出在共享命名空间中创建的服务

以下 `list-services` 示例使用 RESOURCE_OWNER 筛选条件列出在通过其他 Amazon 账户与调用者账户 123456789012 共享的命名空间中创建的服务。

```
aws servicediscovery list-services \
    --filters Name=RESOURCE_OWNER,Values=OTHER_ACCOUNTS,Condition=EQ
```

输出：

```
{
    "Services": [
        {
            "Id": "srv-abcd1234xmpl5678",
            "Arn": "arn:aws:servicediscovery:us-west-2:123456789111:service/srv-
abcd1234xmpl5678",
            "ResourceOwner": "123456789111",
            "Name": "shared-service",
            "NamespaceId": "ns-abcd1234xmpl5678",
            "Type": "HTTP",
            "Description": "Service in shared namespace",
            "DnsConfig": {},
            "CreateDate": "2025-01-13T13:35:21.874000-06:00",
            "CreatorRequestId": "abcd1234-5678-90ab-cdef-xmpl12345678",
            "CreatedByAccount": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间和列出命名空间中的 Amazon Cloud Map 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出与指定资源关联的标签

以下 `list-tags-for-resource` 示例列出指定资源的标签。

```
aws servicediscovery list-tags-for-resource \
    --resource-arn arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
e4anhexample0004
```

输出：

```
{
```

```
    "Tags": [
        {
            "Key": "Project",
            "Value": "Zeta"
        },
        {
            "Key": "Department",
            "Value": "Engineering"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Tagging your Amazon Cloud Map resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

register-instance

以下代码示例演示了如何使用 register-instance。

Amazon CLI

示例 1：使用服务 ID 注册服务实例

以下 register-instance 示例注册一个服务实例。

```
aws servicediscovery register-instance \
    --service-id srv-p5zdwlg5uvvzjita \
    --instance-id myservice-53 \
    --attributes=AWS_INSTANCE_IPV4=172.2.1.3,AWS_INSTANCE_PORT=808
```

输出：

```
{
    "OperationId": "4yejorelbukcjzpnr6tlmrghsjwpngf4-k95yg2u7"
}
```

要确认操作成功，可以运行 get-operation。有关更多信息，请参阅 get-operation。

有关注册实例的更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的将资源注册为 Amazon Cloud Map 服务实例。

示例 2：使用服务 ARN 注册服务实例

以下 register-instance 示例使用服务 ARN 注册服务实例。在与您的账户共享的服务中注册实例时，必须指定 ARN。

```
aws servicediscovery register-instance \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita \
    --instance-id web-server-01 \
    --attributes=AWS_INSTANCE_IPV4=203.0.113.15,AWS_INSTANCE_PORT=80
```

输出：

```
{
    "OperationId": "gv4g5meo7ndmkqjrhpn39wk42xmpl"
}
```

有关跨账户命名空间共享的更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterInstance。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签与指定的资源相关联

以下 tag-resource 示例将值为 Engineering 的 Department 标签与指定的命名空间关联。

```
aws servicediscovery tag-resource \
    --resource-arn arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-e4anhexample0004 \
    --tags Key=Department, Value=Engineering
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Tagging your Amazon Cloud Map resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从指定的资源中移除标签

以下 untag-resource 示例从指定的命名空间中移除 Department 标签。

```
aws servicediscovery untag-resource \
    --resource-arn arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-e4anhexample0004 \
    --tags Key=Department, Value=Engineering
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Tagging your Amazon Cloud Map resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-http-namespace

以下代码示例演示了如何使用 update-http-namespace。

Amazon CLI

示例 1：更新 HTTP 命名空间

以下 update-http-namespace 示例更新指定 HTTP 命名空间的描述。

```
aws servicediscovery update-http-namespace \
    --id ns-abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
```

```
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

要确认操作成功，可以运行 `get-operation`。有关更多信息，请参阅 get-operation。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

示例 2：使用 ARN 更新 HTTP 命名空间

以下 `update-http-namespace` 示例使用命名空间 ARN 更新指定的 HTTP 命名空间。

```
aws servicediscovery update-http-namespace \
    --id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateHttpNamespace。

update-instance-custom-health-status

以下代码示例演示了如何使用 `update-instance-custom-health-status`。

Amazon CLI

示例 1：更新自定义运行状况检查

以下 `update-instance-custom-health-status` 示例将指定服务和示例服务实例的自定义运行状况检查的状态更新为 HEALTHY。

```
aws servicediscovery update-instance-custom-health-status \
```

```
    --service-id srv-e4anhexample0004 \
    --instance-id example \
    --status HEALTHY
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map service health check configuration。

示例 2：使用服务 ARN 更新自定义运行状况检查

以下 update-instance-custom-health-status 示例使用服务 ARN 更新自定义运行状况检查的状态。对于与您的账户共享的命名空间，在更新与这些命名空间关联的实例的运行状况时，必须指定 ARN。

```
aws servicediscovery update-instance-custom-health-status \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita \
    --instance-id web-server-01 \
    --status HEALTHY
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的 Amazon Cloud Map 服务运行状况检查配置和跨账户 Amazon Cloud Map 命名空间共享。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateInstanceCustomHealthStatus。

update-private-dns-namespace

以下代码示例演示了如何使用 update-private-dns-namespace。

Amazon CLI

示例 1：使用 ID 更新私有 DNS 命名空间

以下 update-private-dns-namespace 示例使用命名空间 ID 更新私有 DNS 命名空间的描述。

```
aws servicediscovery update-private-dns-namespace \
```

```
    --id ns-abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

要确认操作成功，可以运行 get-operation。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

示例 2：使用 ARN 更新私有 DNS 命名空间

以下 update-private-dns-namespace 示例使用命名空间 ARN 更新私有 DNS 命名空间。

```
aws servicediscovery update-private-dns-namespace \
    --id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdatePrivateDnsNamespace。

update-public-dns-namespace

以下代码示例演示了如何使用 update-public-dns-namespace。

Amazon CLI

示例 1：使用 ID 更新公共 DNS 命名空间

以下 update-public-dns-namespace 示例使用命名空间 ID 更新公共 DNS 命名空间的描述。

```
aws servicediscovery update-public-dns-namespace \
    --id ns-abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

要确认操作成功，可以运行 get-operation。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

示例 2：使用 ARN 更新公共 DNS 命名空间

以下 update-public-dns-namespace 示例使用命名空间 ARN 更新公共 DNS 命名空间。

```
aws servicediscovery update-public-dns-namespace \
    --id arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcd1234xmpl5678 \
    --updater-request-id abcd1234-5678-90ab-cdef-xmpl12345678 \
    --namespace Description="The updated namespace description."
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map namespaces。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdatePublicDnsNamespace。

update-service-attributes

以下代码示例演示了如何使用 update-service-attributes。

Amazon CLI

示例 1：更新服务以添加属性

以下 update-service-attributes 示例更新指定的服务，以添加带有键 Port 和值 80 的服务属性。

```
aws servicediscovery update-service-attributes \
    --service-id srv-abcd1234xmpl5678 \
    --attributes Port=80
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Amazon Cloud Map services。

示例 2：使用 ARN 更新服务属性

以下 update-service-attributes 示例使用服务 ARN 更新服务以添加服务属性。要向在与您的账户共享的命名空间中创建的服务添加属性，必须指定 ARN。

```
aws servicediscovery update-service-attributes \
    --service-id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-abcd1234xmpl5678 \
    --attributes Port=80
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的 Amazon Cloud Map 服务和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateServiceAttributes。

update-service

以下代码示例演示了如何使用 update-service。

Amazon CLI

示例 1：更新服务

以下 update-service 示例更新服务以更新 DnsConfig 和 HealthCheckConfig 设置。

```
aws servicediscovery update-service \
    --id srv-abcd1234xmpl5678 \
    --
service "DnsConfig={DnsRecords=[{Type=A,TTL=60}]},HealthCheckConfig={Type=HTTP,ResourcePath=
```

输出：

```
{
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

要确认操作成功，可以运行 get-operation。

有关更新服务的更多信息，请参阅《Amazon Cloud Map Developer Guide》中的 Updating an Amazon Cloud Map service。

示例 2：使用 ARN 更新服务

以下 update-service 示例使用服务 ARN 更新服务。对于在与您的账户共享的命名空间中创建的服务，必须指定 ARN。

```
aws servicediscovery update-service \
    --id arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
abcd1234xmpl5678 \
    --
service "DnsConfig={DnsRecords=[{Type=A,TTL=60}]},HealthCheckConfig={Type=HTTP,ResourcePath=
```

输出：

```
{
```

```
    "OperationId": "abcd1234-5678-90ab-cdef-xmpl12345678"
}
```

有关更新服务的更多信息，请参阅《Amazon Cloud Map 开发人员指南》中的更新 Amazon Cloud Map 服务和共享的 Amazon Cloud Map 命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateService。

使用 Amazon CLI 的 Amazon Cloud9 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Cloud9 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-environment-ec2`

以下代码示例演示了如何使用 `create-environment-ec2`。

Amazon CLI

创建 Amazon Cloud9 EC2 开发环境

以下 `create-environment-ec2` 示例使用指定设置创建 Amazon Cloud9 开发环境，启动 Amazon Elastic Compute Cloud（Amazon EC2）实例，然后将该实例连接到环境。

```
aws cloud9 create-environment-ec2 \
    --name my-demo-env \
    --description "My demonstration development environment." \
    --instance-type t2.micro --image-id amazonlinux-2023-x86_64 \
    --subnet-id subnet-1fab8aEX \
```

```
    --automatic-stop-time-minutes 60 \
    --owner-arn arn:aws:iam::123456789012:user/MyDemoUser
```

输出：

```
{
    "environmentId": "8a34f51ce1e04a08882f1e811bd706EX"
}
```

有关更多信息，请参阅《Amazon Cloud9 用户指南》中的创建 EC2 环境。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEnvironmentEc2。

`create-environment-membership`

以下代码示例演示了如何使用 `create-environment-membership`。

Amazon CLI

将环境成员添加到 Amazon Cloud9 开发环境中

此示例会将指定的环境成员添加到指定的 Amazon Cloud9 开发环境中。

命令:

```
aws cloud9 create-environment-membership --environment-
id 8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser --permissions read-write
```

输出：

```
{
  "membership": {
    "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
    "userId": "AIDAJ3LOROMOUXTBSU6EX",
    "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
    "permissions": "read-write"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEnvironmentMembership。

delete-environment-membership

以下代码示例演示了如何使用 delete-environment-membership。

Amazon CLI

从 Amazon Cloud9 开发环境中删除环境成员

此示例从指定的 Amazon Cloud9 开发环境中删除指定的环境成员。

命令:

```
aws cloud9 delete-environment-membership --environment-
id 8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEnvironmentMembership。

delete-environment

以下代码示例演示了如何使用 delete-environment。

Amazon CLI

删除 Amazon Cloud9 开发环境

此示例删除指定的 Amazon Cloud9 开发环境。如果一个 Amazon EC2 实例连接到该环境，也会终止该实例。

命令:

```
aws cloud9 delete-environment --environment-id 8a34f51ce1e04a08882f1e811bd706EX
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEnvironment。

describe-environment-memberships

以下代码示例演示了如何使用 describe-environment-memberships。

Amazon CLI

获取有关 Amazon Cloud9 开发环境的环境成员信息

此示例获取有关指定 Amazon Cloud9 开发环境的环境成员信息。

命令:

```
aws cloud9 describe-environment-memberships --environment-
id 8a34f51ce1e04a08882f1e811bd706EX
```

输出：

```
{
  "memberships": [
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJ3LOROMOUXTBSU6EX",
      "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
      "permissions": "read-write"
    },
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

获取有关 Amazon Cloud9 开发环境所有者的信息

此示例获取有关指定 Amazon Cloud9 开发环境所有者的信息。

命令:

```
aws cloud9 describe-environment-memberships --environment-
id 8a34f51ce1e04a08882f1e811bd706EX --permissions owner
```

输出：

```
{
  "memberships": [
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

获取有关多个 Amazon Cloud9 开发环境的环境成员信息

此示例获取有关多个 Amazon Cloud9 开发环境的指定环境成员的信息。

命令:

```
aws cloud9 describe-environment-memberships --user-
arn arn:aws:iam::123456789012:user/MyDemoUser
```

输出：

```
{
  "memberships": [
    {
      "environmentId": "10a75714bd494714929e7f5ec4125aEX",
      "lastAccess": 1516213427.0,
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    },
    {
      "environmentId": "1980b80e5f584920801c09086667f0EX",
      "lastAccess": 1516144884.0,
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironmentMemberships。

describe-environment-status

以下代码示例演示了如何使用 describe-environment-status。

Amazon CLI

获取 Amazon Cloud9 开发环境的状态信息

此示例获取指定 Amazon Cloud9 开发环境的状态信息。

命令:

```
aws cloud9 describe-environment-status --environment-
id 685f892f431b45c2b28cb69eadcdb0EX
```

输出：

```
{
  "status": "ready",
  "message": "Environment is ready to use"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironmentStatus。

describe-environments

以下代码示例演示了如何使用 describe-environments。

Amazon CLI

获取有关 Amazon Cloud9 开发环境的信息

此示例获取有关指定 Amazon Cloud9 开发环境的信息。

命令:

```
aws cloud9 describe-environments --environment-
ids 685f892f431b45c2b28cb69eadcdb0EX 349c86d4579e4e7298d500ff57a6b2EX
```

输出：

```
{
  "environments": [
    {
      "id": "685f892f431b45c2b28cb69eadcdb0EX",
      "name": "my-demo-ec2-env",
      "description": "Created from CodeStar.",
      "type": "ec2",
      "arn": "arn:aws:cloud9:us-
east-1:123456789012:environment:685f892f431b45c2b28cb69eadcdb0EX",
      "ownerArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "lifecycle": {
        "status": "CREATED"
      }
    },
    {
      "id": "349c86d4579e4e7298d500ff57a6b2EX",
      "name": my-demo-ssh-env",
      "description": "",
      "type": "ssh",
      "arn": "arn:aws:cloud9:us-
east-1:123456789012:environment:349c86d4579e4e7298d500ff57a6b2EX",
      "ownerArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "lifecycle": {
        "status": "CREATED"
      }
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironments。

`list-environments`

以下代码示例演示了如何使用 `list-environments`。

Amazon CLI

获取可用的 Amazon Cloud9 开发环境标识符列表

此示例获取可用的 Amazon Cloud9 开发环境标识符列表。

命令:

```
aws cloud9 list-environments
```

输出：

```
{
  "environmentIds": [
    "685f892f431b45c2b28cb69eadcdb0EX",
    "1980b80e5f584920801c09086667f0EX"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEnvironments。

update-environment-membership

以下代码示例演示了如何使用 update-environment-membership。

Amazon CLI

更改 Amazon Cloud9 开发环境的现有环境成员的设置

此示例更改指定 Amazon Cloud9 开发环境的指定现有环境成员的设置。

命令:

```
aws cloud9 update-environment-membership --environment-
id 8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser --permissions read-only
```

输出：

```
{
  "membership": {
    "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
    "userId": "AIDAJ3LOROMOUXTBSU6EX",
    "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
    "permissions": "read-only"
  }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEnvironmentMembership。

update-environment

以下代码示例演示了如何使用 update-environment。

Amazon CLI

更改现有 Amazon Cloud9 开发环境的设置

此示例更改指定现有 Amazon Cloud9 开发环境的指定设置。

命令:

```
aws cloud9 update-environment --environment-id 8a34f51ce1e04a08882f1e811bd706EX
 --name my-changed-demo-env --description "My changed demonstration development
 environment."
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEnvironment。

使用 Amazon CLI 的 Amazon CloudFormation 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon CloudFormation 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`activate-type`

以下代码示例演示了如何使用 `activate-type`。

Amazon CLI

激活类型

以下 `activate-type` 示例激活公有第三方扩展，使其可用于堆栈模板。

```
aws cloudformation activate-type \
    --region us-west-2 \
    --type RESOURCE \
    --type-name Example::Test::1234567890abcdef0 \
    --type-name-alias Example::Test::Alias
```

输出：

```
{
    "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/Example-
Test-Alias"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ActivateType。

`batch-describe-type-configurations`

以下代码示例演示了如何使用 `batch-describe-type-configurations`。

Amazon CLI

批量描述类型配置

以下 `batch-describe-type-configurations` 示例配置特定类型的数据。

```
aws cloudformation batch-describe-type-configurations \
    --region us-west-2 \
```

```
    --type-configuration-identifiers TypeArn="arn:aws:cloudformation:us-
west-2:123456789012:type/resource/Example-Test-
Type,TypeConfigurationAlias=MyConfiguration"
```

输出：

```
{
    "Errors": [],
    "UnprocessedTypeConfigurations": [],
    "TypeConfigurations": [
        {
            "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/
Example-Test-Type",
            "Alias": "MyConfiguration",
            "Configuration": "{\n        \"Example\": {\n            \"ApiKey\":
 \"examplekey\",\n            \"ApplicationKey\": \"examplekey1\",\n
 \"ApiURL\": \"exampleurl\"\n            }\n}",
            "LastUpdated": "2021-10-01T15:25:46.210000+00:00",
            "TypeArn": "arn:aws:cloudformation:us-east-1:123456789012:type/resource/
Example-Test-Type"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDescribeTypeConfigurations。

cancel-update-stack

以下代码示例演示了如何使用 cancel-update-stack。

Amazon CLI

取消正在进行的堆栈更新

以下 cancel-update-stack 命令会取消 myteststack 堆栈上的堆栈更新：

```
aws cloudformation cancel-update-stack --stack-name myteststack
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelUpdateStack。

continue-update-rollback

以下代码示例演示了如何使用 continue-update-rollback。

Amazon CLI

重试更新回滚

以下 continue-update-rollback 示例从之前失败的堆栈更新中恢复回滚操作。

```
aws cloudformation continue-update-rollback \
    --stack-name my-stack
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ContinueUpdateRollback。

create-change-set

以下代码示例演示了如何使用 create-change-set。

Amazon CLI

创建更改集

以下 create-change-set 示例创建一个具有 CAPABILITY_IAM 功能的更改集。文件 template.yaml 是当前文件夹中的 Amazon CloudFormation 模板，用于定义包含 IAM 资源的堆栈。

```
aws cloudformation create-change-set \
    --stack-name my-application \
    --change-set-name my-change-set \
    --template-body file://template.yaml \
    --capabilities CAPABILITY_IAM
```

输出：

```
{
    "Id": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-change-set/
bc9555ba-a949-xmpl-bfb8-f41d04ec5784",
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-application/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204"
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChangeSet。

create-generated-template

以下代码示例演示了如何使用 create-generated-template。

Amazon CLI

根据扫描的资源创建生成的模板

以下 create-generated-template 示例根据扫描的资源创建一个名为 MyTemplate 的生成模板。

```
aws cloudformation create-generated-template \
    --generated-template-name MyTemplate \
    --resources file://resources.json
```

的内容resources.json：

```
[
    {
        "ResourceType": "AWS::EKS::Cluster",
        "LogicalResourceId":"MyCluster",
        "ResourceIdentifier": {
            "ClusterName": "MyAppClusterName"
        }
    },
    {
        "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
        "LogicalResourceId":"MyASG",
        "ResourceIdentifier": {
            "AutoScalingGroupName": "MyAppASGName"
        }
    },
    {
        "ResourceType": "AWS::EKS::Nodegroup",
        "LogicalResourceId":"MyNodegroup",
        "ResourceIdentifier": {
            "NodegroupName": "MyAppNodegroupName"
        }
```

```
    },
    {
        "ResourceType": "AWS::IAM::Role",
        "LogicalResourceId":"MyRole",
        "ResourceIdentifier": {
            "RoleId": "arn:aws::iam::123456789012:role/MyAppIAMRole"
        }
    }
]
```

输出：

```
{
  "Arn":
    "arn:aws:cloudformation:us-east-1:123456789012:generatedtemplate/7fc8512c-
d8cb-4e02-b266-d39c48344e48",
  "Name": "MyTemplate"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 IaC 生成器根据扫描的资源创建 CloudFormation 模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGeneratedTemplate。

`create-stack-instances`

以下代码示例演示了如何使用 `create-stack-instances`。

Amazon CLI

创建堆栈实例

以下 `create-stack-instances` 示例在两个账户和四个区域中创建堆栈集的实例。容错设置可确保在所有账户和地区尝试更新，即使某些堆栈无法创建。

```
aws cloudformation create-stack-instances \
    --stack-set-name my-stack-set \
    --accounts 123456789012 223456789012 \
    --regions us-east-1 us-east-2 us-west-1 us-west-2 \
    --operation-preferences FailureToleranceCount=7
```

输出：

```
{
    "OperationId": "d7995c31-83c2-xmpl-a3d4-e9ca2811563f"
}
```

使用 create-stack-set 命令创建堆栈。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStackInstances。

create-stack-refactor

以下代码示例演示了如何使用 create-stack-refactor。

Amazon CLI

为堆栈重构操作创建堆栈定义

以下 create-stack-refactor 示例为堆栈重构创建堆栈定义。

```
aws cloudformation create-stack-refactor \
    --stack-definitions \
      StackName=Stack1,TemplateBody@=file://template1-updated.yaml \
      StackName=Stack2,TemplateBody@=file://template2-updated.yaml
```

输出：

```
{
    "StackRefactorId": "9c384f70-4e07-4ed7-a65d-fee5eb430841"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的堆栈重构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStackRefactor。

create-stack-set

以下代码示例演示了如何使用 create-stack-set。

Amazon CLI

创建堆栈集

以下 create-stack-set 示例使用指定的 YAML 文件模板创建堆栈集。template.yaml 是当前文件夹中用于定义堆栈的 Amazon CloudFormation 模板。

```
aws cloudformation create-stack-set \
    --stack-set-name my-stack-set \
    --template-body file://template.yaml \
    --description "SNS topic"
```

输出：

```
{
    "StackSetId": "my-stack-set:8d0f160b-d157-xmpl-a8e6-c0ce8e5d8cc1"
}
```

使用 create-stack-instances 命令将堆栈实例添加到堆栈集中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStackSet。

create-stack

以下代码示例演示了如何使用 create-stack。

Amazon CLI

创建 Amazon CloudFormation 堆栈

以下 create-stacks 命令将使用 sampletemplate.json 模板创建名为 myteststack 的堆栈：

```
aws cloudformation create-stack --stack-name myteststack --template-body file://
sampletemplate.json --parameters ParameterKey=KeyPairName,ParameterValue=TestKey
 ParameterKey=SubnetIDs,ParameterValue=SubnetID1\\,SubnetID2
```

输出：

```
{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的“堆栈”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStack。

deactivate-type

以下代码示例演示了如何使用 deactivate-type。

Amazon CLI

停用类型

以下 deactivate-type 示例停用先前在此账户和区域中激活的公有扩展。

```
aws cloudformation deactivate-type \
    --region us-west-2 \
    --type MODULE \
    --type-name Example::Test::Type::MODULE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeactivateType。

delete-change-set

以下代码示例演示了如何使用 delete-change-set。

Amazon CLI

删除更改集

以下 delete-change-set 示例通过指定更改集名称和堆栈名称来删除更改集。

```
aws cloudformation delete-change-set \
    --stack-name my-stack \
    --change-set-name my-change-set
```

此命令不生成任何输出。

以下 delete-change-set 示例通过指定更改集的完整 ARN 来删除更改集。

```
aws cloudformation delete-change-set \
    --change-set-name arn:aws:cloudformation:us-east-2:123456789012:changeSet/my-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteChangeSet。

delete-generated-template

以下代码示例演示了如何使用 delete-generated-template。

Amazon CLI

删除生成的模板

以下 delete-generated-template 示例删除指定的模板。

```
aws cloudformation delete-generated-template \
    --generated-template-name MyTemplate
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的根据现有资源生成模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGeneratedTemplate。

delete-stack-instances

以下代码示例演示了如何使用 delete-stack-instances。

Amazon CLI

删除堆栈实例

以下 delete-stack-instances 示例删除两个区域中两个账户的堆栈集实例，并终止堆栈。

```
aws cloudformation delete-stack-instances \
    --stack-set-name my-stack-set \
    --accounts 123456789012 567890123456 \
    --regions us-east-1 us-west-1 \
    --no-retain-stacks
```

输出：

```
{
    "OperationId": "ad49f10c-fd1d-413f-a20a-8de6e2fa8f27"
}
```

使用 delete-stack-set 命令删除空堆栈集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStackInstances。

delete-stack-set

以下代码示例演示了如何使用 delete-stack-set。

Amazon CLI

删除堆栈集

以下命令删除指定的空堆栈集。堆栈集必须为空。

```
aws cloudformation delete-stack-set \
    --stack-set-name my-stack-set
```

此命令不生成任何输出。

使用 delete-stack-instances 命令从堆栈集中删除实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStackSet。

delete-stack

以下代码示例演示了如何使用 delete-stack。

Amazon CLI

删除堆栈

以下 delete-stack 示例删除指定的堆栈。

```
aws cloudformation delete-stack \
    --stack-name my-stack
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStack。

deploy

以下代码示例演示了如何使用 deploy。

Amazon CLI

以下命令会将名为 template.json 的模板部署到名为 my-new-stack 的堆栈中：

```
aws cloudformation deploy --template-file /path_to_template/template.json
 --stack-name my-new-stack --parameter-overrides Key1=Value1 Key2=Value2 --
tags Key1=Value1 Key2=Value2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Deploy。

deregister-type

以下代码示例演示了如何使用 deregister-type。

Amazon CLI

取消注册类型版本

以下 deregister-type 示例会将指定的类型版本从 CloudFormation 注册表中移除，使其无法再在 CloudFormation 操作中使用。

```
aws cloudformation deregister-type \
    --type RESOURCE \
    --type-name My::Logs::LogGroup \
    --version-id 00000002
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterType。

describe-account-limits

以下代码示例演示了如何使用 describe-account-limits。

Amazon CLI

获取有关您账户限制的信息

以下命令检索当前账户的区域限制列表。

```
aws cloudformation describe-account-limits
```

输出：

```
{
    "AccountLimits": [
        {
            "Name": "StackLimit",
            "Value": 200
        },
        {
            "Name": "StackOutputsLimit",
            "Value": 60
        },
        {
            "Name": "ConcurrentResourcesLimit",
            "Value": 2500
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountLimits。

describe-change-set

以下代码示例演示了如何使用 describe-change-set。

Amazon CLI

获取有关更改集的信息

以下 describe-change-set 示例显示由更改集名称和堆栈名称指定的更改集的详细信息。

```
aws cloudformation describe-change-set \
    --change-set-name my-change-set \
```

```
    --stack-name my-stack
```

以下 describe-change-set 示例显示由更改集的完整 ARN 指定的更改集的详细信息：

```
aws cloudformation describe-change-set \
    --change-set-name arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-change-set/bc9555ba-a949-xmpl-bfb8-f41d04ec5784
```

输出：

```
{
    "Changes": [
        {
            "Type": "Resource",
            "ResourceChange": {
                "Action": "Modify",
                "LogicalResourceId": "function",
                "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
                "ResourceType": "AWS::Lambda::Function",
                "Replacement": "False",
                "Scope": [
                    "Properties"
                ],
                "Details": [
                    {
                        "Target": {
                            "Attribute": "Properties",
                            "Name": "Timeout",
                            "RequiresRecreation": "Never"
                        },
                        "Evaluation": "Static",
                        "ChangeSource": "DirectModification"
                    }
                ]
            }
        }
    ],
    "ChangeSetName": "my-change-set",
    "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "StackName": "my-stack",
```

```
    "Description": null,
    "Parameters": null,
    "CreationTime": "2019-10-02T05:20:56.651Z",
    "ExecutionStatus": "AVAILABLE",
    "Status": "CREATE_COMPLETE",
    "StatusReason": null,
    "NotificationARNs": [],
    "RollbackConfiguration": {},
    "Capabilities": [
        "CAPABILITY_IAM"
    ],
    "Tags": null
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeChangeSet。

describe-generated-template

以下代码示例演示了如何使用 describe-generated-template。

Amazon CLI

描述生成的模板

以下 describe-generated-template 示例描述了指定的模板。

```
aws cloudformation describe-generated-template \
    --generated-template-name MyTemplate
```

输出：

```
{
    "GeneratedTemplateId": "arn:aws:cloudformation:us-
east-1:123456789012:generatedTemplate/7d881acf-f307-4ded-910e-f8fb49b96894",
    "GeneratedTemplateName": "MyTemplate",
    "Resources": [
        {
            "ResourceType": "AWS::EC2::SecurityGroup",
            "LogicalResourceId": "EC2SecurityGroup",
            "ResourceIdentifier": {
                "Id": "sg-1234567890abcdef0"
            },
            "ResourceStatus": "COMPLETE",
```

```
            "ResourceStatusReason": "Resource Template complete",
            "Warnings": []
        },
        {
            "ResourceType": "AWS::EC2::Instance",
            "LogicalResourceId": "EC2Instance",
            "ResourceIdentifier": {
                "InstanceId": "i-1234567890abcdef0"
            },
            "ResourceStatus": "COMPLETE",
            "ResourceStatusReason": "Resource Template complete",
            "Warnings": []
        },
        {
            "ResourceType": "AWS::EC2::KeyPair",
            "LogicalResourceId": "EC2KeyPairSshkeypair",
            "ResourceIdentifier": {
                "KeyName": "sshkeypair"
            },
            "ResourceStatus": "COMPLETE",
            "ResourceStatusReason": "Resource Template complete",
            "Warnings": []
        }
    ],
    "Status": "COMPLETE",
    "StatusReason": "All resources complete",
    "CreationTime": "2025-09-23T19:38:06.435000+00:00",
    "LastUpdatedTime": "2025-09-23T19:38:10.798000+00:00",
    "Progress": {
        "ResourcesSucceeded": 3,
        "ResourcesFailed": 0,
        "ResourcesProcessing": 0,
        "ResourcesPending": 0
    },
    "TemplateConfiguration": {
        "DeletionPolicy": "RETAIN",
        "UpdateReplacePolicy": "RETAIN"
    },
    "TotalWarnings": 0
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的根据现有资源生成模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGeneratedTemplate。

describe-publisher

以下代码示例演示了如何使用 describe-publisher。

Amazon CLI

描述发布者

以下 describe-publisher 示例配置发布者的信息。

```
aws cloudformation describe-publisher \
    --region us-west-2 \
    --publisher-id 000q6TfUovXsEMmgKowxDZLlwqr2QUsh
```

输出：

```
{
    "PublisherId": "000q6TfUovXsEMmgKowxDZLlwqr2QUshd2e75c8c",
    "PublisherStatus": "VERIFIED",
    "IdentityProvider": "AWS_Marketplace",
    "PublisherProfile": "https://aws.amazon.com/marketplace/seller-profile?
id=2c5dc1f0-17cd-4259-8e46-822a83gdtegd"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePublisher。

describe-resource-scan

以下代码示例演示了如何使用 describe-resource-scan。

Amazon CLI

描述资源扫描

以下 describe-resource-scan 示例描述具有所指定扫描 ID 的资源扫描。

```
aws cloudformation describe-resource-scan --region \
    --resource-scan-id arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60
```

输出：

```
{
    "ResourceScanId": "arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60",
    "Status": "COMPLETE",
    "StartTime": "2025-08-21T03:10:38.485000+00:00",
    "EndTime": "2025-08-21T03:20:28.485000+00:00",
    "PercentageCompleted": 100.0,
    "ResourceTypes": [
        "AWS::CloudFront::CachePolicy",
        "AWS::CloudFront::OriginRequestPolicy",
        "AWS::EC2::DHCPOptions",
        "AWS::EC2::InternetGateway",
        "AWS::EC2::KeyPair",
        "AWS::EC2::NetworkAcl",
        "AWS::EC2::NetworkInsightsPath",
        "AWS::EC2::NetworkInterface",
        "AWS::EC2::PlacementGroup",
        "AWS::EC2::Route",
        "AWS::EC2::RouteTable",
        "AWS::EC2::SecurityGroup",
        "AWS::EC2::Subnet",
        "AWS::EC2::SubnetCidrBlock",
        "AWS::EC2::SubnetNetworkAclAssociation",
        "AWS::EC2::SubnetRouteTableAssociation",
        ...
    ],
    "ResourcesRead": 676
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的根据现有资源生成模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResourceScan。

describe-stack-drift-detection-status

以下代码示例演示了如何使用 describe-stack-drift-detection-status。

Amazon CLI

检查偏离检测操作的状态

以下 describe-stack-drift-detection-status 示例显示偏离检测操作的状态。运行 detect-stack-drift 命令获取该 ID。

```
aws cloudformation describe-stack-drift-detection-status \
    --stack-drift-detection-id 1a229160-e4d9-xmpl-ab67-0a4f93df83d4
```

输出：

```
{
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "StackDriftDetectionId": "1a229160-e4d9-xmpl-ab67-0a4f93df83d4",
    "StackDriftStatus": "DRIFTED",
    "DetectionStatus": "DETECTION_COMPLETE",
    "DriftedStackResourceCount": 1,
    "Timestamp": "2019-10-02T05:54:30.902Z"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackDriftDetectionStatus。

describe-stack-events

以下代码示例演示了如何使用 describe-stack-events。

Amazon CLI

描述堆栈事件

以下 describe-stack-events 示例显示指定堆栈的 2 个最新事件。

```
aws cloudformation describe-stack-events \
    --stack-name my-stack \
    --max-items 2

{
    "StackEvents": [
        {
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "EventId": "4e1516d0-e4d6-xmpl-b94f-0a51958a168c",
            "StackName": "my-stack",
```

```
            "LogicalResourceId": "my-stack",
            "PhysicalResourceId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "ResourceType": "AWS::CloudFormation::Stack",
            "Timestamp": "2019-10-02T05:34:29.556Z",
            "ResourceStatus": "UPDATE_COMPLETE"
        },
        {
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "EventId": "4dd3c810-e4d6-xmpl-bade-0aaf8b31ab7a",
            "StackName": "my-stack",
            "LogicalResourceId": "my-stack",
            "PhysicalResourceId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "ResourceType": "AWS::CloudFormation::Stack",
            "Timestamp": "2019-10-02T05:34:29.127Z",
            "ResourceStatus": "UPDATE_COMPLETE_CLEANUP_IN_PROGRESS"
        }
    ],

 "NextToken": "eyJOZXh0VG9XMPLiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackEvents。

describe-stack-instance

以下代码示例演示了如何使用 describe-stack-instance。

Amazon CLI

描述堆栈实例

以下命令描述指定账户和区域中指定堆栈集的实例。堆栈集在当前区域和账户中，实例在账户 123456789012 中的 us-west-2 区域中：

```
aws cloudformation describe-stack-instance \
    --stack-set-name my-stack-set \
    --stack-instance-account 123456789012 \
    --stack-instance-region us-west-2
```

输出：

```
{
    "StackInstance": {
        "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
        "Region": "us-west-2",
        "Account": "123456789012",
        "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/
StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532/4287f9a0-e615-
xmpl-894a-12b31d3117be",
        "ParameterOverrides": [],
        "Status": "OUTDATED",
        "StatusReason": "ResourceLogicalId:ConfigBucket,
 ResourceType:AWS::S3::Bucket, ResourceStatusReason:You have attempted to create
 more buckets than allowed (Service: Amazon S3; Status Code: 400; Error Code:
 TooManyBuckets; Request ID: F7F21CXMPL580224; S3 Extended Request ID: egd/
Fdt89BXMPLyiqbMNljVk55Yqqvi3NYW2nKLUVWhUGEhNfCmZdyj967lhriaG/dWMobSO40o=)."
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackInstance。

describe-stack-refactor

以下代码示例演示了如何使用 describe-stack-refactor。

Amazon CLI

描述堆栈重构操作

以下 describe-stack-refactor 示例描述了具有所指定堆栈重构 ID 的堆栈重构操作。

```
aws cloudformation describe-stack-refactor \
    --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

输出：

```
{
    "StackRefactorId": "9c384f70-4e07-4ed7-a65d-fee5eb430841",
    "StackIds": [
        "arn:aws:cloudformation:us-east-1:123456789012:stack/
Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
        "arn:aws:cloudformation:us-east-1:123456789012:stack/
Stack2/5da91650-94b1-11f0-81cf-0a23500e151b"
```

```
    ],
    "ExecutionStatus": "AVAILABLE",
    "Status": "CREATE_COMPLETE"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的堆栈重构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackRefactor。

describe-stack-resource-drifts

以下代码示例演示了如何使用 describe-stack-resource-drifts。

Amazon CLI

获取有关偏离堆栈定义的资源的信息

以下命令显示有关指定堆栈中偏离资源的信息。使用 detect-stack-drift 命令启动偏离检测：

```
aws cloudformation describe-stack-resource-drifts \
    --stack-name my-stack
```

输出显示了由带外数据修改的 Amazon Lambda 函数：

```
{
    "StackResourceDrifts": [
        {
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "LogicalResourceId": "function",
            "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
            "ResourceType": "AWS::Lambda::Function",
            "ExpectedProperties": "{\"Description\":\"Write a file to S3.\",
\"Environment\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf
\"}},\"Handler\":\"index.handler\",\"MemorySize\":128,\"Role\":
\"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLEOM9E\",\"Runtime\":
\"nodejs10.x\",\"Tags\":[{\"Key\":\"lambda:createdBy\",\"Value\":\"SAM\"}],\"Timeout
\":900,\"TracingConfig\":{\"Mode\":\"Active\"}}",
            "ActualProperties": "{\"Description\":\"Write a file to S3.\",
\"Environment\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf
\"}},\"Handler\":\"index.handler\",\"MemorySize\":256,\"Role\":
\"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLEOM9E\",\"Runtime\":
```

```
\"nodejs10.x\",\"Tags\":[{\"Key\":\"lambda:createdBy\",\"Value\":\"SAM\"}],\"Timeout
\":22,\"TracingConfig\":{\"Mode\":\"Active\"}}",
            "PropertyDifferences": [
                {
                    "PropertyPath": "/MemorySize",
                    "ExpectedValue": "128",
                    "ActualValue": "256",
                    "DifferenceType": "NOT_EQUAL"
                },
                {
                    "PropertyPath": "/Timeout",
                    "ExpectedValue": "900",
                    "ActualValue": "22",
                    "DifferenceType": "NOT_EQUAL"
                }
            ],
            "StackResourceDriftStatus": "MODIFIED",
            "Timestamp": "2019-10-02T05:54:44.064Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackResourceDrifts。

describe-stack-resource

以下代码示例演示了如何使用 describe-stack-resource。

Amazon CLI

获取有关堆栈资源的信息

以下 describe-stack-resource 示例显示指定堆栈中名为 MyFunction 的资源的详细信息。

```
aws cloudformation describe-stack-resource \
    --stack-name MyStack \
    --logical-resource-id MyFunction
```

输出：

```
{
    "StackResourceDetail": {
        "StackName": "MyStack",
```

```
        "StackId": "arn:aws:cloudformation:us-east-2:123456789012:stack/MyStack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "LogicalResourceId": "MyFunction",
        "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
        "ResourceType": "AWS::Lambda::Function",
        "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",
        "ResourceStatus": "UPDATE_COMPLETE",
        "Metadata": "{}",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackResource。

describe-stack-resources

以下代码示例演示了如何使用 describe-stack-resources。

Amazon CLI

获取有关堆栈资源的信息

以下 describe-stack-resources 示例显示指定堆栈中的资源详细信息。

```
aws cloudformation describe-stack-resources \
    --stack-name my-stack
```

输出：

```
{
    "StackResources": [
        {
            "StackName": "my-stack",
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "LogicalResourceId": "bucket",
            "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",
            "ResourceType": "AWS::S3::Bucket",
            "Timestamp": "2019-10-02T04:34:11.345Z",
            "ResourceStatus": "CREATE_COMPLETE",
            "DriftInformation": {
```

```
                "StackResourceDriftStatus": "IN_SYNC"
            }
        },
        {
            "StackName": "my-stack",
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "LogicalResourceId": "function",
            "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
            "ResourceType": "AWS::Lambda::Function",
            "Timestamp": "2019-10-02T05:34:27.989Z",
            "ResourceStatus": "UPDATE_COMPLETE",
            "DriftInformation": {
                "StackResourceDriftStatus": "IN_SYNC"
            }
        },
        {
            "StackName": "my-stack",
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "LogicalResourceId": "functionRole",
            "PhysicalResourceId": "my-functionRole-HIZXMPLEOM9E",
            "ResourceType": "AWS::IAM::Role",
            "Timestamp": "2019-10-02T04:34:06.350Z",
            "ResourceStatus": "CREATE_COMPLETE",
            "DriftInformation": {
                "StackResourceDriftStatus": "IN_SYNC"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackResources。

`describe-stack-set-operation`

以下代码示例演示了如何使用 `describe-stack-set-operation`。

Amazon CLI

获取有关堆栈集操作的信息

以下“describe-stack-set-operation`”示例显示对指定堆栈集执行更新操作的详细信息。

```
aws cloudformation describe-stack-set-operation \
    --stack-set-name enable-config \
    --operation-id 35d45ebc-ed88-xmpl-ab59-0197a1fc83a0
```

输出：

```
{
    "StackSetOperation": {
        "OperationId": "35d45ebc-ed88-xmpl-ab59-0197a1fc83a0",
        "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
        "Action": "UPDATE",
        "Status": "SUCCEEDED",
        "OperationPreferences": {
            "RegionOrder": [
                "us-east-1",
                "us-west-2",
                "eu-west-1",
                "us-west-1"
            ],
            "FailureToleranceCount": 7,
            "MaxConcurrentCount": 2
        },
        "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
        "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
        "CreationTimestamp": "2019-10-03T16:28:44.377Z",
        "EndTimestamp": "2019-10-03T16:42:08.607Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参数》中的 DescribeStackSetOperation。

describe-stack-set

以下代码示例演示了如何使用 describe-stack-set。

Amazon CLI

获取有关堆栈集的信息

以下“describe-stack-set`”示例显示有关指定堆栈集的详细信息。

```
aws cloudformation describe-stack-set \
    --stack-set-name my-stack-set
```

输出：

```
{
    "StackSet": {
        "StackSetName": "my-stack-set",
        "StackSetId": "my-stack-set:296a3360-xmpl-40af-be78-9341e95bf743",
        "Description": "Create an Amazon SNS topic",
        "Status": "ACTIVE",
        "TemplateBody": "AWSTemplateFormatVersion: '2010-09-09'\nDescription: An AWS
 SNS topic\nResources:\n  topic:\n    Type: AWS::SNS::Topic",
        "Parameters": [],
        "Capabilities": [],
        "Tags": [],
        "StackSetARN": "arn:aws:cloudformation:us-west-2:123456789012:stackset/
enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
        "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
        "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStackSet。

describe-stacks

以下代码示例演示了如何使用 describe-stacks。

Amazon CLI

描述 Amazon CloudFormation 堆栈

以下 describe-stacks 命令显示 myteststack 堆栈的摘要信息：

```
aws cloudformation describe-stacks --stack-name myteststack
```

输出：

```
{
```

```
    "Stacks":  [
        {
            "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
            "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample
 template showing how to create a publicly accessible S3 bucket. **WARNING** This
 template creates an S3 bucket. You will be billed for the AWS resources used if you
 create a stack from this template.",
            "Tags": [],
            "Outputs": [
                {
                    "Description": "Name of S3 bucket to hold website content",
                    "OutputKey": "BucketName",
                    "OutputValue": "myteststack-s3bucket-jssofi1zie2w"
                }
            ],
            "StackStatusReason": null,
            "CreationTime": "2013-08-23T01:02:15.422Z",
            "Capabilities": [],
            "StackName": "myteststack",
            "StackStatus": "CREATE_COMPLETE",
            "DisableRollback": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的“堆栈”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStacks。

describe-type-registration

以下代码示例演示了如何使用 describe-type-registration。

Amazon CLI

显示类型注册信息

以下 describe-type-registration 示例显示有关指定类型注册的信息，包括该类型的当前状态、类型和版本。

```
aws cloudformation describe-type-registration \
    --registration-token a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "ProgressStatus": "COMPLETE",
    "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup",
    "Description": "Deployment is currently in DEPLOY_STAGE of status COMPLETED; ",
    "TypeVersionArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup/00000001"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTypeRegistration。

describe-type

以下代码示例演示了如何使用 describe-type。

Amazon CLI

显示类型信息

以下 describe-type 示例显示指定类型的信息。

```
aws cloudformation describe-type \
    --type-name My::Logs::LogGroup \
    --type RESOURCE
```

输出：

```
{
    "SourceUrl": "https://github.com/aws-cloudformation/aws-cloudformation-resource-providers-logs.git",
    "Description": "Customized resource derived from AWS::Logs::LogGroup",
    "TimeCreated": "2019-12-03T23:29:33.321Z",
    "Visibility": "PRIVATE",
    "TypeName": "My::Logs::LogGroup",
    "LastUpdated": "2019-12-03T23:29:33.321Z",
    "DeprecatedStatus": "LIVE",
    "ProvisioningType": "FULLY_MUTABLE",
    "Type": "RESOURCE",
```

```
    "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-
LogGroup/00000001",
    "Schema": "[details omitted]"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeType。

detect-stack-drift

以下代码示例演示了如何使用 detect-stack-drift。

Amazon CLI

检测偏离的资源

以下 detect-stack-drift 示例对指定堆栈启动偏离检测。

```
aws cloudformation detect-stack-drift \
    --stack-name my-stack
```

输出：

```
{
    "StackDriftDetectionId": "1a229160-e4d9-xmpl-ab67-0a4f93df83d4"
}
```

然后，您可以将此 ID 与 describe-stack-resource-drifts 命令一起使用来描述偏离的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectStackDrift。

detect-stack-resource-drift

以下代码示例演示了如何使用 detect-stack-resource-drift。

Amazon CLI

检测资源的偏离

以下 detect-stack-resource-drift 示例检查名为 MyStack 的堆栈中名为 MyFunction 的资源是否偏离：

```
aws cloudformation detect-stack-resource-drift \
   --stack-name MyStack \
   --logical-resource-id MyFunction
```

输出显示了由带外数据修改的 Amazon Lambda 函数：

```
{
    "StackResourceDrift": {
        "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/MyStack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "LogicalResourceId": "MyFunction",
        "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
        "ResourceType": "AWS::Lambda::Function",
        "ExpectedProperties": "{\"Description\":\"Write a file to S3.\",
\"Environment\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf
\"}},\"Handler\":\"index.handler\",\"MemorySize\":128,\"Role\":
\"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLEOM9E\",\"Runtime\":
\"nodejs10.x\",\"Tags\":[{\"Key\":\"lambda:createdBy\",\"Value\":\"SAM\"}],\"Timeout
\":900,\"TracingConfig\":{\"Mode\":\"Active\"}}",
        "ActualProperties": "{\"Description\":\"Write a file to S3.\",\"Environment
\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf\"}},\"Handler\":
\"index.handler\",\"MemorySize\":256,\"Role\":\"arn:aws:iam::123456789012:role/
my-functionRole-HIZXMPLEOM9E\",\"Runtime\":\"nodejs10.x\",\"Tags\":[{\"Key\":
\"lambda:createdBy\",\"Value\":\"SAM\"}],\"Timeout\":22,\"TracingConfig\":{\"Mode\":
\"Active\"}}",
        "PropertyDifferences": [
            {
                "PropertyPath": "/MemorySize",
                "ExpectedValue": "128",
                "ActualValue": "256",
                "DifferenceType": "NOT_EQUAL"
            },
            {
                "PropertyPath": "/Timeout",
                "ExpectedValue": "900",
                "ActualValue": "22",
                "DifferenceType": "NOT_EQUAL"
            }
        ],
        "StackResourceDriftStatus": "MODIFIED",
```

```
        "Timestamp": "2019-10-02T05:58:47.433Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectStackResourceDrift。

detect-stack-set-drift

以下代码示例演示了如何使用 detect-stack-set-drift。

Amazon CLI

检测堆栈集及其所有关联堆栈实例的偏离

以下 detect-stack-set-drift 示例在指定的堆栈集（包括与该堆栈集关联的所有堆栈实例）上启动偏离检测操作，并返回一个可用于跟踪偏离操作状态的操作 ID。

```
aws cloudformation detect-stack-set-drift \
    --stack-set-name stack-set-drift-example
```

输出：

```
{
    "OperationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的检测堆栈集中的非托管配置更改。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectStackSetDrift。

estimate-template-cost

以下代码示例演示了如何使用 estimate-template-cost。

Amazon CLI

估算模板成本

以下 estimate-template-cost 示例为当前文件夹中名为 template.yaml 的模板生成成本估算。

```
aws cloudformation estimate-template-cost \
    --template-body file://template.yaml
```

输出：

```
{
    "Url": "http://calculator.s3.amazonaws.com/calc5.html?
key=cloudformation/7870825a-xmpl-4def-92e7-c4f8dd360cca"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EstimateTemplateCost。

execute-change-set

以下代码示例演示了如何使用 execute-change-set。

Amazon CLI

执行更改集

以下 execute-change-set 示例执行由更改集名称和堆栈名称指定的更改集。

```
aws cloudformation execute-change-set \
    --change-set-name my-change-set \
    --stack-name my-stack
```

以下 execute-change-set 示例执行由更改集的完整 ARN 指定的更改集。

```
aws cloudformation execute-change-set \
    --change-set-name arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-
change-set/bc9555ba-a949-xmpl-bfb8-f41d04ec5784
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExecuteChangeSet。

execute-stack-refactor

以下代码示例演示了如何使用 execute-stack-refactor。

Amazon CLI

完成堆栈重构操作

以下 execute-stack-refactor 示例完成具有所指定堆栈重构 ID 的堆栈重构操作。

```
aws cloudformation execute-stack-refactor \
    --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的堆栈重构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExecuteStackRefactor。

get-stack-policy

以下代码示例演示了如何使用 get-stack-policy。

Amazon CLI

查看堆栈策略

以下 get-stack-policy 示例显示指定堆栈的堆栈策略。使用 set-stack-policy 命令将策略附加到堆栈。

```
aws cloudformation get-stack-policy \
    --stack-name my-stack
```

输出：

```
{
    "StackPolicyBody": "{\n  \"Statement\" : [\n    {\n      \"Effect\" :
 \"Allow\",\n      \"Action\" : \"Update:*\",\n      \"Principal\": \"*\",\n
     \"Resource\" : \"*\"\n    },\n    {\n      \"Effect\" : \"Deny\",\n
 \"Action\" : \"Update:*\",\n      \"Principal\": \"*\",\n      \"Resource\" :
 \"LogicalResourceId/bucket\"\n    }\n  ]\n}\n"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStackPolicy。

get-template-summary

以下代码示例演示了如何使用 get-template-summary。

Amazon CLI

显示模板摘要

以下命令显示有关指定模板文件的资源和元数据的摘要信息。

```
aws cloudformation get-template-summary \
   --template-body file://template.yaml
```

输出：

```
{
    "Parameters": [],
    "Description": "A VPC and subnets.",
    "ResourceTypes": [
        "AWS::EC2::VPC",
        "AWS::EC2::Subnet",
        "AWS::EC2::Subnet",
        "AWS::EC2::RouteTable",
        "AWS::EC2::VPCEndpoint",
        "AWS::EC2::SubnetRouteTableAssociation",
        "AWS::EC2::SubnetRouteTableAssociation",
        "AWS::EC2::VPCEndpoint"
    ],
    "Version": "2010-09-09"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTemplateSummary。

get-template

以下代码示例演示了如何使用 get-template。

Amazon CLI

查看 Amazon CloudFormation 堆栈的模板正文

以下 get-template 命令显示 myteststack 堆栈的模板：

```
aws cloudformation get-template --stack-name myteststack
```

输出：

```
{
    "TemplateBody": {
        "AWSTemplateFormatVersion": "2010-09-09",
        "Outputs": {
            "BucketName": {
                "Description": "Name of S3 bucket to hold website content",
                "Value": {
                    "Ref": "S3Bucket"
                }
            }
        },
        "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample
 template showing how to create a publicly accessible S3 bucket. **WARNING** This
 template creates an S3 bucket. You will be billed for the AWS resources used if you
 create a stack from this template.",
        "Resources": {
            "S3Bucket": {
                "Type": "AWS::S3::Bucket",
                "Properties": {
                    "AccessControl": "PublicRead"
                }
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTemplate。

list-change-sets

以下代码示例演示了如何使用 list-change-sets。

Amazon CLI

列出更改集

以下 list-change-sets 示例显示指定堆栈的待处理更改集列表。

```
aws cloudformation list-change-sets \
   --stack-name my-stack
```

输出：

```
{
    "Summaries": [
        {
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
            "StackName": "my-stack",
            "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/
my-change-set/70160340-7914-xmpl-bcbf-128a1fa78b5d",
            "ChangeSetName": "my-change-set",
            "ExecutionStatus": "AVAILABLE",
            "Status": "CREATE_COMPLETE",
            "CreationTime": "2019-10-02T05:38:54.297Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChangeSets。

list-exports

以下代码示例演示了如何使用 list-exports。

Amazon CLI

列出导出

以下 list-exports 示例显示当前区域堆栈的导出列表。

```
aws cloudformation list-exports
```

输出：

```
{
    "Exports": [
        {
            "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
            "Name": "private-vpc-subnet-a",
            "Value": "subnet-07b410xmplddcfa03"
```

```
        },
        {
            "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
            "Name": "private-vpc-subnet-b",
            "Value": "subnet-075ed3xmplebd2fb1"
        },
        {
            "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
            "Name": "private-vpc-vpcid",
            "Value": "vpc-011d7xmpl100e9841"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListExports。

`list-generated-templates`

以下代码示例演示了如何使用 `list-generated-templates`。

Amazon CLI

列出生成的模板

以下 `list-generated-templates` 示例列出所有生成的模板。

```
aws cloudformation list-generated-templates
```

输出：

```
{
    "Summaries": [
        {
            "GeneratedTemplateId": "arn:aws:cloudformation:us-
east-1:123456789012:generatedtemplate/7fc8512c-d8cb-4e02-b266-d39c48344e48",
            "GeneratedTemplateName": "MyTemplate",
            "Status": "COMPLETE",
            "StatusReason": "All resources complete",
            "CreationTime": "2025-09-23T20:13:24.283000+00:00",
            "LastUpdatedTime": "2025-09-23T20:13:28.610000+00:00",
```

```
            "NumberOfResources": 4
        },
        {
            "GeneratedTemplateId": "arn:aws:cloudformation:us-
east-1:123456789012:generatedTemplate/f10dd1c4-edc6-4823-8153-ab6112b8d051",
            "GeneratedTemplateName": "MyEC2InstanceTemplate",
            "Status": "COMPLETE",
            "StatusReason": "All resources complete",
            "CreationTime": "2024-08-08T19:35:49.790000+00:00",
            "LastUpdatedTime": "2024-08-08T19:35:52.207000+00:00",
            "NumberOfResources": 3
        },
        {
            "GeneratedTemplateId": "arn:aws:cloudformation:us-
east-1:123456789012:generatedTemplate/e5a1c89f-7ce2-41bd-9bdf-75b7c852e3ca",
            "GeneratedTemplateName": "MyEKSNodeGroupTemplate",
            "Status": "COMPLETE",
            "StatusReason": "All resources complete",
            "CreationTime": "2024-07-16T20:39:27.883000+00:00",
            "LastUpdatedTime": "2024-07-16T20:39:35.766000+00:00",
            "NumberOfResources": 4
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的根据现有资源生成模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGeneratedTemplates。

`list-imports`

以下代码示例演示了如何使用 `list-imports`。

Amazon CLI

列出导入

以下 `list-imports` 示例列出导入指定导出的堆栈。使用 `list-exports` 命令获取可用的导出列表。

```
aws cloudformation list-imports \
    --export-name private-vpc-vpcid
```

输出：

```
{
    "Imports": [
        "my-database-stack"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImports。

list-resource-scan-related-resources

以下代码示例演示了如何使用 list-resource-scan-related-resources。

Amazon CLI

列出资源扫描中的相关资源

以下 list-resource-scan-related-resources 示例列出所指定资源扫描中与 resources.json 内资源相关的资源。

```
aws cloudformation list-resource-scan-related-resources \
    --resource-scan-id arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60 \
    --resources file://resources.json
```

的内容resources.json：

```
[
    {
        "ResourceType": "AWS::EKS::Cluster",
        "ResourceIdentifier": {
            "ClusterName": "MyAppClusterName"
        }
    },
    {
        "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
        "ResourceIdentifier": {
            "AutoScalingGroupName": "MyAppASGName"
        }
    }
```

```
]
```

输出：

```
{
    "RelatedResources": [
        {
            "ResourceType": "AWS::EKS::Nodegroup",
            "ResourceIdentifier": {
                "NodegroupName": "MyAppNodegroupName"
            },
            "ManagedByStack": false
        },
        {
            "ResourceType": "AWS::IAM::Role",
            "ResourceIdentifier": {
                "RoleId": "arn:aws::iam::123456789012:role/MyAppIAMRole"
            },
            "ManagedByStack": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 IaC 生成器根据扫描的资源创建 CloudFormation 模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceScanRelatedResources。

`list-resource-scan-resources`

以下代码示例演示了如何使用 `list-resource-scan-resources`。

Amazon CLI

列出资源扫描中的资源

以下 `list-resource-scan-resources` 示例列出所指定资源扫描中的资源，并按资源标识符进行筛选。

```
aws cloudformation list-resource-scan-resources \
    --resource-scan-id arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60 \
```

```
    --resource-identifier MyApp
```

输出：

```
{
    "Resources": [
        {
            "ResourceType": "AWS::EKS::Cluster",
            "ResourceIdentifier": {
                "ClusterName": "MyAppClusterName"
            },
            "ManagedByStack": false
        },
        {
            "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
            "ResourceIdentifier": {
                "AutoScalingGroupName": "MyAppASGName"
            },
            "ManagedByStack": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 IaC 生成器根据扫描的资源创建 CloudFormation 模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceScanResources。

list-stack-instances

以下代码示例演示了如何使用 list-stack-instances。

Amazon CLI

列出堆栈的实例

以下 list-stack-instances 示例列出根据指定堆栈集创建的实例。

```
aws cloudformation list-stack-instances \
    --stack-set-name enable-config
```

示例输出包括有关由于错误而无法更新的堆栈的详细信息：

```
{
    "Summaries": [
        {
            "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
            "Region": "us-west-2",
            "Account": "123456789012",
            "StackId": "arn:aws:cloudformation:ap-northeast-1:123456789012:stack/
StackSet-enable-config-35a6ac50-d9f8-4084-86e4-7da34d5de4c4/a1631cd0-e5fb-xmpl-
b474-0aa20f14f06e",
            "Status": "CURRENT"
        },
        {
            "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
            "Region": "us-west-2",
            "Account": "123456789012",
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/
StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532/eab53680-e5fa-xmpl-
ba14-0a522351f81e",
            "Status": "OUTDATED",
            "StatusReason": "ResourceLogicalId:ConfigDeliveryChannel,
 ResourceType:AWS::Config::DeliveryChannel, ResourceStatusReason:Failed to put
 delivery channel 'StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532-
ConfigDeliveryChannel-1OJWJ7XD59WR0' because the maximum number of delivery
 channels: 1 is reached. (Service: AmazonConfig; Status Code: 400; Error Code:
 MaxNumberOfDeliveryChannelsExceededException; Request ID: d14b34a0-ef7c-xmpl-
acf8-8a864370ae56)."
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackInstances。

list-stack-refactor-actions

以下代码示例演示了如何使用 `list-stack-refactor-actions`。

Amazon CLI

列出堆栈重构操作的详细操作

以下 `list-stack-refactor-actions` 示例列出具有所指定堆栈重构 ID 的堆栈重构操作的详细操作。

```
aws cloudformation list-stack-refactor-actions \
    --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

输出：

```
{
    "StackRefactorActions": [
        {
            "Action": "MOVE",
            "Entity": "RESOURCE",
            "PhysicalResourceId": "MyTestLambdaRole",
            "Description": "No configuration changes detected.",
            "Detection": "AUTO",
            "TagResources": [],
            "UntagResources": [],
            "ResourceMapping": {
                "Source": {
                    "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
                    "LogicalResourceId": "MyLambdaRole"
                },
                "Destination": {
                    "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b",
                    "LogicalResourceId": "MyLambdaRole"
                }
            }
        },
        {
            "Action": "MOVE",
            "Entity": "RESOURCE",
            "PhysicalResourceId": "MyTestFunction",
            "Description": "Resource configuration changes will be validated during
 refactor execution.",
            "Detection": "AUTO",
            "TagResources": [
                {
                    "Key": "aws:cloudformation:stack-name",
                    "Value": "Stack2"
                },
                {
                    "Key": "aws:cloudformation:logical-id",
                    "Value": "MyFunction"
```

```
                },
                {
                    "Key": "aws:cloudformation:stack-id",
                    "Value": "arn:aws:cloudformation:us-east-1:123456789012:stack/
Stack2/5da91650-94b1-11f0-81cf-0a23500e151b"
                }
            ],
            "UntagResources": [
                "aws:cloudformation:stack-name",
                "aws:cloudformation:logical-id",
                "aws:cloudformation:stack-id"
            ],
            "ResourceMapping": {
                "Source": {
                    "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
                    "LogicalResourceId": "MyFunction"
                },
                "Destination": {
                    "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b",
                    "LogicalResourceId": "MyFunction"
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的堆栈重构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackRefactorActions。

list-stack-resources

以下代码示例演示了如何使用 list-stack-resources。

Amazon CLI

列出堆栈中的资源

以下命令显示指定堆栈中的资源列表。

```
aws cloudformation list-stack-resources \
```

```
    --stack-name my-stack
```

输出：

```
{
    "StackResourceSummaries": [
        {
            "LogicalResourceId": "bucket",
            "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",
            "ResourceType": "AWS::S3::Bucket",
            "LastUpdatedTimestamp": "2019-10-02T04:34:11.345Z",
            "ResourceStatus": "CREATE_COMPLETE",
            "DriftInformation": {
                "StackResourceDriftStatus": "IN_SYNC"
            }
        },
        {
            "LogicalResourceId": "function",
            "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
            "ResourceType": "AWS::Lambda::Function",
            "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",
            "ResourceStatus": "UPDATE_COMPLETE",
            "DriftInformation": {
                "StackResourceDriftStatus": "IN_SYNC"
            }
        },
        {
            "LogicalResourceId": "functionRole",
            "PhysicalResourceId": "my-functionRole-HIZXMPLEOM9E",
            "ResourceType": "AWS::IAM::Role",
            "LastUpdatedTimestamp": "2019-10-02T04:34:06.350Z",
            "ResourceStatus": "CREATE_COMPLETE",
            "DriftInformation": {
                "StackResourceDriftStatus": "IN_SYNC"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackResources。

list-stack-set-operation-results

以下代码示例演示了如何使用 list-stack-set-operation-results。

Amazon CLI

列出堆栈集操作结果

以下命令显示对指定堆栈集中的实例执行更新操作的结果。

```
aws cloudformation list-stack-set-operation-results \
    --stack-set-name enable-config \
    --operation-id 35d45ebc-ed88-xmpl-ab59-0197a1fc83a0
```

输出：

```
{
    "Summaries": [
        {
            "Account": "223456789012",
            "Region": "us-west-2",
            "Status": "SUCCEEDED",
            "AccountGateResult": {
                "Status": "SKIPPED",
                "StatusReason": "Function not found: arn:aws:lambda:eu-
west-1:223456789012:function:AWSCloudFormationStackSetAccountGate"
            }
        },
        {
            "Account": "223456789012",
            "Region": "ap-south-1",
            "Status": "CANCELLED",
            "StatusReason": "Cancelled since failure tolerance has exceeded"
        }
    ]
}
```

注意：除非您创建账户门禁函数，否则 AccountGateResult 的 SKIPPED 状态有望成功操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackSetOperationResults。

list-stack-set-operations

以下代码示例演示了如何使用 list-stack-set-operations。

Amazon CLI

列出堆栈集操作

以下 list-stack-set-operations 示例显示指定堆栈集的最新操作列表。

```
aws cloudformation list-stack-set-operations \
    --stack-set-name my-stack-set
```

输出：

```
{
    "Summaries": [
        {
            "OperationId": "35d45ebc-ed88-xmpl-ab59-0197a1fc83a0",
            "Action": "UPDATE",
            "Status": "SUCCEEDED",
            "CreationTimestamp": "2019-10-03T16:28:44.377Z",
            "EndTimestamp": "2019-10-03T16:42:08.607Z"
        },
        {
            "OperationId": "891aa98f-7118-xmpl-00b2-00954d1dd0d6",
            "Action": "UPDATE",
            "Status": "FAILED",
            "CreationTimestamp": "2019-10-03T15:43:53.916Z",
            "EndTimestamp": "2019-10-03T15:45:58.925Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackSetOperations。

list-stack-sets

以下代码示例演示了如何使用 list-stack-sets。

Amazon CLI

列出堆栈集

以下 list-stack-sets 示例显示当前区域和账户中的堆栈集列表。

```
aws cloudformation list-stack-sets
```

输出：

```
{
    "Summaries": [
        {
            "StackSetName": "enable-config",
            "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
            "Description": "Enable AWS Config",
            "Status": "ACTIVE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStackSets。

list-stacks

以下代码示例演示了如何使用 list-stacks。

Amazon CLI

列出 Amazon CloudFormation 堆栈

以下 list-stacks 命令显示具有 CREATE_COMPLETE 状态的所有堆栈的摘要信息：

```
aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE
```

输出：

```
[
    {
        "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
        "TemplateDescription": "AWS CloudFormation Sample Template S3_Bucket: Sample
 template showing how to create a publicly accessible S3 bucket. **WARNING** This
 template creates an S3 bucket. You will be billed for the AWS resources used if you
 create a stack from this template.",
```

```
        "StackStatusReason": null,
        "CreationTime": "2013-08-26T03:27:10.190Z",
        "StackName": "myteststack",
        "StackStatus": "CREATE_COMPLETE"
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStacks。

list-type-registrations

以下代码示例演示了如何使用 list-type-registrations。

Amazon CLI

列出某一类型已完成的注册

以下 list-type-registrations 示例显示指定类型的已完成类型注册的列表。

```
aws cloudformation list-type-registrations \
    --type RESOURCE \
    --type-name My::Logs::LogGroup \
    --registration-status-filter COMPLETE
```

输出：

```
{
    "RegistrationTokenList": [
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTypeRegistrations。

list-type-versions

以下代码示例演示了如何使用 list-type-versions。

Amazon CLI

列出扩展的版本

以下 `list-type-versions` 示例返回有关扩展版本的摘要信息。

```
aws cloudformation list-type-versions \
  --endpoint https://example.com \
  --region us-west-2 \
  --type RESOURCE \
  --type-name My::Resource::Example \
  --publisher-id 123456789012
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTypeVersions。

list-types

以下代码示例演示了如何使用 `list-types`。

Amazon CLI

列出账户中的私有资源类型

以下 `list-types` 示例显示当前在当前 Amazon 账户中注册的私有资源类型的列表。

```
aws cloudformation list-types
```

输出：

```
{
    "TypeSummaries": [
        {
            "Description": "WordPress blog resource for internal use",
            "LastUpdated": "2019-12-04T18:28:15.059Z",
            "TypeName": "My::WordPress::BlogExample",
            "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/
My-WordPress-BlogExample",
```

```
            "DefaultVersionId": "00000005",
            "Type": "RESOURCE"
        },
        {
            "Description": "Customized resource derived from AWS::Logs::LogGroup",
            "LastUpdated": "2019-12-04T18:28:15.059Z",
            "TypeName": "My::Logs::LogGroup",
            "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/
My-Logs-LogGroup",
            "DefaultVersionId": "00000003",
            "Type": "RESOURCE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTypes。

package

以下代码示例演示了如何使用 package。

Amazon CLI

以下命令将通过将本地构件上传到 S3 存储桶 bucket-name 来导出一个名为 template.json 的模板，并将导出的模板写入 packaged-template.json：

```
aws cloudformation package --template-file /path_to_template/template.json --s3-
bucket bucket-name --output-template-file packaged-template.json --use-json
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Package。

publish-type

以下代码示例演示了如何使用 publish-type。

Amazon CLI

发布扩展

以下 publish-type 示例会将指定扩展作为此区域中的公有扩展发布到 CloudFormation 注册表。

```
aws cloudformation publish-type \
  --region us-west-2 \
  --type RESOURCE \
  --type-name Example::Test::1234567890abcdef0
```

输出：

```
{
    "PublicTypeArn":"arn:aws:cloudformation:us-west-2::type/
resource/000q6TfUovXsEMmgKowxDZLlwqr2QUshd2e75c8c/Example-
Test-1234567890abcdef0/1.0.0"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PublishType。

register-publisher

以下代码示例演示了如何使用 register-publisher。

Amazon CLI

注册发布者

以下 register-publisher 示例注册发布者并接受条款和条件参数。

```
aws cloudformation register-publisher \
  --region us-west-2 \
  --accept-terms-and-conditions
```

输出：

```
{
    "PublisherId": "000q6TfUovXsEMmgKowxDZLlwqr2QUshd2e75c8c"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterPublisher。

register-type

以下代码示例演示了如何使用 register-type。

Amazon CLI

注册资源类型

以下 register-type 示例会将指定的资源类型注册为用户账户中的私有资源类型。

```
aws cloudformation register-type \
    --type-name My::Organization::ResourceName \
    --schema-handler-package s3://bucket_name/my-organization-resource_name.zip \
    --type RESOURCE
```

输出：

```
{
    "RegistrationToken": "f5525280-104e-4d35-bef5-8f1f1example"
}
```

有关更多信息，请参阅《CloudFormation 命令行界面类型开发用户指南》中的注册资源提供方。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterType。

set-stack-policy

以下代码示例演示了如何使用 set-stack-policy。

Amazon CLI

应用堆栈策略

以下 set-stack-policy 示例禁用指定堆栈中指定的资源更新。stack-policy.json 是一个 JSON 文档，可定义允许对堆栈中的资源执行的操作。

```
aws cloudformation set-stack-policy \
    --stack-name my-stack \
    --stack-policy-body file://stack-policy.json
```

输出：

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    },
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/bucket"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetStackPolicy。

set-type-configuration

以下代码示例演示了如何使用 set-type-configuration。

Amazon CLI

配置数据

以下 set-type-configuration 示例指定给定账户和区域中已注册的 CloudFormation 扩展的配置数据。

```
aws cloudformation set-type-configuration \
    --region us-west-2 \
    --type RESOURCE \
    --type-name Example::Test::Type \
    --configuration-alias default \
    --configuration "{\"CredentialKey\": \"testUserCredential\"}"
```

输出：

```
{
```

```
    "ConfigurationArn": "arn:aws:cloudformation:us-west-2:123456789012:type-
configuration/resource/Example-Test-Type/default"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetTypeConfiguration。

set-type-default-version

以下代码示例演示了如何使用 set-type-default-version。

Amazon CLI

设置类型的默认版本

以下 set-type-default-version 示例会将指定的类型版本设置为该类型的默认版本。

```
aws cloudformation set-type-default-version \
    --type RESOURCE \
    --type-name My::Logs::LogGroup \
    --version-id 00000003
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetTypeDefaultVersion。

signal-resource

以下代码示例演示了如何使用 signal-resource。

Amazon CLI

发出资源信号

以下 signal-resource 示例发出信号 success，表示在名为 my-stack 的堆栈中满足名为 MyWaitCondition 的等待条件。

```
aws cloudformation signal-resource \
```

```
    --stack-name my-stack \
    --logical-resource-id MyWaitCondition \
    --unique-id 1234 \
    --status SUCCESS
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SignalResource。

start-resource-scan

以下代码示例演示了如何使用 start-resource-scan。

Amazon CLI

启动资源扫描

以下 start-resource-scan 示例启动一个资源扫描，这会扫描当前账户和区域中的所有现有资源。

```
aws cloudformation start-resource-scan
```

输出：

```
{
    "ResourceScanId":
      "arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 CloudFormation IaC 生成器启动资源扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartResourceScan。

stop-stack-set-operation

以下代码示例演示了如何使用 stop-stack-set-operation。

Amazon CLI

停止堆栈集操作

以下 stop-stack-set-operation 示例停止正在对指定堆栈集进行的更新操作。

```
aws cloudformation stop-stack-set-operation \
    --stack-set-name my-stack-set \
    --operation-id 1261cd27-490b-xmpl-ab42-793a896c69e6
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopStackSetOperation。

test-type

以下代码示例演示了如何使用 test-type。

Amazon CLI

测试扩展

以下 test-type 示例测试已注册的扩展，以确保其符合在 CloudFormation 注册表中发布的所有必要要求。

```
aws cloudformation test-type \
    --arn arn:aws:cloudformation:us-west-2:123456789012:type/resource/Sample-Test-Resource123/00000001
```

输出：

```
{
    "TypeVersionArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/Sample-Test-Resource123/00000001"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestType。

update-stack-instances

以下代码示例演示了如何使用 update-stack-instances。

Amazon CLI

更新堆栈实例

以下 update-stack-instances 示例使用最新设置重试两个区域中两个账户的堆栈实例的更新。指定的容错设置可确保在所有账户和地区尝试更新，即使某些堆栈无法更新。

```
aws cloudformation update-stack-instances \
    --stack-set-name my-stack-set \
    --accounts 123456789012 567890123456 \
    --regions us-east-1 us-west-2 \
    --operation-preferences FailureToleranceCount=3
```

输出：

```
{
    "OperationId": "103ebdf2-21ea-xmpl-8892-de5e30733132"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStackInstances。

update-stack-set

以下代码示例演示了如何使用 update-stack-set。

Amazon CLI

更新堆栈集

以下 update-stack-set 示例为指定堆栈集中的堆栈实例添加键名称为 Owner 和值为 IT 的标签。

```
aws cloudformation update-stack-set \
    --stack-set-name my-stack-set \
    --use-previous-template \
    --tags Key=Owner,Value=IT
```

输出：

```
{
    "OperationId": "e2b60321-6cab-xmpl-bde7-530c6f47950e"
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStackSet。

update-stack

以下代码示例演示了如何使用 update-stack。

Amazon CLI

更新 Amazon CloudFormation 堆栈

以下 update-stack 命令更新 mystack 堆栈的模板和输入参数：

```
aws cloudformation update-stack --stack-name mystack --
template-url https://s3.amazonaws.com/sample/updated.template --
parameters ParameterKey=KeyPairName,ParameterValue=SampleKeyPair
 ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,SampleSubnetID2
```

以下 update-stack 命令更新 mystack 堆栈的 SubnetIDs 参数值：如果您没有指定参数值，则将使用模板中指定的默认值：

```
aws cloudformation update-stack --stack-name mystack --
template-url https://s3.amazonaws.com/sample/updated.template
 --parameters ParameterKey=KeyPairName,UsePreviousValue=true
 ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,UpdatedSampleSubnetID2
```

以下 update-stack 命令向 mystack 堆栈添加两个堆栈通知主题：

```
aws cloudformation update-stack --stack-name mystack --use-previous-template --
notification-arns "arn:aws:sns:use-east-1:123456789012:mytopic1" "arn:aws:sns:us-
east-1:123456789012:mytopic2"
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 Amazon 堆栈更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStack。

update-termination-protection

以下代码示例演示了如何使用 update-termination-protection。

Amazon CLI

启用终止保护

以下 update-termination-protection 示例在指定堆栈上启用终止保护。

```
aws cloudformation update-termination-protection \
    --stack-name my-stack \
    --enable-termination-protection
```

输出：

```
{
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTerminationProtection。

validate-template

以下代码示例演示了如何使用 validate-template。

Amazon CLI

验证 Amazon CloudFormation 模板

以下 validate-template 命令验证 sampletemplate.json 模板：

```
aws cloudformation validate-template --template-body file://sampletemplate.json
```

输出：

```
{
    "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template
 showing how to create a publicly accessible S3 bucket. **WARNING** This template
 creates an S3 bucket. You will be billed for the AWS resources used if you create a
 stack from this template.",
    "Parameters": [],
    "Capabilities": []
```

```
}
```

有关更多信息，请参阅《Amazon CloudTrail 用户指南》中的“使用 Amazon CloudFormation 模板”。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateTemplate。

使用 Amazon CLI 的 CloudFront 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CloudFront 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-distribution-tenant-web-acl`

以下代码示例演示了如何使用 `associate-distribution-tenant-web-acl`。

Amazon CLI

将 Web ACL 与 CloudFront 分配租户关联

以下 `associate-distribution-tenant-web-acl` 示例将 Web ACL 与具有 ETag E13V1IB3VIYABC 的 CloudFront 分配关联。

```
aws cloudfront associate-distribution-tenant-web-acl \
    --id dt_2wjDZi3hD1iv0Xf6rpZJO1AB \
    --if-match E13V1IB3VIYABC \
    --web-acl-arn arn:aws:wafv2:us-east-1:123456789012:global/webacl/web-global-example/626900da-5f64-418b-ba9b-743f37123ABC
```

输出：

```
{
    "ETag": "E1VC38T7YXBABC",
    "Id": "dt_2wjDZi3hD1ivOXf6rpZJO1AB",
    "WebACLArn": "arn:aws:wafv2:us-east-1:123456789012:global/webacl/web-global-
example/626900da-5f64-418b-ba9b-743f37123ABC"
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用 Amazon WAF 保护功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateDistributionTenantWebAcl。

`associate-distribution-web-acl`

以下代码示例演示了如何使用 `associate-distribution-web-acl`。

Amazon CLI

将 Web ACL 与 CloudFront 分配关联

以下 `associate-distribution-web-acl` 示例将 Web ACL 与 CloudFront 分配关联。

```
aws cloudfront associate-distribution-web-acl \
    --id E1XNX8R2GOAABC \
    --if-match E2YWS1C2J3OABC \
    --web-acl-arn arn:aws:wafv2:us-east-1:123456789012:global/webacl/web-global-
example/626900da-5f64-418b-ba9b-743f3746cABC
```

输出：

```
{
    "ETag": "E3QE7ED60U0ABC",
    "Id": "E1XNX8R2GOAABC",
    "WebACLArn": "arn:aws:wafv2:us-east-1:123456789012:global/webacl/web-global-
example/626900da-5f64-418b-ba9b-743f3746cABC"
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用 Amazon WAF 保护功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateDistributionWebAcl。

create-cloud-front-origin-access-identity

以下代码示例演示了如何使用 create-cloud-front-origin-access-identity。

Amazon CLI

创建 CloudFront 来源访问身份

以下示例通过将 OAI 配置作为命令行参数来创建 CloudFront 来源访问身份（OAI）：

```
aws cloudfront create-cloud-front-origin-access-identity \
    --cloud-front-origin-access-identity-config \
        CallerReference="cli-example",Comment="Example OAI"
```

您可以通过在 JSON 文件中提供 OAI 配置来完成同样的事情，如以下示例所示：

```
aws cloudfront create-cloud-front-origin-access-identity \
    --cloud-front-origin-access-identity-config file://OAI-config.json
```

文件 OAI-config.json 是当前目录中包含以下内容的 JSON 文档：

```
{
    "CallerReference": "cli-example",
    "Comment": "Example OAI"
}
```

无论您使用命令行参数还是 JSON 文件提供 OAI 配置，输出都相同：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/origin-access-identity/
cloudfront/E74FTE3AEXAMPLE",
    "ETag": "E2QWRUHEXAMPLE",
    "CloudFrontOriginAccessIdentity": {
        "Id": "E74FTE3AEXAMPLE",
        "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
        "CloudFrontOriginAccessIdentityConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example OAI"
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCloudFrontOriginAccessIdentity。

create-connection-group

以下代码示例演示了如何使用 create-connection-group。

Amazon CLI

在 CloudFront 中创建连接组

以下 create-connection-group 示例创建一个启用的连接组，指定任播静态 IP 列表，并禁用 IPv6。

```
aws cloudfront create-connection-group \
    --name cg-with-anycast-ip-list \
    --no-ipv6-enabled \
    --enabled \
    --anycast-ip-list-id aip_CCkW6gKrDiBD4n78123ABC \
    --tags "Items=[{Key=abc,Value=123}]"
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "ConnectionGroup": {
        "Id": "cg_2yb6uj74B4PCbfhT31WFdiSABC",
        "Name": "cg-with-anycast-ip-list",
        "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2yb6uj74B4PCbfhT31WFdiSABC",
        "CreatedTime": "2025-06-16T16:25:50.061000+00:00",
        "LastModifiedTime": "2025-06-16T16:25:50.061000+00:00",
        "Tags": {
            "Items": [
                {
                    "Key": "abc",
                    "Value": "123"
                }
            ]
        },
```

```
        "Ipv6Enabled": false,
        "RoutingEndpoint": "dj6xusxq65abc.cloudfront.net",
        "AnycastIpListId": "aip_CCkW6gKrDiBD4n78123ABC",
        "Status": "InProgress",
        "Enabled": true,
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateConnectionGroup。

`create-distribution-tenant`

以下代码示例演示了如何使用 `create-distribution-tenant`。

Amazon CLI

示例 1：创建使用自定义证书的 CloudFront 分配租户

以下 `create-distribution-tenant` 示例创建一个 CloudFront 分配租户，该租户指定用于禁用 WAF、添加地理限制和使用其它 TLS 证书的自定义项。

```
aws cloudfront create-distribution-tenant \
    --cli-input-json file://tenant.json
```

的内容`tenant.json`：

```
{
    "DistributionId": "E1XNX8R2GOAABC",
    "Domains": [
        {
            "Domain": "example.com"
        }
    ],
    "Parameters": [
        {
            "Name": "testParam",
            "Value": "defaultValue"
        }
```

```
    ],
    "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
    "Enabled": false,
    "Tags": {
        "Items": [
            {
                "Key": "tag",
                "Value": "tagValue"
            }
        ]
    },
    "Name": "new-tenant-customizations",
    "Customizations": {
        "GeoRestrictions": {
            "Locations": ["DE"],
            "RestrictionType": "whitelist"
        },
        "WebAcl": {
            "Action": "disable"
        },
        "Certificate": {
            "Arn": "arn:aws:acm:us-east-1:123456789012:certificate/ec53f564-
ea5a-4e4a-a0a2-e3c989449abc"
        }
    }
}
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "DistributionTenant": {
        "Id": "dt_2yN5tYwVbPKr7m2IB69M1yp1AB",
        "DistributionId": "E1XNX8R2GOAABC",
        "Name": "new-tenant-customizations",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2yN5tYwVbPKr7m2IB69M1yp1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "active"
            }
        ],
```

```
        "Tags": {
            "Items": [
                {
                    "Key": "tag",
                    "Value": "tagValue"
                }
            ]
        },
        "Customizations": {
            "WebAcl": {
                "Action": "disable"
            },
            "Certificate": {
                "Arn": "arn:aws:acm:us-east-1:123456789012:certificate/ec53f564-
ea5a-4e4a-a0a2-e3c989449abc"
            },
            "GeoRestrictions": {
                "RestrictionType": "whitelist",
                "Locations": [
                    "DE"
                ]
            }
        },
        "Parameters": [
            {
                "Name": "testParam",
                "Value": "defaultValue"
            }
        ],
        "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
        "CreatedTime": "2025-06-11T17:20:06.432000+00:00",
        "LastModifiedTime": "2025-06-11T17:20:06.432000+00:00",
        "Enabled": false,
        "Status": "InProgress"
    }
}
```

示例 2：使用继承的证书创建分配租户

以下 create-distribution-tenant 示例创建一个分配租户，并指定从多租户分配继承的 TLS 证书。

```
aws cloudfront create-distribution-tenant \
```

```
    --cli-input-json file://tenant.json
```

的内容tenant.json：

```
{
    "DistributionId": "E1HVIAU7U12ABC",
    "Domains": [
        {
            "Domain": "example.com"
        }
    ],
    "Parameters": [
        {
            "Name": "tenantName",
            "Value": "first-tenant"
        }
    ],
    "Enabled": true,
    "Name": "new-tenant-no-cert"
}
```

输出：

```
{
    "ETag": "E23ZP02F0ABC",
    "DistributionTenant": {
        "Id": "dt_2zhRB0vBe0B72LZCVy1mgzI1AB",
        "DistributionId": "E1HVIAU7U12ABC",
        "Name": "new-tenant-no-cert",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2zhRB0vBe0B72LZCVy1mgzI1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "active"
            }
        ],
        "Parameters": [
            {
                "Name": "tenantName",
                "Value": "first-tenant"
            }
        ],
```

```
        "ConnectionGroupId": "cg_2yQEwpipGFN0hhA0ZemPabOABC",
        "CreatedTime": "2025-07-10T20:59:38.414000+00:00",
        "LastModifiedTime": "2025-07-10T20:59:38.414000+00:00",
        "Enabled": true,
        "Status": "InProgress"
    }
}
```

示例 3：使用 CloudFront 托管的验证令牌创建 CloudFront 分配租户

以下 `create-distribution-tenant` 示例创建一个分配租户，并为域名使用 CloudFront 托管的验证令牌。

```
aws cloudfront create-distribution-tenant \
    --cli-input-json file://tenant.json
```

的内容`tenant.json`：

```
{
    "DistributionId": "E2GJ5J9QN12ABC",
    "Domains": [
        {
            "Domain": "example.com"
        }
    ],
    "Parameters": [
        {
            "Name": "tenantName",
            "Value": "first-tenant"
        }
    ],
    "ConnectionGroupId": "cg_2yQEwpipGFN0hhA0ZemPabOABC",
    "Enabled": true,
    "Name": "new-tenant-cf-hosted",
    "ManagedCertificateRequest": {
        "ValidationTokenHost": "cloudfront"
    }
}
```

重要：要成功运行此命令，必须配置 CNAME DNS 记录，该记录将新域（example.com）指向与分配租户关联的连接组的路由端点。在 CloudFront 成功完成此请求之前，还必须传播此 CNAME 记录。

输出：

```
{
    "ETag": "E23ZP02F0ABC",
    "DistributionTenant": {
        "Id": "dt_2zhStKrA524GvvTWJX92Ozl1AB",
        "DistributionId": "E2GJ5J9QN12ABC",
        "Name": "new-tenant-cf-hosted",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2zhStKrA524GvvTWJX92Ozl1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "inactive"
            }
        ],
        "Parameters": [
            {
                "Name": "tenantName",
                "Value": "first-tenant"
            }
        ],
        "ConnectionGroupId": "cg_2zhSaGatwwXjTjE42nneZzqABC",
        "CreatedTime": "2025-07-10T21:13:46.416000+00:00",
        "LastModifiedTime": "2025-07-10T21:13:46.416000+00:00",
        "Enabled": true,
        "Status": "InProgress"
    }
}
```

示例 4：使用自托管的验证令牌创建 CloudFront 分配租户

以下 `create-distribution-tenant` 示例创建一个 CloudFront 分配租户并使用自托管的验证令牌。

```
aws cloudfront create-distribution-tenant \
    --cli-input-json file://tenant.json
```

的内容`tenant.json`：

```
{
    "DistributionId": "E2GJ5J9QN12ABC",
```

```
    "Domains": [
        {
            "Domain": "example.com"
        }
    ],
    "Parameters": [
        {
            "Name": "tenantName",
            "Value": "first-tenant"
        }
    ],
    "Enabled": true,
    "Name": "new-tenant-self-hosted",
    "ManagedCertificateRequest": {
        "ValidationTokenHost": "self-hosted"
    }
}
```

输出：

```
{
    "ETag": "E23ZP02F0ABC",
    "DistributionTenant": {
        "Id": "dt_2zhTFBV93OfFJJ3YMdNM5BC1AB",
        "DistributionId": "E2GJ5J9QN12ABC",
        "Name": "new-tenant-self-hosted",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2zhTFBV93OfFJJ3YMdNM5BC1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "inactive"
            }
        ],
        "Parameters": [
            {
                "Name": "tenantName",
                "Value": "first-tenant"
            }
        ],
        "ConnectionGroupId": "cg_2yQEwpipGFN0hhA0ZemPabOABC",
        "CreatedTime": "2025-07-10T21:16:39.828000+00:00",
        "LastModifiedTime": "2025-07-10T21:16:39.828000+00:00",
```

```
        "Enabled": true,
        "Status": "InProgress"
    }
}
```

重要：运行此命令后，将在不进行验证的情况下创建分配租户。要在准备好开始接收流量时验证托管式证书请求并配置 DNS，请参阅《Amazon CloudFront 开发人员指南》中的完成域设置。

有关创建分配租户的更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建分配。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateDistributionTenant。

create-distribution-with-tags

以下代码示例演示了如何使用 create-distribution-with-tags。

Amazon CLI

创建带有标签的 CloudFront 分配

以下 create-distribution-with-tags 示例通过在名为 dist-config-with-tags.json 的 JSON 文件中提供分配配置和标签来创建带两个标签的分配。

```
aws cloudfront create-distribution-with-tags \
    --distribution-config-with-tags file://dist-config-with-tags.json
```

文件 dist-config-with-tags.json 是当前文件夹中的 JSON 文档。请注意文件顶部的 Tags 对象，其中包含两个标签：

Name = ExampleDistributionProject = ExampleProject

的内容dist-config-with-tags.json：

```
{
    "Tags": {
        "Items": [
            {
                "Key": "Name",
                "Value": "ExampleDistribution"
            },
            {
                "Key": "Project",
```

```
                "Value": "ExampleProject"
            }
        ]
    },
    "DistributionConfig": {
        "CallerReference": "cli-example",
        "Aliases": {
            "Quantity": 0
        },
        "DefaultRootObject": "index.html",
        "Origins": {
            "Quantity": 1,
            "Items": [
                {
                    "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                    "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                    "OriginPath": "",
                    "CustomHeaders": {
                        "Quantity": 0
                    },
                    "S3OriginConfig": {
                        "OriginAccessIdentity": ""
                    }
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                },
                "Headers": {
                    "Quantity": 0
                },
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
```

```
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
            "Prefix": ""
        },
        "PriceClass": "PriceClass_All",
        "Enabled": true,
        "ViewerCertificate": {
```

```
            "CloudFrontDefaultCertificate": true,
            "MinimumProtocolVersion": "TLSv1",
            "CertificateSource": "cloudfront"
        },
        "Restrictions": {
            "GeoRestriction": {
                "RestrictionType": "none",
                "Quantity": 0
            }
        },
        "WebACLId": "",
        "HttpVersion": "http2",
        "IsIPV6Enabled": true
    }
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE",
    "ETag": "E2QWRUHEXAMPLE",
    "Distribution": {
        "Id": "EDFDVBD6EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-12-04T23:35:41.433Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
```

```
                        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "MinTTL": 0,
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
```

```
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
            "Prefix": ""
        },
        "PriceClass": "PriceClass_All",
        "Enabled": true,
        "ViewerCertificate": {
            "CloudFrontDefaultCertificate": true,
            "MinimumProtocolVersion": "TLSv1",
            "CertificateSource": "cloudfront"
        },
        "Restrictions": {
            "GeoRestriction": {
                "RestrictionType": "none",
                "Quantity": 0
            }
        },
        "WebACLId": "",
        "HttpVersion": "http2",
        "IsIPV6Enabled": true
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDistributionWithTags。

create-distribution

以下代码示例演示了如何使用 create-distribution。

Amazon CLI

示例 1：创建 CloudFront 分配

以下 create-distribution 示例使用命令行参数，为名为 amzn-s3-demo-bucket 的 S3 存储桶创建分配，还将 index.html 指定为默认根对象。

```
aws cloudfront create-distribution \
    --origin-domain-name amzn-s3-demo-bucket.s3.amazonaws.com \
    --default-root-object index.html
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EMLARXS9EXAMPLE",
    "ETag": "E9LHASXEXAMPLE",
    "Distribution": {
        "Id": "EMLARXS9EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-11-22T00:55:15.705Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example",
            "Aliases": {
                "Quantity": 0
```

```
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "MinTTL": 0,
                "AllowedMethods": {
                    "Quantity": 2,
```

```
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
```

```
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

示例 2：使用 JSON 文件创建 CloudFront 分配

以下 `create-distribution` 示例使用 JSON 文件，为名为 `amzn-s3-demo-bucket` 的 S3 存储桶创建分配，还将 `index.html` 指定为默认根对象。

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

的内容`dist-config.json`：

```
{
    "CallerReference": "cli-example",
    "Aliases": {
        "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                "OriginPath": "",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
    "OriginGroups": {
```

```
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            },
            "QueryStringCacheKeys": {
                "Quantity": 0
            }
        },
        "TrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ViewerProtocolPolicy": "allow-all",
        "MinTTL": 0,
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        },
        "SmoothStreaming": false,
        "DefaultTTL": 86400,
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
            "Quantity": 0
        },
        "FieldLevelEncryptionId": ""
```

```
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
```

有关输出示例，请参阅示例 1。

示例 3：使用证书创建 CloudFront 多租户分配

以下 `create-distribution` 示例创建一个支持多租户的 CloudFront 分配，并指定 TLS 证书。

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

的内容`dist-config.json`：

```
{
    "CallerReference": "cli-example-with-cert",
    "Comment": "CLI example distribution",
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "OriginPath": "/{{tenantName}}",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
        "ViewerProtocolPolicy": "allow-all",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": ["HEAD", "GET"],
            "CachedMethods": {
                "Quantity": 2,
                "Items": ["HEAD", "GET"]
            }
        }
    },
    "Enabled": true,
    "ViewerCertificate": {
        "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/191306a1-db01-49ca-90ef-fc414ee5dabc",
        "SSLSupportMethod": "sni-only"
    },
    "HttpVersion": "http2",
    "ConnectionMode": "tenant-only",
    "TenantConfig": {
        "ParameterDefinitions": [
```

```
            {
                "Name": "tenantName",
                "Definition": {
                    "StringSchema": {
                        "Comment": "tenantName parameter",
                        "DefaultValue": "root",
                        "Required": false
                    }
                }
            }
        ]
    }
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E1HVIAU7UABC",
    "ETag": "E20LT7R1BABC",
    "Distribution": {
        "Id": "E1HVIAU7U12ABC",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/E1HVIAU7U12ABC",
        "Status": "InProgress",
        "LastModifiedTime": "2025-07-10T20:33:31.117000+00:00",
        "InProgressInvalidationBatches": 0,
        "DomainName": "example.com",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ActiveTrustedKeyGroups": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example-with-cert",
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
```

```
                        "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                        "OriginPath": "/{{tenantName}}",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        },
                        "ConnectionAttempts": 3,
                        "ConnectionTimeout": 10,
                        "OriginShield": {
                            "Enabled": false
                        },
                        "OriginAccessControlId": ""
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "TrustedKeyGroups": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": ["HEAD", "GET"],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": ["HEAD", "GET"]
                    }
                },
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": "",
```

```
                "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
                "GrpcConfig": {
                    "Enabled": false
                }
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "CLI example distribution",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": false,
                "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
                "SSLSupportMethod": "sni-only",
                "MinimumProtocolVersion": "TLSv1.2_2021",
                "Certificate": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
                "CertificateSource": "acm"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "TenantConfig": {
                "ParameterDefinitions": [
                    {
                        "Name": "tenantName",
                        "Definition": {
                            "StringSchema": {
                                "Comment": "tenantName parameter",
```

```
                                "DefaultValue": "root",
                                "Required": false
                            }
                        }
                    }
                ]
            },
            "ConnectionMode": "tenant-only"
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用分配。

示例 4：创建不带证书的 CloudFront 多租户分配

以下 `create-distribution` 示例创建一个支持多租户但没有 TLS 证书的 CloudFront 分配。

```
aws cloudfront create-distribution \
    --distribution-config file://dist-config.json
```

的内容`dist-config.json`：

```
{
    "CallerReference": "cli-example",
    "Comment": "CLI example distribution",
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "OriginPath": "/{{tenantName}}",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
```

```
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
        "ViewerProtocolPolicy": "allow-all",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        }
    },
    "Enabled": true,
    "HttpVersion": "http2",
    "ConnectionMode": "tenant-only",
    "TenantConfig": {
        "ParameterDefinitions": [
            {
                "Name": "tenantName",
                "Definition": {
                    "StringSchema": {
                        "Comment": "tenantName parameter",
                        "DefaultValue": "root",
                        "Required": false
                    }
                }
            }
        ]
    }
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E2GJ5J9QN12ABC",
```

```
    "ETag": "E37YLVVQIABC",
    "Distribution": {
        "Id": "E2GJ5J9QNABC",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/E2GJ5J9QN12ABC",
        "Status": "InProgress",
        "LastModifiedTime": "2025-07-10T20:35:20.565000+00:00",
        "InProgressInvalidationBatches": 0,
        "DomainName": "example.com",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ActiveTrustedKeyGroups": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example-no-cert",
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                        "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                        "OriginPath": "/{{tenantName}}",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        },
                        "ConnectionAttempts": 3,
                        "ConnectionTimeout": 10,
                        "OriginShield": {
                            "Enabled": false
                        },
                        "OriginAccessControlId": ""
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
```

```
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                "TrustedKeyGroups": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": "",
                "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
                "GrpcConfig": {
                    "Enabled": false
                }
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "CLI example distribution",
            "Logging": {
                "Enabled": false,
```

```
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "SSLSupportMethod": "sni-only",
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "TenantConfig": {
                "ParameterDefinitions": [
                    {
                        "Name": "tenantName",
                        "Definition": {
                            "StringSchema": {
                                "Comment": "tenantName parameter",
                                "DefaultValue": "root",
                                "Required": false
                            }
                        }
                    }
                ]
            },
            "ConnectionMode": "tenant-only"
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的配置分配。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDistribution。

create-field-level-encryption-config

以下代码示例演示了如何使用 create-field-level-encryption-config。

Amazon CLI

创建 CloudFront 字段级加密配置

以下示例通过在名为 fle-config.json 的 JSON 文件中提供配置参数来创建字段级加密配置。在创建字段级加密配置之前，必须具有字段级加密配置文件。要创建配置文件，请参阅 create-field-level-encryption-profile 命令。

有关 CloudFront 字段级加密的更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用字段级加密帮助保护敏感数据。

```
aws cloudfront create-field-level-encryption-config \
    --field-level-encryption-config file://fle-config.json
```

文件 fle-config.json 是当前文件夹中包含以下内容的 JSON 文档：

```
{
    "CallerReference": "cli-example",
    "Comment": "Example FLE configuration",
    "QueryArgProfileConfig": {
        "ForwardWhenQueryArgProfileIsUnknown": true,
        "QueryArgProfiles": {
            "Quantity": 0
        }
    },
    "ContentTypeProfileConfig": {
        "ForwardWhenContentTypeIsUnknown": true,
        "ContentTypeProfiles": {
            "Quantity": 1,
            "Items": [
                {
                    "Format": "URLEncoded",
                    "ProfileId": "P280MFCLSYOCVU",
                    "ContentType": "application/x-www-form-urlencoded"
                }
            ]
        }
    }
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/field-level-encryption/
C3KM2WVD605UAY",
    "ETag": "E2P4Z4VU7TY5SG",
    "FieldLevelEncryption": {
        "Id": "C3KM2WVD605UAY",
        "LastModifiedTime": "2019-12-10T21:30:18.974Z",
        "FieldLevelEncryptionConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example FLE configuration",
            "QueryArgProfileConfig": {
                "ForwardWhenQueryArgProfileIsUnknown": true,
                "QueryArgProfiles": {
                    "Quantity": 0,
                    "Items": []
                }
            },
            "ContentTypeProfileConfig": {
                "ForwardWhenContentTypeIsUnknown": true,
                "ContentTypeProfiles": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "Format": "URLEncoded",
                            "ProfileId": "P280MFCLSYOCVU",
                            "ContentType": "application/x-www-form-urlencoded"
                        }
                    ]
                }
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFieldLevelEncryptionConfig。

create-field-level-encryption-profile

以下代码示例演示了如何使用 create-field-level-encryption-profile。

Amazon CLI

创建 CloudFront 字段级加密配置文件

以下示例通过在名为 `fle-profile-config.json` 的 JSON 文件中提供参数来创建字段级加密配置文件。在创建字段级加密配置文件之前，必须具有 CloudFront 公钥。要创建 CloudFront 公钥，请参阅 create-public-key 命令。

有关 CloudFront 字段级加密的更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用字段级加密帮助保护敏感数据。

```
aws cloudfront create-field-level-encryption-profile \
    --field-level-encryption-profile-config file://fle-profile-config.json
```

文件 `fle-profile-config.json` 是当前文件夹中包含以下内容的 JSON 文档：

```
{
    "Name": "ExampleFLEProfile",
    "CallerReference": "cli-example",
    "Comment": "FLE profile for AWS CLI example",
    "EncryptionEntities": {
        "Quantity": 1,
        "Items": [
            {
                "PublicKeyId": "K2K8NC4HVFE3M0",
                "ProviderId": "ExampleFLEProvider",
                "FieldPatterns": {
                    "Quantity": 1,
                    "Items": [
                        "ExampleSensitiveField"
                    ]
                }
            }
        ]
    }
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/field-level-encryption-profile/PPK0UOSIF5WSV",
```

```
    "ETag": "E2QWRUHEXAMPLE",
    "FieldLevelEncryptionProfile": {
        "Id": "PPK0UOSIF5WSV",
        "LastModifiedTime": "2019-12-10T01:03:16.537Z",
        "FieldLevelEncryptionProfileConfig": {
            "Name": "ExampleFLEProfile",
            "CallerReference": "cli-example",
            "Comment": "FLE profile for AWS CLI example",
            "EncryptionEntities": {
                "Quantity": 1,
                "Items": [
                    {
                        "PublicKeyId": "K2K8NC4HVFE3M0",
                        "ProviderId": "ExampleFLEProvider",
                        "FieldPatterns": {
                            "Quantity": 1,
                            "Items": [
                                "ExampleSensitiveField"
                            ]
                        }
                    }
                ]
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFieldLevelEncryptionProfile。

create-invalidation-for-distribution-tenant

以下代码示例演示了如何使用 create-invalidation-for-distribution-tenant。

Amazon CLI

为分配租户创建 CloudFront 失效

以下 create-invalidation-for-distribution-tenant 示例为 CloudFront 分配租户中的所有文件创建失效。

```
aws cloudfront create-invalidation-for-distribution-tenant \
    --id dt_2wjDZi3hD1ivOXf6rpZJO1AB \
```

```
    --invalidation-batch '{"Paths": {"Quantity": 1, "Items": ["/*"]},
 "CallerReference": "invalidation-$(date +%s)"}'
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution-tenant/
dt_2wjDZi3hD1ivOXf6rpZJO1AB/invalidation/I2JGL2F1ZAA426PGG0YLLKABC",
    "Invalidation": {
        "Id": "I2JGL2F1ZAA426PGG0YLLKABC",
        "Status": "InProgress",
        "CreateTime": "2025-05-07T16:59:25.947000+00:00",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 1,
                "Items": [
                    "/*"
                ]
            },
            "CallerReference": "invalidation-$(date +%s)"
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使文件失效以删除内容。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateInvalidationForDistributionTenant。

create-invalidation

以下代码示例演示了如何使用 create-invalidation。

Amazon CLI

为 CloudFront 分配创建失效

以下 create-invalidation 示例为指定 CloudFront 分配中的指定文件创建失效：

```
aws cloudfront create-invalidation \
    --distribution-id EDFDVBD6EXAMPLE \
    --paths "/example-path/example-file.jpg" "/example-path/example-file2.png"
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE/invalidation/I1JLWSDAP8FU89",
    "Invalidation": {
        "Id": "I1JLWSDAP8FU89",
        "Status": "InProgress",
        "CreateTime": "2019-12-05T18:24:51.407Z",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 2,
                "Items": [
                    "/example-path/example-file2.png",
                    "/example-path/example-file.jpg"
                ]
            },
            "CallerReference": "cli-1575570291-670203"
        }
    }
}
```

在前面的示例中，Amazon CLI 自动生成了一个随机 CallerReference。要指定自己的 CallerReference，或者为了避免将失效参数作为命令行参数传递，可以使用 JSON 文件。以下示例通过在名为 inv-batch.json 的 JSON 文件中提供失效参数，为两个文件创建失效：

```
aws cloudfront create-invalidation \
    --distribution-id EDFDVBD6EXAMPLE \
    --invalidation-batch file://inv-batch.json
```

的内容inv-batch.json：

```
{
    "Paths": {
        "Quantity": 2,
        "Items": [
            "/example-path/example-file.jpg",
            "/example-path/example-file2.png"
        ]
    },
    "CallerReference": "cli-example"
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE/invalidation/I2J0I21PCUYOIK",
    "Invalidation": {
        "Id": "I2J0I21PCUYOIK",
        "Status": "InProgress",
        "CreateTime": "2019-12-05T18:40:49.413Z",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 2,
                "Items": [
                    "/example-path/example-file.jpg",
                    "/example-path/example-file2.png"
                ]
            },
            "CallerReference": "cli-example"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInvalidation。

create-public-key

以下代码示例演示了如何使用 create-public-key。

Amazon CLI

创建 CloudFront 公钥

以下示例通过在名为 pub-key-config.json 的 JSON 文件中提供参数来创建 CloudFront 公钥。必须先拥有 PEM 编码的公钥，然后才能使用此命令。有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建 RSA 密钥对。

```
aws cloudfront create-public-key \
    --public-key-config file://pub-key-config.json
```

文件 pub-key-config.json 是当前文件夹中包含以下内容的 JSON 文档：请注意，公钥以 PEM 格式编码。

```
{
    "CallerReference": "cli-example",
    "Name": "ExampleKey",
    "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----END
 PUBLIC KEY-----\n",
    "Comment": "example public key"
}
```

输出：

```
{
    "Location": "https://cloudfront.amazonaws.com/2019-03-26/public-key/
KDFB19YGCR002",
    "ETag": "E2QWRUHEXAMPLE",
    "PublicKey": {
        "Id": "KDFB19YGCR002",
        "CreatedTime": "2019-12-05T18:51:43.781Z",
        "PublicKeyConfig": {
            "CallerReference": "cli-example",
            "Name": "ExampleKey",
            "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----END
 PUBLIC KEY-----\n",
            "Comment": "example public key"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePublicKey。

delete-cloud-front-origin-access-identity

以下代码示例演示了如何使用 delete-cloud-front-origin-access-identity。

Amazon CLI

删除 CloudFront 来源访问身份

以下示例删除 ID 为 E74FTE3AEXAMPLE 的来源访问身份（OAI）。要删除 OAI，您必须拥有 OAI 的 ID 和 ETag。OAI ID 将在 create-cloud-front-origin-access-identity 和 list-cloud-front-origin-access-identities 命令的输出中返回。要获取 ETag，请使用 get-cloud-front-origin-access-identity 或 get-cloud-front-origin-access-identity-config 命令。使用 --if-match 选项提供 OAI 的 ETag。

```
aws cloudfront delete-cloud-front-origin-access-identity \
    --id E74FTE3AEXAMPLE \
    --if-match E2QWRUHEXAMPLE
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCloudFrontOriginAccessIdentity。

delete-connection-group

以下代码示例演示了如何使用 delete-connection-group。

Amazon CLI

删除连接组

以下 delete-connection-group 示例删除连接组。必须禁用连接组，并且连接组不能与任何 CloudFront 资源相关联。

```
aws cloudfront delete-connection-group \
    --id cg_2wjLpjbHkLUdhWAjHllcOeABC \
    --if-match ETVPDKIKX0DABC
```

成功时，此命令没有输出。

有关管理连接组的更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteConnectionGroup。

delete-distribution-tenant

以下代码示例演示了如何使用 delete-distribution-tenant。

Amazon CLI

删除分配租户

以下 delete-distribution-tenant 示例删除带有 ETag ETVPDKIKX0DABC 的分配租户。必须禁用分配租户，并且分配租户不能与任何 CloudFront 资源相关联。

```
aws cloudfront delete-distribution-tenant \
    --id dt_2wjMUbg3NHZEQ7OfoalP5zi1AB \
    --if-match ETVPDKIKX0DABC
```

成功时，此命令没有输出。

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的删除分配。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteDistributionTenant。

delete-distribution

以下代码示例演示了如何使用 delete-distribution。

Amazon CLI

删除 CloudFront 分配

以下示例删除 ID 为 EDFDVBD6EXAMPLE 的 CloudFront 分配。删除分配之前，必须先禁用它。要禁用分配，请使用 update-distribution 命令。有关更多信息，请参阅 update-distribution 示例。

分配已禁用，您可以将其删除。要删除分配，您必须使用 --if-match 选项来提供分配的 ETag。要获取 ETag，请使用 get-distribution 或 get-distribution-config 命令。

```
aws cloudfront delete-distribution \
    --id EDFDVBD6EXAMPLE \
    --if-match E2QWRUHEXAMPLE
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDistribution。

delete-field-level-encryption-config

以下代码示例演示了如何使用 delete-field-level-encryption-config。

Amazon CLI

删除 CloudFront 字段级加密配置

以下示例删除 ID 为 C3KM2WVD605UAY 的 CloudFront 字段级加密配置。要删除字段级加密配置，必须提供其 ID 和 ETag。该 ID 将在 create-field-level-encryption-config 和 list-field-level-encryption-configs 命令的输出中返回。要获取 ETag，请使用 get-field-level-encryption 或 get-field-level-encryption-config 命令。使用 --if-match 选项提供配置的 ETag。

```
aws cloudfront delete-field-level-encryption-config \
    --id C3KM2WVD605UAY \
    --if-match E26M4BIAV81ZF6
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFieldLevelEncryptionConfig。

delete-field-level-encryption-profile

以下代码示例演示了如何使用 delete-field-level-encryption-profile。

Amazon CLI

删除 CloudFront 字段级加密配置文件

以下示例删除 ID 为 PPK0UOSIF5WSV 的 CloudFront 字段级加密配置文件。要删除字段级加密配置文件，必须提供其 ID 和 ETag。该 ID 将在 create-field-level-encryption-profile 和 list-field-level-encryption-profiles 命令的输出中返回。要获取 ETag，请使用 get-field-level-encryption-profile 或 get-field-level-encryption-profile-config 命令。使用 --if-match 选项提供配置文件的 ETag。

```
aws cloudfront delete-field-level-encryption-profile \
    --id PPK0UOSIF5WSV \
    --if-match EJETYFJ9CL66D
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFieldLevelEncryptionProfile。

delete-public-key

以下代码示例演示了如何使用 delete-public-key。

Amazon CLI

删除 CloudFront 公钥

以下示例删除 ID 为 KDFB19YGCR002 的 CloudFront 公钥。要删除公钥，必须提供其 ID 和 ETag。该 ID 将在 create-public-key 和 list-public-keys 命令的输出中返回。要获取 ETag，请使用 get-public-key 或 get-public-key-config 命令。使用 --if-match 选项提供公钥的 ETag。

```
aws cloudfront delete-public-key \
    --id KDFB19YGCR002 \
    --if-match E2QWRUHEXAMPLE
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePublicKey。

disassociate-distribution-tenant-web-acl

以下代码示例演示了如何使用 disassociate-distribution-tenant-web-acl。

Amazon CLI

将 Web ACL 与分配租户取消关联

以下 disassociate-distribution-tenant-web-acl 示例将 Web ACL 与具有 ETag E1PA6795UKMABC 的分配租户取消关联。

```
aws cloudfront disassociate-distribution-tenant-web-acl \
    --id dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB \
    --if-match E1PA6795UKMABC
```

输出：

```
{
```

```
    "ETag": "E13V1IB3VIYABC",
    "Id": "dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB"
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的禁用 Amazon WAF 安全保护功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateDistributionTenantWebAcl。

`disassociate-distribution-web-acl`

以下代码示例演示了如何使用 `disassociate-distribution-web-acl`。

Amazon CLI

将 Web ACL 与 CloudFront 分配取消关联

以下 `disassociate-distribution-web-acl` 示例取消 Web ACL 与具有 ETag E13V1IB3VIYABC 的 CloudFront 分配之间的关联。

```
aws cloudfront disassociate-distribution-web-acl \
    --id E1XNX8R2GOAABC \
    --if-match EEZQ9Z24VM1ABC
```

输出：

```
{
    "ETag": "E2YWS1C2J3OABC",
    "Id": "E1XNX8R2GOAABC"
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的禁用 Amazon WAF 安全保护功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateDistributionWebAcl。

`get-cloud-front-origin-access-identity-config`

以下代码示例演示了如何使用 `get-cloud-front-origin-access-identity-config`。

Amazon CLI

获取 CloudFront 来源访问身份配置

以下示例获取有关 ID 为 E74FTE3AEXAMPLE 的 CloudFront 来源访问身份（OAI）的元数据，包括其 ETag。OAI ID 将在 create-cloud-front-origin-access-identity 和 list-cloud-front-origin-access-identities 命令的输出中返回。

```
aws cloudfront get-cloud-front-origin-access-identity-config --id E74FTE3AEXAMPLE
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "CloudFrontOriginAccessIdentityConfig": {
        "CallerReference": "cli-example",
        "Comment": "Example OAI"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCloudFrontOriginAccessIdentityConfig。

get-cloud-front-origin-access-identity

以下代码示例演示了如何使用 get-cloud-front-origin-access-identity。

Amazon CLI

获取 CloudFront 来源访问身份

以下示例获取 ID 为 E74FTE3AEXAMPLE 的 CloudFront 来源访问身份（OAI），包括其 ETag 和关联的 S3 规范 ID。OAI ID 将在 create-cloud-front-origin-access-identity 和 list-cloud-front-origin-access-identities 命令的输出中返回。

```
aws cloudfront get-cloud-front-origin-access-identity --id E74FTE3AEXAMPLE
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
```

```
    "CloudFrontOriginAccessIdentity": {
        "Id": "E74FTE3AEXAMPLE",
        "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
        "CloudFrontOriginAccessIdentityConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example OAI"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCloudFrontOriginAccessIdentity。

get-connection-group-by-routing-endpoint

以下代码示例演示了如何使用 get-connection-group-by-routing-endpoint。

Amazon CLI

通过路由端点获取连接组

以下 get-connection-group-by-routing-endpoint 示例使用连接组的路由端点检索有关该连接组的信息。

```
aws cloudfront get-connection-group-by-routing-endpoint \
    --routing-endpoint dvdg9gprgabc.cloudfront.net
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "ConnectionGroup": {
        "Id": "cg_2wjDWTBKTlRB87cAaUQFaakABC",
        "Name": "connection-group-2",
        "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2wjDWTBKTlRB87cAaUQFaakABC",
        "CreatedTime": "2025-05-06T15:42:00.790000+00:00",
        "LastModifiedTime": "2025-05-06T15:42:00.790000+00:00",
        "Ipv6Enabled": true,
        "RoutingEndpoint": "dvdg9gprgabc.cloudfront.net",
        "Status": "Deployed",
```

```
        "Enabled": true,
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetConnectionGroupByRoutingEndpoint。

get-connection-group

以下代码示例演示了如何使用 get-connection-group。

Amazon CLI

获取 CloudFront 连接组

以下 get-connection-group 示例检索有关 CloudFront 连接组的信息。

```
aws cloudfront get-connection-group \
    --identifier cg_2wjDWTBKTlRB87cAaUQFaakABC
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "ConnectionGroup": {
        "Id": "cg_2wjDWTBKTlRB87cAaUQFaakABC",
        "Name": "connection-group-2",
        "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2wjDWTBKTlRB87cAaUQFaakABC",
        "CreatedTime": "2025-05-06T15:42:00.790000+00:00",
        "LastModifiedTime": "2025-05-06T15:42:00.790000+00:00",
        "Ipv6Enabled": true,
        "RoutingEndpoint": "dvdg9gprgabc.cloudfront.net",
        "Status": "Deployed",
        "Enabled": true,
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetConnectionGroup。

get-distribution-config

以下代码示例演示了如何使用 get-distribution-config。

Amazon CLI

获取 CloudFront 分配配置

以下示例获取有关 ID 为 EDFDVBD6EXAMPLE 的 CloudFront 分配的元数据，包括其 ETag。分配 ID 将在 create-distribution 和 list-distributions 命令中返回。

```
aws cloudfront get-distribution-config \
    --id EDFDVBD6EXAMPLE
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "DistributionConfig": {
        "CallerReference": "cli-example",
        "Aliases": {
            "Quantity": 0
        },
        "DefaultRootObject": "index.html",
        "Origins": {
            "Quantity": 1,
            "Items": [
                {
                    "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                    "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                    "OriginPath": "",
                    "CustomHeaders": {
                        "Quantity": 0
                    },
                    "S3OriginConfig": {
                        "OriginAccessIdentity": ""
                    }
                }
            ]
        },
        "OriginGroups": {
```

```
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                },
                "Headers": {
                    "Quantity": 0
                },
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "SmoothStreaming": false,
            "DefaultTTL": 86400,
            "MaxTTL": 31536000,
            "Compress": false,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": ""
```

```
        },
        "CacheBehaviors": {
            "Quantity": 0
        },
        "CustomErrorResponses": {
            "Quantity": 0
        },
        "Comment": "",
        "Logging": {
            "Enabled": false,
            "IncludeCookies": false,
            "Bucket": "",
            "Prefix": ""
        },
        "PriceClass": "PriceClass_All",
        "Enabled": true,
        "ViewerCertificate": {
            "CloudFrontDefaultCertificate": true,
            "MinimumProtocolVersion": "TLSv1",
            "CertificateSource": "cloudfront"
        },
        "Restrictions": {
            "GeoRestriction": {
                "RestrictionType": "none",
                "Quantity": 0
            }
        },
        "WebACLId": "",
        "HttpVersion": "http2",
        "IsIPV6Enabled": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDistributionConfig。

get-distribution-tenant-by-domain

以下代码示例演示了如何使用 get-distribution-tenant-by-domain。

Amazon CLI

按域获取分配租户信息

以下 get-distribution-tenant-by-domain 示例使用指定的域检索有关分配租户的信息。

```
aws cloudfront get-distribution-tenant-by-domain \
    --domain example.com
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "DistributionTenant": {
        "Id": "dt_2xVInRKCfUzQHgxosDs9hiLk1AB",
        "DistributionId": "E1XNX8R2GOAABC",
        "Name": "example-tenant-4",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2xVInRKCfUzQHgxosDs9hiLk1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "active"
            }
        ],
        "Parameters": [
            {
                "Name": "testParam",
                "Value": "defaultValue"
            }
        ],
        "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
        "CreatedTime": "2025-05-23T16:16:20.871000+00:00",
        "LastModifiedTime": "2025-05-23T16:16:20.871000+00:00",
        "Enabled": false,
        "Status": "Deployed"
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的了解多租户分配的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDistributionTenantByDomain。

get-distribution-tenant

以下代码示例演示了如何使用 get-distribution-tenant。

Amazon CLI

获取有关 CloudFront 分配租户的详细信息

以下 get-distribution-tenant 示例检索有关 CloudFront 分配租户的信息。

```
aws cloudfront get-distribution-tenant \
    --id dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB
```

输出：

```
{
    "ETag": "E23ZP02F085ABC",
    "DistributionTenant": {
        "Id": "dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB",
        "DistributionId": "E1XNX8R2GOAABC",
        "Name": "example-tenant-2",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "inactive"
            }
        ],
        "ConnectionGroupId": "cg_2wjDWTBKTlRB87cAaUQFaakABC",
        "CreatedTime": "2025-05-06T15:42:28.542000+00:00",
        "LastModifiedTime": "2025-05-06T15:42:37.724000+00:00",
        "Enabled": true,
        "Status": "InProgress"
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的了解多租户分配的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDistributionTenant。

get-distribution

以下代码示例演示了如何使用 get-distribution。

Amazon CLI

获取 CloudFront 分配

以下 get-distribution 示例获取 ID 为 EDFDVBD6EXAMPLE 的 CloudFront 分配，包括其 ETag。分配 ID 将在 create-distribution 和 list-distributions 命令中返回。

```
aws cloudfront get-distribution \
    --id EDFDVBD6EXAMPLE
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "Distribution": {
        "Id": "EDFDVBD6EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
        "Status": "Deployed",
        "LastModifiedTime": "2019-12-04T23:35:41.433Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-example",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
```

```
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
                "MinTTL": 0,
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
```

```
                    }
                },
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDistribution。

get-field-level-encryption-config

以下代码示例演示了如何使用 get-field-level-encryption-config。

Amazon CLI

获取有关 CloudFront 字段级加密配置的元数据

以下示例获取 ID 为 C3KM2WVD605UAY 的 CloudFront 字段级加密配置的元数据，包括其 ETag：

```
aws cloudfront get-field-level-encryption-config --id C3KM2WVD605UAY
```

输出：

```
{
    "ETag": "E2P4Z4VU7TY5SG",
    "FieldLevelEncryptionConfig": {
        "CallerReference": "cli-example",
        "Comment": "Example FLE configuration",
        "QueryArgProfileConfig": {
            "ForwardWhenQueryArgProfileIsUnknown": true,
            "QueryArgProfiles": {
                "Quantity": 0,
                "Items": []
            }
        },
        "ContentTypeProfileConfig": {
            "ForwardWhenContentTypeIsUnknown": true,
            "ContentTypeProfiles": {
                "Quantity": 1,
                "Items": [
                    {
                        "Format": "URLEncoded",
                        "ProfileId": "P280MFCLSYOCVU",
                        "ContentType": "application/x-www-form-urlencoded"
                    }
                ]
            }
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFieldLevelEncryptionConfig。

get-field-level-encryption-profile-config

以下代码示例演示了如何使用 get-field-level-encryption-profile-config。

Amazon CLI

获取 CloudFront 字段级加密配置文件配置

以下示例获取 ID 为 PPK0UOSIF5WSV 的 CloudFront 字段级加密配置文件的元数据，包括其 ETag：

```
aws cloudfront get-field-level-encryption-profile-config --id PPK0UOSIF5WSV
```

输出：

```
{
    "ETag": "E1QQG65FS2L2GC",
    "FieldLevelEncryptionProfileConfig": {
        "Name": "ExampleFLEProfile",
        "CallerReference": "cli-example",
        "Comment": "FLE profile for AWS CLI example",
        "EncryptionEntities": {
            "Quantity": 1,
            "Items": [
                {
                    "PublicKeyId": "K2K8NC4HVFE3M0",
                    "ProviderId": "ExampleFLEProvider",
                    "FieldPatterns": {
                        "Quantity": 1,
                        "Items": [
                            "ExampleSensitiveField"
                        ]
                    }
                }
            ]
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFieldLevelEncryptionProfileConfig。

get-field-level-encryption-profile

以下代码示例演示了如何使用 get-field-level-encryption-profile。

Amazon CLI

获取 CloudFront 字段级加密配置文件

以下示例获取 ID 为 PPK0UOSIF5WSV 的 CloudFront 字段级加密配置文件，包括其 ETag：

```
aws cloudfront get-field-level-encryption-profile --id PPK0UOSIF5WSV
```

输出：

```
{
    "ETag": "E1QQG65FS2L2GC",
    "FieldLevelEncryptionProfile": {
        "Id": "PPK0UOSIF5WSV",
        "LastModifiedTime": "2019-12-10T01:03:16.537Z",
        "FieldLevelEncryptionProfileConfig": {
            "Name": "ExampleFLEProfile",
            "CallerReference": "cli-example",
            "Comment": "FLE profile for AWS CLI example",
            "EncryptionEntities": {
                "Quantity": 1,
                "Items": [
                    {
                        "PublicKeyId": "K2K8NC4HVFE3M0",
                        "ProviderId": "ExampleFLEProvider",
                        "FieldPatterns": {
                            "Quantity": 1,
                            "Items": [
                                "ExampleSensitiveField"
                            ]
                        }
                    }
```

```
                ]
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFieldLevelEncryptionProfile。

get-field-level-encryption

以下代码示例演示了如何使用 get-field-level-encryption。

Amazon CLI

获取 CloudFront 字段级加密配置

以下示例获取 ID 为 C3KM2WVD605UAY 的 CloudFront 字段级加密配置，包括其 ETag：

```
aws cloudfront get-field-level-encryption --id C3KM2WVD605UAY
```

输出：

```
{
    "ETag": "E2P4Z4VU7TY5SG",
    "FieldLevelEncryption": {
        "Id": "C3KM2WVD605UAY",
        "LastModifiedTime": "2019-12-10T21:30:18.974Z",
        "FieldLevelEncryptionConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example FLE configuration",
            "QueryArgProfileConfig": {
                "ForwardWhenQueryArgProfileIsUnknown": true,
                "QueryArgProfiles": {
                    "Quantity": 0,
                    "Items": []
                }
            },
            "ContentTypeProfileConfig": {
                "ForwardWhenContentTypeIsUnknown": true,
                "ContentTypeProfiles": {
                    "Quantity": 1,
                    "Items": [
```

```
                        {
                            "Format": "URLEncoded",
                            "ProfileId": "P280MFCLSYOCVU",
                            "ContentType": "application/x-www-form-urlencoded"
                        }
                    ]
                }
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFieldLevelEncryption。

get-invalidation-for-distribution-tenant

以下代码示例演示了如何使用 get-invalidation-for-distribution-tenant。

Amazon CLI

为分配租户获取失效

以下 get-invalidation-for-distribution-tenant 示例获取有关分配租户的失效的信息。

```
aws cloudfront get-invalidation-for-distribution-tenant \
    --distribution-tenant-id dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB \
    --id I4CU23QAPKMUDUU06F9OFGFABC
```

输出：

```
{
    "Invalidation": {
        "Id": "I4CU23QAPKMUDUU06F9OFGFABC",
        "Status": "Completed",
        "CreateTime": "2025-05-06T15:46:12.824000+00:00",
        "InvalidationBatch": {
            "Paths": {
                "Quantity": 2,
                "Items": [
                    "/example/invalidation",
```

```
                    "/more/invalidations"
                ]
            },
            "CallerReference": "007ee5a6-d0a0-42be-bb61-e7b915969b48"
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使文件失效以删除内容。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetInvalidationForDistributionTenant。

`get-invalidation`

以下代码示例演示了如何使用 `get-invalidation`。

Amazon CLI

获取 CloudFront 失效

以下示例获取 ID 为 `I2J0I21PCUYOIK` 的 CloudFront 分配的失效，其 ID 为 `EDFDVBD6EXAMPLE`：

```
aws cloudfront get-invalidation --id I2J0I21PCUYOIK --distribution-
id EDFDVBD6EXAMPLE
```

输出：

```
{
    "Invalidation": {
        "Status": "Completed",
        "InvalidationBatch": {
            "Paths": {
                "Items": [
                    "/example-path/example-file.jpg",
                    "/example-path/example-file-2.jpg"
                ],
                "Quantity": 2
            },
            "CallerReference": "cli-example"
```

```
        },
        "Id": "I2J0I21PCUYOIK",
        "CreateTime": "2019-12-05T18:40:49.413Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInvalidation。

get-managed-certificate-details

以下代码示例演示了如何使用 get-managed-certificate-details。

Amazon CLI

获取托管式证书详细信息

以下 get-managed-certificate-details 示例检索 CloudFront 托管式 ACM 证书的详细信息。

```
aws cloudfront get-managed-certificate-details \
    --identifier dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB
```

输出：

```
{
    "ManagedCertificateDetails": {
        "CertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/655dc1fe-6d37-451d-a013-c2db3a034abc",
        "CertificateStatus": "pending-validation",
        "ValidationTokenHost": "self-hosted",
        "ValidationTokenDetails": [
            {
                "Domain": "example.com",
                "RedirectTo": "validation.us-east-1.acm-
validations.aws/123456789012/.well-known/pki-validation/
b315c9ae21284e7918bb9f3f422ab1c7.txt",
                "RedirectFrom": "example.com/.well-known/pki-validation/
b315c9ae21284e7918bb9f3f422ac3c7.txt"
            }
        ]
    }
```

```
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的为 CloudFront 分配租户申请证书。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetManagedCertificateDetails。

get-public-key-config

以下代码示例演示了如何使用 get-public-key-config。

Amazon CLI

获取 CloudFront 公钥配置

以下示例获取 ID 为 KDFB19YGCR002 的 CloudFront 公钥的元数据，包括其 ETag。该公钥 ID 将在 create-public-key 和 list-public-keys 命令中返回。

```
aws cloudfront get-public-key-config --id KDFB19YGCR002
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "PublicKeyConfig": {
        "CallerReference": "cli-example",
        "Name": "ExampleKey",
        "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----END
 PUBLIC KEY-----\n",
        "Comment": "example public key"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicKeyConfig。

get-public-key

以下代码示例演示了如何使用 get-public-key。

Amazon CLI

获取 CloudFront 公钥

以下示例获取 ID 为 KDFB19YGCR002 的 CloudFront 公钥，包括其 ETag。该公钥 ID 将在 create-public-key 和 list-public-keys 命令中返回。

```
aws cloudfront get-public-key --id KDFB19YGCR002
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "PublicKey": {
        "Id": "KDFB19YGCR002",
        "CreatedTime": "2019-12-05T18:51:43.781Z",
        "PublicKeyConfig": {
            "CallerReference": "cli-example",
            "Name": "ExampleKey",
            "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----END
 PUBLIC KEY-----\n",
            "Comment": "example public key"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicKey。

list-cloud-front-origin-access-identities

以下代码示例演示了如何使用 list-cloud-front-origin-access-identities。

Amazon CLI

列出 CloudFront 来源访问身份

以下示例获取您 Amazon 账户中的 CloudFront 来源访问身份（OAI）列表：

```
aws cloudfront list-cloud-front-origin-access-identities
```

输出：

```
{
    "CloudFrontOriginAccessIdentityList": {
        "Items": [
            {
                "Id": "E74FTE3AEXAMPLE",
                "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
                "Comment": "Example OAI"
            },
            {
                "Id": "EH1HDMBEXAMPLE",
                "S3CanonicalUserId":
 "1489f6f2e6faacaae7ff64c4c3e6956c24f78788abfc1718c3527c263bf7a17EXAMPLE",
                "Comment": "Test OAI"
            },
            {
                "Id": "E2X2C9TEXAMPLE",
                "S3CanonicalUserId":
 "cbfeebb915a64749f9be546a45b3fcfd3a31c779673c13c4dd460911ae402c2EXAMPLE",
                "Comment": "Example OAI #2"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCloudFrontOriginAccessIdentities。

`list-connection-groups`

以下代码示例演示了如何使用 `list-connection-groups`。

Amazon CLI

列出连接组

以下 `list-connection-groups` 示例列出您的 Amazon 账户中的可用连接组。

```
aws cloudfront list-connection-groups
```

输出：

```
{
    "ConnectionGroups": [
        {
            "Id": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "Name": "CreatedByCloudFront-cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "RoutingEndpoint": "d3sx0pso7m5abc.cloudfront.net",
            "CreatedTime": "2025-05-05T22:32:29.630000+00:00",
            "LastModifiedTime": "2025-05-05T22:32:29.630000+00:00",
            "ETag": "E23ZP02F085ABC",
            "Enabled": true,
            "Status": "Deployed",
            "IsDefault": true
        },
        {
            "Id": "cg_2wjDWTBKTlRB87cAaUQFaakABC",
            "Name": "connection-group-2",
            "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2wjDWTBKTlRB87cAaUQFaakABC",
            "RoutingEndpoint": "dvdg9gprgabc.cloudfront.net",
            "CreatedTime": "2025-05-06T15:42:00.790000+00:00",
            "LastModifiedTime": "2025-05-06T15:42:00.790000+00:00",
            "ETag": "E23ZP02F085ABC",
            "Enabled": true,
            "Status": "Deployed",
            "IsDefault": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListConnectionGroups。

`list-distribution-tenants-by-customization`

以下代码示例演示了如何使用 `list-distribution-tenants-by-customization`。

Amazon CLI

通过自定义项列出分配租户

以下 `list-distribution-tenants-by-customization` 示例列出使用指定的 Web ACL 的分配租户。

```
aws cloudfront list-distribution-tenants-by-customization \
    --web-acl-arn arn:aws:wafv2:us-east-1:123456789012:global/webacl/
CreatedByCloudFront-0273cd2f/a3c19bce-42b5-48a1-a8d4-b2bb2f28eabc
```

输出：

```
{
    "DistributionTenantList": [
        {
            "Id": "dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB",
            "DistributionId": "E1XNX8R2GOAABC",
            "Name": "example-tenant-2",
            "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB",
            "Domains": [
                {
                    "Domain": "example.com",
                    "Status": "inactive"
                }
            ],
            "ConnectionGroupId": "cg_2wjDWTBKTlRB87cAaUQFaakABC",
            "Customizations": {
                "WebAcl": {
                    "Action": "override",
                    "Arn": "arn:aws:wafv2:us-east-1:123456789012:global/webacl/
CreatedByCloudFront-0273cd2f/a3c19bce-42b5-48a1-a8d4-b2bb2f28eabc"
                },
                "GeoRestrictions": {
```

```
                    "RestrictionType": "whitelist",
                    "Locations": [
                        "AL"
                    ]
                }
            },
            "CreatedTime": "2025-05-06T15:42:28.542000+00:00",
            "LastModifiedTime": "2025-05-06T16:14:08.710000+00:00",
            "ETag": "E1F83G8C2ARABC",
            "Enabled": true,
            "Status": "Deployed"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的分配租户自定义。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDistributionTenantsByCustomization。

`list-distribution-tenants`

以下代码示例演示了如何使用 `list-distribution-tenants`。

Amazon CLI

列出 CloudFront 分配租户

以下 `list-distribution-tenants` 示例按关联的连接组列出您 Amazon 账户中的 3 个 CloudFront 分配租户。

```
aws cloudfront list-distribution-tenants \
    --association-filter ConnectionGroupId=cg_2whCJoXMYCjHcxaLGrkllvyABC \
    --max-items 3
```

输出：

```
{
    "DistributionTenantList": [
        {
            "Id": "dt_2yMvQgam3QkJo2z54FDl91dk1AB",
            "DistributionId": "E1XNX8R2GOAABC",
            "Name": "new-tenant-customizations",
```

```
            "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2yMvQgam3QkJo2z54FDl91dk1AB",
            "Domains": [
                {
                    "Domain": "example.com",
                    "Status": "active"
                }
            ],
            "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "Customizations": {
                "WebAcl": {
                    "Action": "disable"
                },
                "GeoRestrictions": {
                    "RestrictionType": "whitelist",
                    "Locations": [
                        "DE"
                    ]
                }
            },
            "CreatedTime": "2025-06-11T15:54:02.142000+00:00",
            "LastModifiedTime": "2025-06-11T15:54:02.142000+00:00",
            "ETag": "E23ZP02F085ABC",
            "Enabled": false,
            "Status": "Deployed"
        },
        {
            "Id": "dt_2yMuV7NJuBcAB0cwwxMCBZQ1AB",
            "DistributionId": "E1XNX8R2GOAABC",
            "Name": "new-tenant",
            "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2yMuV7NJuBcAB0cwwxMCBZQ1AB",
            "Domains": [
                {
                    "Domain": "1.example.com",
                    "Status": "active"
                }
            ],
            "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "Customizations": {
                "GeoRestrictions": {
                    "RestrictionType": "whitelist",
                    "Locations": [
                        "DE"
```

```
                    ]
                }
            },
            "CreatedTime": "2025-06-11T15:46:23.466000+00:00",
            "LastModifiedTime": "2025-06-11T15:46:23.466000+00:00",
            "ETag": "E23ZP02F085ABC",
            "Enabled": false,
            "Status": "Deployed"
        },
        {
            "Id": "dt_2xVInRKCfUzQHgxosDs9hiLk1AB",
            "DistributionId": "E1XNX8R2GOAABC",
            "Name": "new-tenant-2",
            "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2xVInRKCfUzQHgxosDs9hiLk1AB",
            "Domains": [
                {
                    "Domain": "2.example.com",
                    "Status": "active"
                }
            ],
            "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
            "CreatedTime": "2025-05-23T16:16:20.871000+00:00",
            "LastModifiedTime": "2025-05-23T16:16:20.871000+00:00",
            "ETag": "E23ZP02F085ABC",
            "Enabled": false,
            "Status": "Deployed"
        }
    ],
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAzfQ=="
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的了解多租户分配的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDistributionTenants。

`list-distributions-by-connection-mode`

以下代码示例演示了如何使用 `list-distributions-by-connection-mode`。

Amazon CLI

按连接模式列出 CloudFront 分配

以下 `list-distributions-by-connection-mode` 示例列出具有指定的连接模式的 CloudFront 分配。

```
aws cloudfront list-distributions-by-connection-mode \
    --connection-mode tenant-only
```

输出：

```
{
    "DistributionList": {
        "Items": [
            {
                "Id": "E1XNX8R2GOAABC",
                "ARN": "arn:aws:cloudfront::123456789012:distribution/
E1XNX8R2GOAABC",
                "ETag": "EPT4JPJQDY1ABC",
                "Status": "Deployed",
                "LastModifiedTime": "2025-05-23T16:16:15.691000+00:00",
                "DomainName": "-",
                "Aliases": {
                    "Quantity": 0
                },
                "Origins": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "Id": "example-cfn-simple-distribution123",
                            "DomainName": "example.com",
                            "OriginPath": "",
                            "CustomHeaders": {
                                "Quantity": 0
                            },
                            "S3OriginConfig": {
                                "OriginAccessIdentity": ""
                            },
                            "ConnectionAttempts": 3,
                            "ConnectionTimeout": 10,
                            "OriginShield": {
                                "Enabled": false
```

```
                    },
                    "OriginAccessControlId": "E2CJRMB5LKEABC"
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "example-cfn-simple-distribution123",
            "TrustedKeyGroups": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
            "Compress": true,
            "LambdaFunctionAssociations": {
                "Quantity": 0
            },
            "FunctionAssociations": {
                "Quantity": 0
            },
            "FieldLevelEncryptionId": "",
            "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5abc",
            "GrpcConfig": {
                "Enabled": false
            }
        },
        "CacheBehaviors": {
            "Quantity": 0
```

```
                },
                "CustomErrorResponses": {
                    "Quantity": 0
                },
                "Comment": "",
                "PriceClass": "PriceClass_All",
                "Enabled": true,
                "ViewerCertificate": {
                    "CloudFrontDefaultCertificate": false,
                    "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/ec53f564-ea5a-4e4a-a0a2-e3c989449abc",
                    "SSLSupportMethod": "sni-only",
                    "MinimumProtocolVersion": "TLSv1.2_2021",
                    "Certificate": "arn:aws:acm:us-east-1:123456789012:certificate/
ec53f564-ea5a-4e4a-a0a2-e3c989449abc",
                    "CertificateSource": "acm"
                },
                "Restrictions": {
                    "GeoRestriction": {
                        "RestrictionType": "none",
                        "Quantity": 0
                    }
                },
                "WebACLId": "arn:aws:wafv2:us-east-1:123456789012:global/webacl/web-
global-example/626900da-5f64-418b-ba9b-743f3746cabc",
                "HttpVersion": "http2",
                "IsIPV6Enabled": false,
                "Staging": false,
                "ConnectionMode": "tenant-only"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDistributionsByConnectionMode。

`list-distributions`

以下代码示例演示了如何使用 `list-distributions`。

Amazon CLI

列出 CloudFront 分配

以下示例获取您的 Amazon 账户中的 CloudFront 分配列表。

```
aws cloudfront list-distributions
```

输出：

```
{
    "DistributionList": {
        "Items": [
            {
                "Id": "E23YS8OEXAMPLE",
                "ARN": "arn:aws:cloudfront::123456789012:distribution/
E23YS8OEXAMPLE",
                "Status": "Deployed",
                "LastModifiedTime": "2024-08-05T18:23:40.375000+00:00",
                "DomainName": "abcdefgh12ijk.cloudfront.net",
                "Aliases": {
                    "Quantity": 0
                },
                "Origins": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
                            "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                            "OriginPath": "",
                            "CustomHeaders": {
                                "Quantity": 0
                            },
                            "S3OriginConfig": {
                                "OriginAccessIdentity": ""
                            },
                            "ConnectionAttempts": 3,
                            "ConnectionTimeout": 10,
                            "OriginShield": {
                                "Enabled": false
                            },
                            "OriginAccessControlId": "EIAP8PEXAMPLE"
```

```
                        }
                    ]
                },
                "OriginGroups": {
                    "Quantity": 0
                },
                "DefaultCacheBehavior": {
                    "TargetOriginId": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
                    "TrustedSigners": {
                        "Enabled": false,
                        "Quantity": 0
                    },
                    "TrustedKeyGroups": {
                        "Enabled": false,
                        "Quantity": 0
                    },
                    "ViewerProtocolPolicy": "allow-all",
                    "AllowedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ],
                        "CachedMethods": {
                            "Quantity": 2,
                            "Items": [
                                "HEAD",
                                "GET"
                            ]
                        }
                    },
                    "SmoothStreaming": false,
                    "Compress": true,
                    "LambdaFunctionAssociations": {
                        "Quantity": 0
                    },
                    "FunctionAssociations": {
                        "Quantity": 0
                    },
                    "FieldLevelEncryptionId": "",
                    "CachePolicyId": "658327ea-f89d-4fab-a63d-7e886EXAMPLE"
                },
                "CacheBehaviors": {
```

```
                    "Quantity": 0
                },
                "CustomErrorResponses": {
                    "Quantity": 0
                },
                "Comment": "",
                "PriceClass": "PriceClass_All",
                "Enabled": true,
                "ViewerCertificate": {
                    "CloudFrontDefaultCertificate": true,
                    "SSLSupportMethod": "vip",
                    "MinimumProtocolVersion": "TLSv1",
                    "CertificateSource": "cloudfront"
                },
                "Restrictions": {
                    "GeoRestriction": {
                        "RestrictionType": "none",
                        "Quantity": 0
                    }
                },
                "WebACLId": "",
                "HttpVersion": "HTTP2",
                "IsIPV6Enabled": true,
                "Staging": false
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDistributions。

`list-domain-conflicts`

以下代码示例演示了如何使用 `list-domain-conflicts`。

Amazon CLI

列出 CloudFront 分配的域冲突

以下 `list-domain-conflicts` 示例列出 CloudFront 分配的域冲突。

```
aws cloudfront list-domain-conflicts \
    --domain example.com \
```

```
    --domain-control-validation-
resource "DistributionTenantId=dt_2x9GhoK0TZRsohWzv1b9It8J1AB"
```

输出：

```
{
    "DomainConflicts": [
        {
            "Domain": "example.com",
            "ResourceType": "distribution-tenant",
            "ResourceId": "***************ohWzv1b9It8J1AB",
            "AccountId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的将备用域名移动到其他分配。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDomainConflicts。

list-field-level-encryption-configs

以下代码示例演示了如何使用 list-field-level-encryption-configs。

Amazon CLI

列出 CloudFront 字段级加密配置

以下示例获取您 Amazon 账户中的 CloudFront 字段级加密配置列表：

```
aws cloudfront list-field-level-encryption-configs
```

输出：

```
{
    "FieldLevelEncryptionList": {
        "MaxItems": 100,
        "Quantity": 1,
        "Items": [
            {
                "Id": "C3KM2WVD605UAY",
                "LastModifiedTime": "2019-12-10T21:30:18.974Z",
                "Comment": "Example FLE configuration",
```

```
                "QueryArgProfileConfig": {
                    "ForwardWhenQueryArgProfileIsUnknown": true,
                    "QueryArgProfiles": {
                        "Quantity": 0,
                        "Items": []
                    }
                },
                "ContentTypeProfileConfig": {
                    "ForwardWhenContentTypeIsUnknown": true,
                    "ContentTypeProfiles": {
                        "Quantity": 1,
                        "Items": [
                            {
                                "Format": "URLEncoded",
                                "ProfileId": "P280MFCLSYOCVU",
                                "ContentType": "application/x-www-form-urlencoded"
                            }
                        ]
                    }
                }
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFieldLevelEncryptionConfigs。

list-field-level-encryption-profiles

以下代码示例演示了如何使用 list-field-level-encryption-profiles。

Amazon CLI

列出 CloudFront 字段级加密配置文件

以下示例获取您 Amazon 账户中的 CloudFront 字段级加密配置文件列表：

```
aws cloudfront list-field-level-encryption-profiles
```

输出：

```
{
    "FieldLevelEncryptionProfileList": {
```

```
        "MaxItems": 100,
        "Quantity": 2,
        "Items": [
            {
                "Id": "P280MFCLSYOCVU",
                "LastModifiedTime": "2019-12-05T01:05:39.896Z",
                "Name": "ExampleFLEProfile",
                "EncryptionEntities": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "PublicKeyId": "K2K8NC4HVFE3M0",
                            "ProviderId": "ExampleFLEProvider",
                            "FieldPatterns": {
                                "Quantity": 1,
                                "Items": [
                                    "ExampleSensitiveField"
                                ]
                            }
                        }
                    ]
                },
                "Comment": "FLE profile for AWS CLI example"
            },
            {
                "Id": "PPK0UOSIF5WSV",
                "LastModifiedTime": "2019-12-10T01:03:16.537Z",
                "Name": "ExampleFLEProfile2",
                "EncryptionEntities": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "PublicKeyId": "K2ABC10EXAMPLE",
                            "ProviderId": "ExampleFLEProvider2",
                            "FieldPatterns": {
                                "Quantity": 1,
                                "Items": [
                                    "ExampleSensitiveField2"
                                ]
                            }
                        }
                    ]
                },
                "Comment": "FLE profile #2 for AWS CLI example"
```

```
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFieldLevelEncryptionProfiles。

`list-invalidations-for-distribution-tenant`

以下代码示例演示了如何使用 `list-invalidations-for-distribution-tenant`。

Amazon CLI

列出 CloudFront 分配租户的失效

以下 `list-invalidations-for-distribution-tenant` 示例列出 CloudFront 分配租户的失效。

```
aws cloudfront list-invalidations-for-distribution-tenant \
    --id dt_2wjDZi3hD1ivOXf6rpZJOSNE1AB
```

输出：

```
{
    "InvalidationList": {
        "Items": [
            {
                "Id": "I4CU23QAPKMUDUU06F9OFGFABC",
                "CreateTime": "2025-05-06T15:46:12.824000+00:00",
                "Status": "Completed"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使文件失效以删除内容。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListInvalidationsForDistributionTenant。

`list-invalidations`

以下代码示例演示了如何使用 `list-invalidations`。

Amazon CLI

列出 CloudFront 失效

以下示例获取 ID 为 EDFDVBD6EXAMPLE 的 CloudFront 分配的失效列表：

```
aws cloudfront list-invalidations --distribution-id EDFDVBD6EXAMPLE
```

输出：

```
{
    "InvalidationList": {
        "Marker": "",
        "Items": [
            {
                "Status": "Completed",
                "Id": "YNY2LI2BVJ4NJU",
                "CreateTime": "2019-08-31T21:15:52.042Z"
            }
        ],
        "IsTruncated": false,
        "MaxItems": 100,
        "Quantity": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInvalidations。

`list-public-keys`

以下代码示例演示了如何使用 `list-public-keys`。

Amazon CLI

列出 CloudFront 公钥

以下示例获取您 Amazon 账户中的 CloudFront 公钥列表：

```
aws cloudfront list-public-keys
```

输出：

```
{
    "PublicKeyList": {
        "MaxItems": 100,
        "Quantity": 2,
        "Items": [
            {
                "Id": "K2K8NC4HVFE3M0",
                "Name": "ExampleKey",
                "CreatedTime": "2019-12-05T01:04:28.818Z",
                "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1UOWcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAzO3ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplcOkjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6ZqlV3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nq
+kGZ2NQ0FyIyT2eiLKOX5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----END
 PUBLIC KEY-----\n",
                "Comment": "example public key"
            },
            {
                "Id": "K1S0LWQ2L5HTBU",
                "Name": "ExampleKey2",
                "CreatedTime": "2019-12-09T23:28:11.110Z",
                "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEApOCAg88A8+f4dujn9Izt
\n26LxtgAkn2opGgo/NKpMiaisyw5qlg3f1gol7FV6pYNl78iJg3EO8JBbwtlH
+cR9\nLGSf60NDeVhm76Oc39Np/vWgOdsGQcRbi9WmKZeSODqjQGzVZWqPmito3FzWVk6b
\nfVY5N36U/RdbVAJm95Km+qaMYlbIdF40t72bi3IkKYV5hlB2XoDjlQ9F6ajQKyTB
\nMHa3SN8q+3ZjQ4sJJ7D1V6r4wR8jDcFVD5NckWJmmgIVnkOQM37NYeoDnkaOuTpu\nha/
+3b8tOb2z3LBVHPkp85zJRAOXacSwf5rZtPYKBNFsixTa2n55k2r218mOkMC4\nUwIDAQAB\n-----END
 PUBLIC KEY-----",
                "Comment": "example public key #2"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPublicKeys。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 CloudFront 分配的标签

以下示例获取 CloudFront 分配的标签列表：

```
aws cloudfront list-tags-for-resource \
    --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE
```

输出：

```
{
    "Tags": {
        "Items": [
            {
                "Key": "DateCreated",
                "Value": "2019-12-04"
            },
            {
                "Key": "Name",
                "Value": "Example name"
            },
            {
                "Key": "Project",
                "Value": "Example project"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`sign`

以下代码示例演示了如何使用 `sign`。

Amazon CLI

签署 CloudFront 网址

以下示例签署一个 CloudFront 网址。要签署网址，您需要密钥对 ID（在 Amazon 管理控制台中称为访问密钥 ID）和可信签署人的 CloudFront 密钥对的私钥。有关签署网址的更多信息，请参阅《Amazon CloudFront 开发人员指南》中的使用已签署网址和已签署 Cookie 提供私有内容。

```
aws cloudfront sign \
    --url https://d111111abcdef8.cloudfront.net/private-content/private-file.html \
    --key-pair-id APKAEIBAERJR2EXAMPLE \
    --private-key file://cf-signer-priv-key.pem \
    --date-less-than 2020-01-01
```

输出：

```
https://d111111abcdef8.cloudfront.net/private-content/private-
file.html?Expires=1577836800&Signature=nEXK7Kby47XKeZQKVc6pwkif6oZc-
JWSpDkH0UH7EBGGqvgurkecCbgL5VfUAXyLQuJxFwRQWscz-
owcq9KpmewCXrXQbPaJZNi9XSNwf4YKurPDQYaRQawKoeenH0GFteRf9ELK-
Bs3nljTLjtbgzIUt7QJNKXcWr8AuUYikzGdJ4-qzx6WnxXfH~fxg4-
GGl6l2kgCpXUB6Jx6K~Y3kpVOdzUPOIqFLHAnJojbhxqrVejomZZ2XrquDvNUCCIbePGnR3d24UPaLXG4FKOqNEaWDIB
GNvjRJxqWf93uMobeMOiVYahb-e0KItiQewGcm0eLZQ__&Key-Pair-Id=APKAEIBAERJR2EXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Sign。

tag-resource

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

标记 CloudFront 分配

以下 `tag-resource` 示例为指定的 CloudFront 分配添加两个标签。

```
aws cloudfront tag-resource \
    --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \
    --tags 'Items=[{Key=Name,Value="Example name"},{Key=Project,Value="Example
 project"}]'
```

您可以在 JSON 文件中提供标签，而不必使用命令行参数，如以下示例所示：

```
aws cloudfront tag-resource \
```

```
    --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \
    --tags file://tags.json
```

的内容tags.json：

```
{
    "Items": [
        {
            "Key": "Name",
            "Value": "Example name"
        },
        {
            "Key": "Project",
            "Value": "Example project"
        }
    ]
}
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从 CloudFront 分配中移除标签

以下示例使用命令行参数从 CloudFront 分配中移除两个标签：

```
aws cloudfront untag-resource \
    --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \
    --tag-keys Items=Name,Project
```

您可以在 JSON 文件中提供标签密钥，而不必使用命令行参数，如以下示例所示：

```
aws cloudfront untag-resource \
    --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \
    --tag-keys file://tag-keys.json
```

文件 tag-keys.json 是当前文件夹中包含以下内容的 JSON 文档：

```
{
    "Items": [
        "Name",
        "Project"
    ]
}
```

成功时，此命令没有输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-cloud-front-origin-access-identity

以下代码示例演示了如何使用 update-cloud-front-origin-access-identity。

Amazon CLI

更新 CloudFront 来源访问身份

以下示例更新 ID 为 E74FTE3AEXAMPLE 的来源访问身份（OAI）。您可以更新的唯一字段是 OAI 的 Comment。

要更新 OAI，您必须拥有 OAI 的 ID 和 ETag。OAI ID 将在 create-cloud-front-origin-access-identity 和 list-cloud-front-origin-access-identities 命令的输出中返回。要获取 ETag，请使用 get-cloud-front-origin-access-identity 或 get-cloud-front-origin-access-identity-config 命令。使用 --if-match 选项提供 OAI 的 ETag。

```
aws cloudfront update-cloud-front-origin-access-identity \
    --id E74FTE3AEXAMPLE \
    --if-match E2QWRUHEXAMPLE \
    --cloud-front-origin-access-identity-config \
        CallerReference=cli-example,Comment="Example OAI Updated"
```

您可以通过在 JSON 文件中提供 OAI 配置来完成同样的事情，如以下示例所示：

```
aws cloudfront update-cloud-front-origin-access-identity \
    --id E74FTE3AEXAMPLE \
    --if-match E2QWRUHEXAMPLE \
```

```
    --cloud-front-origin-access-identity-config file://OAI-config.json
```

文件 OAI-config.json 是当前目录中包含以下内容的 JSON 文档：

```
{
    "CallerReference": "cli-example",
    "Comment": "Example OAI Updated"
}
```

无论您使用命令行参数还是 JSON 文件提供 OAI 配置，输出都相同：

```
{
    "ETag": "E9LHASXEXAMPLE",
    "CloudFrontOriginAccessIdentity": {
        "Id": "E74FTE3AEXAMPLE",
        "S3CanonicalUserId":
 "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
        "CloudFrontOriginAccessIdentityConfig": {
            "CallerReference": "cli-example",
            "Comment": "Example OAI Updated"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCloudFrontOriginAccessIdentity。

update-connection-group

以下代码示例演示了如何使用 update-connection-group。

Amazon CLI

更新 CloudFront 连接组

以下 update-connection-group 示例禁用 CloudFront 连接组并禁用 IPv6。

```
aws cloudfront update-connection-group \
    --id cg_2yHsDkcPKeUlVkk3aEgLKcjABC \
    --no-ipv6-enabled \
    --no-enabled \
```

```
    --if-match E3UN6WX5RRO2ABC
```

输出：

```
{
    "ETag": "E1F83G8C2ARABC",
    "ConnectionGroup": {
        "Id": "cg_2yHsDkcPKeUlVkk3aEgLKcjABC",
        "Name": "cg-example",
        "Arn": "arn:aws:cloudfront::123456789012:connection-group/
cg_2yHsDkcPKeUlVkk3aEgLKcjABC",
        "CreatedTime": "2025-06-09T20:58:35.481000+00:00",
        "LastModifiedTime": "2025-06-11T16:25:54.280000+00:00",
        "Ipv6Enabled": false,
        "RoutingEndpoint": "du9xp1elo1abc.cloudfront.net",
        "Status": "InProgress",
        "Enabled": false,
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的创建自定义连接组（可选）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateConnectionGroup。

update-distribution-tenant

以下代码示例演示了如何使用 update-distribution-tenant。

Amazon CLI

更新 CloudFront 分配租户

以下 update-distribution-tenant 示例使用新的参数值更新 CloudFront 分配租户，并在地理限制中添加一个国家/地区。

```
aws cloudfront update-distribution-tenant \
    --cli-input-json file://update-tenant.json
```

的内容update-tenant.json：

```
{
    "Id": "dt_2yMvQgam3QkJo2z54FDl91dk1AB",
    "IfMatch": "E1F83G8C2ARABC",
    "Parameters": [
        {
            "Name": "testParam",
            "Value": "newParameterValue"
        }
    ],
    "Customizations": {
        "WebAcl": {
            "Action": "disable"
        },
        "GeoRestrictions": {
            "RestrictionType": "whitelist",
            "Locations": [
                "DE",
                "GB",
                "ES"
            ]
        }
    }
}
```

输出：

```
{
    "ETag": "E1PA6795UKMABC",
    "DistributionTenant": {
        "Id": "dt_2yMvQgam3QkJo2z54FDl91dk1AB",
        "DistributionId": "E1XNX8R2GOAABC",
        "Name": "new-tenant-customizations",
        "Arn": "arn:aws:cloudfront::123456789012:distribution-tenant/
dt_2yMvQgam3QkJo2z54FDl91dk1AB",
        "Domains": [
            {
                "Domain": "example.com",
                "Status": "active"
            }
        ],
        "Customizations": {
            "WebAcl": {
                "Action": "disable"
```

```
            },
            "GeoRestrictions": {
                "RestrictionType": "whitelist",
                "Locations": [
                    "DE",
                    "ES",
                    "GB"
                ]
            }
        },
        "Parameters": [
            {
                "Name": "testParam",
                "Value": "newParameterValue"
            }
        ],
        "ConnectionGroupId": "cg_2whCJoXMYCjHcxaLGrkllvyABC",
        "CreatedTime": "2025-06-11T15:54:02.142000+00:00",
        "LastModifiedTime": "2025-06-11T16:42:45.531000+00:00",
        "Enabled": false,
        "Status": "InProgress"
    }
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的分配租户自定义。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateDistributionTenant。

update-distribution

以下代码示例演示了如何使用 update-distribution。

Amazon CLI

示例 1：更新 CloudFront 分配的默认根对象

以下示例针对 ID 为 EDFDVBD6EXAMPLE 的 CloudFront 分配，将其默认根对象更新为 index.html。

```
aws cloudfront update-distribution \
    --id EDFDVBD6EXAMPLE \
    --default-root-object index.html
```

输出：

```
{
    "ETag": "E2QWRUHEXAMPLE",
    "Distribution": {
        "Id": "EDFDVBD6EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-12-06T18:55:39.870Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "6b10378d-49be-4c4b-a642-419ccaf8f3b5",
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "example-website",
                        "DomainName": "www.example.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "CustomOriginConfig": {
                            "HTTPPort": 80,
                            "HTTPSPort": 443,
                            "OriginProtocolPolicy": "match-viewer",
                            "OriginSslProtocols": {
                                "Quantity": 2,
                                "Items": [
                                    "SSLv3",
                                    "TLSv1"
                                ]
                            },
                            "OriginReadTimeout": 30,
                            "OriginKeepaliveTimeout": 5
```

```
                    }
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "example-website",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                },
                "Headers": {
                    "Quantity": 1,
                    "Items": [
                        "*"
                    ]
                },
                "QueryStringCacheKeys": {
                    "Quantity": 0
                }
            },
            "TrustedSigners": {
                "Enabled": false,
                "Quantity": 0
            },
            "ViewerProtocolPolicy": "allow-all",
            "MinTTL": 0,
            "AllowedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ],
                "CachedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ]
                }
            },
```

```
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": true,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http1.1",
            "IsIPV6Enabled": true
        }
    }
}
```

示例 2：更新 CloudFront 分配

以下示例通过在名为 dist-config-disable.json 的 JSON 文件中提供分配配置来禁用 ID 为 EMLARXS9EXAMPLE 的 CloudFront 分配。要更新分配，您必须使用 --if-match 选项来提供分配的 ETag。要获取 ETag，请使用 get-distribution 或 get-distribution-config 命令。请注意，在 JSON 文件中，Enabled 字段设置为 false。

使用以下示例禁用分配后，您可以使用 delete-distribute 命令将其删除。

```
aws cloudfront update-distribution \
    --id EMLARXS9EXAMPLE \
    --if-match E2QWRUHEXAMPLE \
    --distribution-config file://dist-config-disable.json
```

的内容dist-config-disable.json：

```
{
    "CallerReference": "cli-1574382155-496510",
    "Aliases": {
        "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-1574382155-273939",
                "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                "OriginPath": "",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
    "OriginGroups": {
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-1574382155-273939",
        "ForwardedValues": {
            "QueryString": false,
```

```
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            },
            "QueryStringCacheKeys": {
                "Quantity": 0
            }
        },
        "TrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ViewerProtocolPolicy": "allow-all",
        "MinTTL": 0,
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {
                "Quantity": 2,
                "Items": [
                    "HEAD",
                    "GET"
                ]
            }
        },
        "SmoothStreaming": false,
        "DefaultTTL": 86400,
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
            "Quantity": 0
        },
        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
```

```
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": false,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
```

输出：

```
{
    "ETag": "E9LHASXEXAMPLE",
    "Distribution": {
        "Id": "EMLARXS9EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-12-06T18:32:35.553Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {
            "CallerReference": "cli-1574382155-496510",
```

```
            "Aliases": {
                "Quantity": 0
            },
            "DefaultRootObject": "index.html",
            "Origins": {
                "Quantity": 1,
                "Items": [
                    {
                        "Id": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
                        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
                        "OriginPath": "",
                        "CustomHeaders": {
                            "Quantity": 0
                        },
                        "S3OriginConfig": {
                            "OriginAccessIdentity": ""
                        }
                    }
                ]
            },
            "OriginGroups": {
                "Quantity": 0
            },
            "DefaultCacheBehavior": {
                "TargetOriginId": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
                "ForwardedValues": {
                    "QueryString": false,
                    "Cookies": {
                        "Forward": "none"
                    },
                    "Headers": {
                        "Quantity": 0
                    },
                    "QueryStringCacheKeys": {
                        "Quantity": 0
                    }
                },
                "TrustedSigners": {
                    "Enabled": false,
                    "Quantity": 0
                },
                "ViewerProtocolPolicy": "allow-all",
```

```
                "MinTTL": 0,
                "AllowedMethods": {
                    "Quantity": 2,
                    "Items": [
                        "HEAD",
                        "GET"
                    ],
                    "CachedMethods": {
                        "Quantity": 2,
                        "Items": [
                            "HEAD",
                            "GET"
                        ]
                    }
                },
                "SmoothStreaming": false,
                "DefaultTTL": 86400,
                "MaxTTL": 31536000,
                "Compress": false,
                "LambdaFunctionAssociations": {
                    "Quantity": 0
                },
                "FieldLevelEncryptionId": ""
            },
            "CacheBehaviors": {
                "Quantity": 0
            },
            "CustomErrorResponses": {
                "Quantity": 0
            },
            "Comment": "",
            "Logging": {
                "Enabled": false,
                "IncludeCookies": false,
                "Bucket": "",
                "Prefix": ""
            },
            "PriceClass": "PriceClass_All",
            "Enabled": false,
            "ViewerCertificate": {
                "CloudFrontDefaultCertificate": true,
                "MinimumProtocolVersion": "TLSv1",
                "CertificateSource": "cloudfront"
            },
```

```
            "Restrictions": {
                "GeoRestriction": {
                    "RestrictionType": "none",
                    "Quantity": 0
                }
            },
            "WebACLId": "",
            "HttpVersion": "http2",
            "IsIPV6Enabled": true
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDistribution。

update-domain-association

以下代码示例演示了如何使用 update-domain-association。

Amazon CLI

更新域关联

以下 update-domain-association 示例更新具有 ETag E23ZP02F085ABC 的分配租户的域关联。

```
aws cloudfront update-domain-association \
    --domain example.com \
    --target-resource DistributionTenantId=dt_2x9GhoK0TZRsohWzv1b9It8J1AB \
    --if-match E23ZP02F085ABC
```

输出：

```
{
    "ETag": "ETVPDKIKX0ABC",
    "Domain": "example.com",
    "ResourceId": "dt_2x9GhoK0TZRsohWzv1b9It8J1AB"
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的将备用域名移动到其他分配。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateDomainAssociation。

update-field-level-encryption-config

以下代码示例演示了如何使用 update-field-level-encryption-config。

Amazon CLI

更新 CloudFront 字段级加密配置

以下示例通过在 JSON 文件中提供参数，使用 ID C3KM2WVD605UAY 更新字段级加密配置的 Comment 字段。

要更新字段级加密配置，您必须拥有该配置的 ID 和 ETag。该 ID 将在 create-field-level-encryption-config 和 list-field-level-encryption-configs 命令的输出中返回。要获取 ETag，请使用 get-field-level-encryption 或 get-field-level-encryption-config 命令。使用 --if-match 选项提供配置的 ETag。

```
aws cloudfront update-field-level-encryption-config \
    --id C3KM2WVD605UAY \
    --if-match E2P4Z4VU7TY5SG \
    --field-level-encryption-config file://fle-config.json
```

文件 fle-config.json 是当前目录中包含以下内容的 JSON 文档：

```
{
    "CallerReference": "cli-example",
    "Comment": "Updated example FLE configuration",
    "QueryArgProfileConfig": {
        "ForwardWhenQueryArgProfileIsUnknown": true,
        "QueryArgProfiles": {
            "Quantity": 0
        }
    },
    "ContentTypeProfileConfig": {
        "ForwardWhenContentTypeIsUnknown": true,
        "ContentTypeProfiles": {
            "Quantity": 1,
            "Items": [
                {
```

```
                    "Format": "URLEncoded",
                    "ProfileId": "P280MFCLSYOCVU",
                    "ContentType": "application/x-www-form-urlencoded"
                }
            ]
        }
    }
}
```

输出：

```
{
    "ETag": "E26M4BIAV81ZF6",
    "FieldLevelEncryption": {
        "Id": "C3KM2WVD605UAY",
        "LastModifiedTime": "2019-12-10T22:26:26.170Z",
        "FieldLevelEncryptionConfig": {
            "CallerReference": "cli-example",
            "Comment": "Updated example FLE configuration",
            "QueryArgProfileConfig": {
                "ForwardWhenQueryArgProfileIsUnknown": true,
                "QueryArgProfiles": {
                    "Quantity": 0,
                    "Items": []
                }
            },
            "ContentTypeProfileConfig": {
                "ForwardWhenContentTypeIsUnknown": true,
                "ContentTypeProfiles": {
                    "Quantity": 1,
                    "Items": [
                        {
                            "Format": "URLEncoded",
                            "ProfileId": "P280MFCLSYOCVU",
                            "ContentType": "application/x-www-form-urlencoded"
                        }
                    ]
                }
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFieldLevelEncryptionConfig。

update-field-level-encryption-profile

以下代码示例演示了如何使用 update-field-level-encryption-profile。

Amazon CLI

更新 CloudFront 字段级加密配置文件

以下示例更新 ID 为 PPK0UOSIF5WSV 的字段级加密配置文件。此示例通过在 JSON 文件中提供参数，更新配置文件的 Name 和 Comment，并添加第二个 FieldPatterns 项目。

要更新字段级加密配置文件，必须提供配置文件的 ID 和 ETag。该 ID 将在 create-field-level-encryption-profile 和 list-field-level-encryption-profiles 命令的输出中返回。要获取 ETag，请使用 get-field-level-encryption-profile 或 get-field-level-encryption-profile-config 命令。使用 --if-match 选项提供配置文件的 ETag。

```
aws cloudfront update-field-level-encryption-profile \
    --id PPK0UOSIF5WSV \
    --if-match E1QQG65FS2L2GC \
    --field-level-encryption-profile-config file://fle-profile-config.json
```

文件 fle-profile-config.json 是当前目录中包含以下内容的 JSON 文档：

```
{
    "Name": "ExampleFLEProfileUpdated",
    "CallerReference": "cli-example",
    "Comment": "Updated FLE profile for AWS CLI example",
    "EncryptionEntities": {
        "Quantity": 1,
        "Items": [
            {
                "PublicKeyId": "K2K8NC4HVFE3M0",
                "ProviderId": "ExampleFLEProvider",
                "FieldPatterns": {
                    "Quantity": 2,
                    "Items": [
                        "ExampleSensitiveField",
                        "SecondExampleSensitiveField"
```

```
                    ]
                }
            }
        ]
    }
}
```

输出：

```
{
    "ETag": "EJETYFJ9CL66D",
    "FieldLevelEncryptionProfile": {
        "Id": "PPK0UOSIF5WSV",
        "LastModifiedTime": "2019-12-10T19:05:58.296Z",
        "FieldLevelEncryptionProfileConfig": {
            "Name": "ExampleFLEProfileUpdated",
            "CallerReference": "cli-example",
            "Comment": "Updated FLE profile for AWS CLI example",
            "EncryptionEntities": {
                "Quantity": 1,
                "Items": [
                    {
                        "PublicKeyId": "K2K8NC4HVFE3M0",
                        "ProviderId": "ExampleFLEProvider",
                        "FieldPatterns": {
                            "Quantity": 2,
                            "Items": [
                                "ExampleSensitiveField",
                                "SecondExampleSensitiveField"
                            ]
                        }
                    }
                ]
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFieldLevelEncryptionProfile。

verify-dns-configuration

以下代码示例演示了如何使用 verify-dns-configuration。

Amazon CLI

验证域的 DNS 配置

以下 verify-dns-configuration 示例验证域的 DNS 配置。

```
aws cloudfront verify-dns-configuration \
    --domain example.com \
    --identifier dt_2x9GhoK0TZRsohWzv1b9It8J1AB
```

输出：

```
{
    "DnsConfigurationList": [
        {
            "Domain": "example.com",
            "Status": "valid-configuration"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudFront 开发人员指南》中的将备用域名移动到其他分配。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 VerifyDnsConfiguration。

使用 Amazon CLI 的 Amazon CloudSearch 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon CloudSearch 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

upload-documents

以下代码示例演示了如何使用 upload-documents。

Amazon CLI

以下 upload-documents 命令会将一批 JSON 文档上传到 Amazon CloudSearch 域：

```
aws cloudsearchdomain upload-documents --endpoint-url https://doc-my-domain.us-west-1.cloudsearch.amazonaws.com --content-type application/json --documents document-batch.json
```

输出：

```
{
  "status": "success",
  "adds": 5000,
  "deletes": 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadDocuments。

使用 Amazon CLI 的 CloudTrail 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CloudTrail 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-tags

以下代码示例演示了如何使用 add-tags。

Amazon CLI

为跟踪添加标签

以下 add-tags 命令为 Trail1 添加标签：

```
aws cloudtrail add-tags --resource-id arn:aws:cloudtrail:us-
east-1:123456789012:trail/Trail1 --tags-
list Key=name,Value=Alice Key=location,Value=us
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTags。

create-subscription

以下代码示例演示了如何使用 create-subscription。

Amazon CLI

为跟踪创建和配置 Amazon 资源

以下 create-subscription 命令为 Trail1 创建新的 S3 存储桶和 SNS 主题。

```
aws cloudtrail create-subscription \
    --name Trail1 \
    --s3-new-bucket amzn-s3-demo-bucket \
    --sns-new-topic my-topic
```

输出：

```
Setting up new S3 bucket amzn-s3-demo-bucket...
Setting up new SNS topic my-topic...
Creating/updating CloudTrail configuration...
CloudTrail configuration:
    {
        "trailList": [
            {
```

```
                "IncludeGlobalServiceEvents": true,
                "Name": "Trail1",
                "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/
Trail1",
                "LogFileValidationEnabled": false,
                "IsMultiRegionTrail": false,
                "S3BucketName": "amzn-s3-demo-bucket",
                "SnsTopicName": "my-topic",
                "HomeRegion": "us-east-1"
            }
        ],
        "ResponseMetadata": {
        "HTTPStatusCode": 200,
        "RequestId": "f39e51f6-c615-11e5-85bd-d35ca21ee3e2"
        }
    }
Starting CloudTrail service...
Logs will be delivered to my-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscription。

create-trail

以下代码示例演示了如何使用 create-trail。

Amazon CLI

创建跟踪

以下 create-trail 示例创建名为 Trail1 的多区域跟踪并指定一个 S3 存储桶。

```
aws cloudtrail create-trail \
    --name Trail1 \
    --s3-bucket-name amzn-s3-demo-bucket \
    --is-multi-region-trail
```

输出：

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "Trail1",
    "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Trail1",
```

```
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "S3BucketName": "amzn-s3-demo-bucket"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTrail。

delete-trail

以下代码示例演示了如何使用 delete-trail。

Amazon CLI

删除跟踪

以下 delete-trail 命令删除名为 Trail1 的跟踪：

```
aws cloudtrail delete-trail --name Trail1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTrail。

describe-trails

以下代码示例演示了如何使用 describe-trails。

Amazon CLI

描述跟踪

以下 describe-trails 示例返回 Trail1 和 Trail2 的设置。

```
aws cloudtrail describe-trails \
    --trail-name-list Trail1 Trail2
```

输出：

```
{
    "trailList": [
        {
            "IncludeGlobalServiceEvents": true,
            "Name": "Trail1",
            "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
```

```
            "LogFileValidationEnabled": false,
            "IsMultiRegionTrail": false,
            "S3BucketName": "amzn-s3-demo-bucket",
            "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/
CloudTrail_CloudWatchLogs_Role",
            "CloudWatchLogsLogGroupArn": "arn:aws:logs:us-east-1:123456789012:log-
group:CloudTrail:*",
            "SnsTopicName": "my-topic",
            "HomeRegion": "us-east-1"
        },
        {
            "IncludeGlobalServiceEvents": true,
            "Name": "Trail2",
            "S3KeyPrefix": "my-prefix",
            "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail2",
            "LogFileValidationEnabled": false,
            "IsMultiRegionTrail": false,
            "S3BucketName": "amzn-s3-demo-bucket2",
            "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/4c5ae5ac-3c13-421e-8335-c7868ef6a769",
            "HomeRegion": "us-east-1"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrails。

get-event-selectors

以下代码示例演示了如何使用 get-event-selectors。

Amazon CLI

查看跟踪的事件选择器设置

以下 get-event-selectors 命令返回 Trail1 的设置：

```
aws cloudtrail get-event-selectors --trail-name Trail1
```

输出：

```
{
```

```
  "EventSelectors": [
      {
          "IncludeManagementEvents": true,
          "DataResources": [],
          "ReadWriteType": "All"
      }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEventSelectors。

get-trail-status

以下代码示例演示了如何使用 get-trail-status。

Amazon CLI

获取跟踪的状态

以下 get-trail-status 命令返回 Trail1 的交付和日志记录的详细信息：

```
aws cloudtrail get-trail-status --name Trail1
```

输出：

```
{
  "LatestNotificationTime": 1454022144.869,
  "LatestNotificationAttemptSucceeded": "2016-01-28T23:02:24Z",
  "LatestDeliveryAttemptTime": "2016-01-28T23:02:24Z",
  "LatestDeliveryTime": 1454022144.869,
  "TimeLoggingStarted": "2015-11-06T18:36:38Z",
  "LatestDeliveryAttemptSucceeded": "2016-01-28T23:02:24Z",
  "IsLogging": true,
  "LatestCloudWatchLogsDeliveryTime": 1454022144.918,
  "StartLoggingTime": 1446834998.695,
  "StopLoggingTime": 1446834996.933,
  "LatestNotificationAttemptTime": "2016-01-28T23:02:24Z",
  "TimeLoggingStopped": "2015-11-06T18:36:36Z"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTrailStatus。

list-public-keys

以下代码示例演示了如何使用 list-public-keys。

Amazon CLI

列出跟踪的所有公钥

以下 list-public-keys 命令返回与用于在指定时间范围对摘要文件进行签名的私钥对应的所有公钥。

```
aws cloudtrail list-public-keys --start-time 2016-01-01T20:30:00.000Z
```

输出：

```
{
  "PublicKeyList": [
      {
         "ValidityStartTime": 1453076702.0,
         "ValidityEndTime": 1455668702.0,
         "Value": "MIIBCgKCAQEAlSS3cl92HDycr/MTj0moOhas8habjrraXw+KzlWF0axSI2tcF
+3iJ9BKQAVSKxGwxwu3m0wG3J
+kUl1xboEcEPHYoIYMbgfSw7KGnuDKwkLzsQWhUJ0cIbOHASox1vv/5fNXkrHhGbDCHeVXm804c83nvHUEFYThr1PfyP
+4WGDk+BGH5m9iuiAKkipEHWmUl8/P7XpfpWQuk4h8g3pXZOrNXr08lbh4d39svj7UqdhvOXoBISp9t/
EXYuePGEtBdrKD9Dz+VHwyUPtBQvYr9BnkF88qBnaPNhS44rzwIDAQAB",
         "Fingerprint": "7f3f401420072e50a65a141430817ab3"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPublicKeys。

list-tags

以下代码示例演示了如何使用 list-tags。

Amazon CLI

列出跟踪的标签

以下 list-tags 命令列出 Trail1 和 Trail2 的标签：

```
aws cloudtrail list-tags --resource-id-list arn:aws:cloudtrail:us-
east-1:123456789012:trail/Trail1 arn:aws:cloudtrail:us-east-1:123456789012:trail/
Trail2
```

输出：

```
{
 "ResourceTagList": [
     {
         "ResourceId": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
         "TagsList": [
             {
                 "Value": "Alice",
                 "Key": "name"
             },
             {
                 "Value": "us",
                 "Key": "location"
             }
         ]
     },
     {
         "ResourceId": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail2",
         "TagsList": [
             {
                 "Value": "Bob",
                 "Key": "name"
             }
         ]
     }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTags。

`lookup-events`

以下代码示例演示了如何使用 `lookup-events`。

Amazon CLI

查找跟踪的事件

以下 lookup-events 命令按属性 EventName 查找 API 活动事件：

```
aws cloudtrail lookup-events --lookup-
attributes AttributeKey=EventName,AttributeValue=ConsoleLogin
```

输出：

```
{
  "Events": [
     {
         "EventId": "654ccbc0-ba0d-486a-9076-dbf7274677a7",
         "Username": "my-session-name",
         "EventTime": "2021-11-18T09:41:02-08:00",
         "CloudTrailEvent": "{\"eventVersion\":\"1.02\",\"userIdentity\":{\"type\":
\"AssumedRole\",\"principalId\":\"AROAJIKPFTA72SWU4L7T4:my-session-name\",\"arn\":
\"arn:aws:sts::123456789012:assumed-role/my-role/my-session-name\",\"accountId\":
\"123456789012\",\"sessionContext\":{\"attributes\":{\"mfaAuthenticated\":\"false
\",\"creationDate\":\"2016-01-26T21:42:12Z\"},\"sessionIssuer\":{\"type\":\"Role\",
\"principalId\":\"AROAJIKPFTA72SWU4L7T4\",\"arn\":\"arn:aws:iam::123456789012:role/
my-role\",\"accountId\":\"123456789012\",\"userName\":\"my-role\"}}},\"eventTime
\":\"2016-01-26T21:42:12Z\",\"eventSource\":\"signin.amazonaws.com\",\"eventName\":
\"ConsoleLogin\",\"awsRegion\":\"us-east-1\",\"sourceIPAddress\":\"72.21.198.70\",
\"userAgent\":\"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_5) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/47.0.2526.111 Safari/537.36\",\"requestParameters
\":null,\"responseElements\":{\"ConsoleLogin\":\"Success\"},\"additionalEventData\":
{\"MobileVersion\":\"No\",\"MFAUsed\":\"No\"},\"eventID\":\"654ccbc0-ba0d-486a-9076-
dbf7274677a7\",\"eventType\":\"AwsConsoleSignIn\",\"recipientAccountId\":
\"123456789012\"}",
         "EventName": "ConsoleLogin",
         "Resources": []
     }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 LookupEvents。

put-event-selectors

以下代码示例演示了如何使用 put-event-selectors。

Amazon CLI

示例 1：使用高级事件选择器配置跟踪以记录管理事件和数据事件

您可以为高级事件选择器添加高级事件选择器和条件，一个跟踪的所有条件和选择器最多可添加 500 个值。您可以使用高级事件选择器记录所有可用的数据事件类型。您可以使用高级事件选择器或基本事件选择器，但不能同时使用两者。如果将高级事件选择器应用于跟踪，则所有现有的基本事件选择器都将被覆盖。

以下 `put-event-selectors` 示例为名为 `myTrail` 的跟踪创建高级事件选择器，用于记录所有管理事件，记录除一个 S3 存储桶之外的所有 S3 PutObject 和 DeleteObject API 调用，记录名为 `myFunction` 的 Lambda 函数的数据 API 调用，以及记录名为 `myTopic` 的 SNS 主题的 Publish API 调用。

```
aws cloudtrail put-event-selectors \
    --trail-name myTrail \
    --advanced-event-selectors '[{"Name": "Log all management events",
 "FieldSelectors": [{ "Field": "eventCategory", "Equals": ["Management"] }] },
{"Name": "Log PutObject and DeleteObject events for all but one
 bucket","FieldSelectors": [{ "Field": "eventCategory", "Equals": ["Data"] },
{ "Field": "resources.type", "Equals": ["AWS::S3::Object"] },{ "Field":
 "eventName", "Equals": ["PutObject","DeleteObject"] },{ "Field": "resources.ARN",
 "NotStartsWith": ["arn:aws:s3:::amzn-s3-demo-bucket/"] }]},{"Name": "Log
 data events for a specific Lambda function","FieldSelectors": [{ "Field":
 "eventCategory", "Equals": ["Data"] },{ "Field": "resources.type",
 "Equals": ["AWS::Lambda::Function"] },{ "Field": "resources.ARN", "Equals":
 ["arn:aws:lambda:us-east-1:123456789012:function:myFunction"] }]},{"Name":
 "Log all Publish API calls on a specific SNS topic","FieldSelectors":
 [{ "Field": "eventCategory", "Equals": ["Data"] },{ "Field": "resources.type",
 "Equals": ["AWS::SNS::Topic"] },{ "Field": "eventName", "Equals":
 ["Publish"] },{ "Field": "resources.ARN", "Equals": ["arn:aws:sns:us-
east-1:123456789012:myTopic.fifo"] }]}]'
```

输出：

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/myTrail",
    "AdvancedEventSelectors": [
        {
            "Name": "Log all management events",
            "FieldSelectors": [
                {
```

```
                "Field": "eventCategory",
                "Equals": [
                    "Management"
                ]
            }
        ]
    },
    {
        "Name": "Log PutObject and DeleteObject events for all but one bucket",
        "FieldSelectors": [
            {
                "Field": "eventCategory",
                "Equals": [
                    "Data"
                ]
            },
            {
                "Field": "resources.type",
                "Equals": [
                    "AWS::S3::Object"
                ]
            },
            {
                "Field": "eventName",
                "Equals": [
                    "PutObject",
                    "DeleteObject"
                ]
            },
            {
                "Field": "resources.ARN",
                "NotStartsWith": [
                    "arn:aws:s3:::amzn-s3-demo-bucket/"
                ]
            }
        ]
    },
    {
        "Name": "Log data events for a specific Lambda function",
        "FieldSelectors": [
            {
                "Field": "eventCategory",
                "Equals": [
                    "Data"
```

```
                ]
            },
            {
                "Field": "resources.type",
                "Equals": [
                    "AWS::Lambda::Function"
                ]
            },
            {
                "Field": "resources.ARN",
                "Equals": [
                    "arn:aws:lambda:us-east-1:123456789012:function:myFunction"
                ]
            }
        ]
    },
    {
        "Name": "Log all Publish API calls on a specific SNS topic",
        "FieldSelectors": [
            {
                "Field": "eventCategory",
                "Equals": [
                    "Data"
                ]
            },
            {
                "Field": "resources.type",
                "Equals": [
                    "AWS::SNS::Topic"
                ]
            },
            {
                "Field": "eventName",
                "Equals": [
                    "Publish"
                ]
            },
            {
                "Field": "resources.ARN",
                "Equals": [
                    "arn:aws:sns:us-east-1:123456789012:myTopic.fifo"
                ]
            }
        ]
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudTrail 用户指南》中的使用高级事件选择器记录事件。

示例 2：为跟踪配置事件选择器以记录所有管理事件和数据事件

您可以为一个跟踪配置最多 5 个事件选择器和最多 250 个数据资源。事件选择器也称为基本事件选择器。您可以使用事件选择器记录 S3 对象、Lambda 函数和 DynnamoDB 表的管理事件和数据事件。要记录其他资源类型的数据事件，您必须使用高级事件选择器。

以下 `put-event-selectors` 示例为名为 `TrailName` 的跟踪创建一个事件选择器，以包括所有管理事件、两个 Amazon S3 存储桶/前缀组合的数据事件以及单个名为 `hello-world-python-function` 的 Amazon Lambda 函数的数据事件。

```
aws cloudtrail put-event-selectors \
    --trail-name TrailName \
    --event-selectors '[{"ReadWriteType": "All","IncludeManagementEvents":
 true,"DataResources": [{"Type":"AWS::S3::Object", "Values": ["arn:aws:s3:::amzn-
s3-demo-bucket/prefix","arn:aws:s3:::amzn-s3-demo-bucket2/prefix2"]},
{"Type": "AWS::Lambda::Function","Values": ["arn:aws:lambda:us-
west-2:999999999999:function:hello-world-python-function"]}]}]'
```

输出：

```
{
    "EventSelectors": [
        {
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Values": [
                        "arn:aws:s3:::amzn-s3-demo-bucket/prefix",
                        "arn:aws:s3:::amzn-s3-demo-bucket2/prefix2"
                    ],
                    "Type": "AWS::S3::Object"
                },
                {
                    "Values": [
                        "arn:aws:lambda:us-west-2:123456789012:function:hello-world-
python-function"
                    ],
```

```
                    "Type": "AWS::Lambda::Function"
                },
            ],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

有关更多信息，请参阅《Amazon CloudTrail 用户指南》中的使用基本事件选择器记录事件。

示例 3：为跟踪配置事件选择器以记录管理事件、S3 对象上的所有 S3 数据事件以及您账户中函数上的所有 Lambda 数据事件

以下 `put-event-selectors` 示例为名为 `TrailName2` 的跟踪创建一个事件选择器，包括所有管理事件以及 Amazon 账户中所有 Amazon S3 存储桶和 Amazon Lambda 函数的所有数据事件。

```
aws cloudtrail put-event-selectors \
    --trail-name TrailName2 \
    --event-selectors '[{"ReadWriteType": "All","IncludeManagementEvents":
 true,"DataResources": [{"Type":"AWS::S3::Object", "Values": ["arn:aws:s3"]},
{"Type": "AWS::Lambda::Function","Values": ["arn:aws:lambda"]}]}]'
```

输出：

```
{
    "EventSelectors": [
        {
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Values": [
                        "arn:aws:s3"
                    ],
                    "Type": "AWS::S3::Object"
                },
                {
                    "Values": [
                        "arn:aws:lambda"
                    ],
                    "Type": "AWS::Lambda::Function"
                },
            ],
```

```
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName2"
}
```

有关更多信息，请参阅《Amazon CloudTrail 用户指南》中的使用基本事件选择器记录事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutEventSelectors。

`remove-tags`

以下代码示例演示了如何使用 `remove-tags`。

Amazon CLI

移除跟踪的标签

以下 `remove-tags` 命令移除 `Trail1` 的指定标签：

```
aws cloudtrail remove-tags --resource-id arn:aws:cloudtrail:us-
east-1:123456789012:trail/Trail1 --tags-list Key=name Key=location
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTags。

`start-logging`

以下代码示例演示了如何使用 `start-logging`。

Amazon CLI

为跟踪启动日志记录

以下 `start-logging` 命令为 `Trail1` 开启日志记录：

```
aws cloudtrail start-logging --name Trail1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartLogging。

`stop-logging`

以下代码示例演示了如何使用 `stop-logging`。

Amazon CLI

停止跟踪的日志记录

以下 stop-logging 命令为 Trail1 关闭日志记录：

```
aws cloudtrail stop-logging --name Trail1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopLogging。

update-subscription

以下代码示例演示了如何使用 update-subscription。

Amazon CLI

更新跟踪的配置设置

以下 update-subscription 示例更新跟踪以指定新的 S3 存储桶和 SNS 主题。

```
aws cloudtrail update-subscription \
    --name Trail1 \
    --s3-new-bucket amzn-s3-demo-bucket \
    --sns-new-topic my-topic-new
```

输出：

```
Setting up new S3 bucket amzn-s3-demo-bucket...
Setting up new SNS topic my-topic-new...
Creating/updating CloudTrail configuration...
CloudTrail configuration:
{
    "trailList": [
        {
            "IncludeGlobalServiceEvents": true,
            "Name": "Trail1",
            "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
            "LogFileValidationEnabled": false,
            "IsMultiRegionTrail": false,
            "S3BucketName": "amzn-s3-demo-bucket",
            "SnsTopicName": "my-topic-new",
```

```
            "HomeRegion": "us-east-1"
        }
    ],
    "ResponseMetadata": {
        "HTTPStatusCode": 200,
        "RequestId": "31126f8a-c616-11e5-9cc6-2fd637936879"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscription。

update-trail

以下代码示例演示了如何使用 update-trail。

Amazon CLI

更新跟踪

以下 update-trail 示例更新跟踪以使用现有存储桶进行日志传输。

```
aws cloudtrail update-trail \
    --name Trail1 \
    --s3-bucket-name amzn-s3-demo-bucket
```

输出：

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "Trail1",
    "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Trail1",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "S3BucketName": "amzn-s3-demo-bucket"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTrail。

validate-logs

以下代码示例演示了如何使用 validate-logs。

Amazon CLI

验证日志文件

以下 validate-logs 命令验证 Trail1 的日志：

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-
east-1:123456789012:trail/Trail1 --start-time 20160129T19:00:00Z
```

输出：

```
Validating log files for trail arn:aws:cloudtrail:us-east-1:123456789012:trail/
Trail1 between 2016-01-29T19:00:00Z and 2016-01-29T22:15:43Z
Results requested for 2016-01-29T19:00:00Z to 2016-01-29T22:15:43Z
Results found for 2016-01-29T19:24:57Z to 2016-01-29T21:24:57Z:
3/3 digest files valid
15/15 log files valid
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateLogs。

使用 Amazon CLI 的 CloudWatch 示例

以下代码示例演示了如何通过将 Amazon Command Line Interface与 CloudWatch 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

delete-alarms

以下代码示例演示了如何使用 delete-alarms。

Amazon CLI

删除警报

以下示例使用 delete-alarms 命令删除名为“myalarm”的 Amazon CloudWatch 警报：

```
aws cloudwatch delete-alarms --alarm-names myalarm
```

输出：

```
This command returns to the prompt if successful.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAlarms。

delete-anomaly-detector

以下代码示例演示了如何使用 delete-anomaly-detector。

Amazon CLI

删除指定的异常检测模型

以下 delete-anomaly-detector 示例删除了指定账户中的异常检测器模型。

```
aws cloudwatch delete-anomaly-detector \
    --namespace AWS/Logs \
    --metric-name IncomingBytes \
    --stat SampleCount
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的删除异常检测模型。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteAnomalyDetector。

delete-dashboards

以下代码示例演示了如何使用 delete-dashboards。

Amazon CLI

删除指定的控制面板

以下 delete-dashboards 示例删除了指定账户中名为 Dashboard-A 和 Dashboard-B 的两个控制面板。

```
aws cloudwatch delete-dashboards \
    --dashboard-names Dashboard-A Dashboard-B
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Amazon CloudWatch 控制面板。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteDashboards。

delete-insight-rules

以下代码示例演示了如何使用 delete-insight-rules。

Amazon CLI

删除指定的 Contributor Insights 规则

以下 delete-insight-rules 示例删除指定账户中两个分别名为 Rule-A 和 Rule-B 的 Contributor Insights 规则。

```
aws cloudwatch delete-insight-rules \
    --rule-names Rule-A Rule-B
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Contributor Insights 分析高基数数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteInsightRules。

delete-metric-stream

以下代码示例演示了如何使用 delete-metric-stream。

Amazon CLI

删除指定的指标流

以下 delete-metric-stream 示例删除指定账户中名为 QuickPartial-gSCKv0 的指标流。

```
aws cloudwatch delete-metric-stream \
    --name QuickPartial-gSCKv0
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteMetricStream。

describe-alarm-history

以下代码示例演示了如何使用 describe-alarm-history。

Amazon CLI

检索警报的历史记录

以下示例使用 describe-alarm-history 命令检索名为“myalarm”的 Amazon CloudWatch 警报的历史记录：

```
aws cloudwatch describe-alarm-history --alarm-name "myalarm" --history-item-
type StateUpdate
```

输出：

```
{
    "AlarmHistoryItems": [
        {
            "Timestamp": "2014-04-09T18:59:06.442Z",
            "HistoryItemType": "StateUpdate",
```

```
            "AlarmName": "myalarm",
            "HistoryData": "{\"version\":\"1.0\",\"oldState\":{\"stateValue\":
\"ALARM\",\"stateReason\":\"testing purposes\"},\"newState\":{\"stateValue\":\"OK
\",\"stateReason\":\"Threshold Crossed: 2 datapoints were not greater than the
 threshold (70.0). The most recent datapoints: [38.958, 40.292].\",\"stateReasonData
\":{\"version\":\"1.0\",\"queryDate\":\"2014-04-09T18:59:06.419+0000\",\"startDate
\":\"2014-04-09T18:44:00.000+0000\",\"statistic\":\"Average\",\"period\":300,
\"recentDatapoints\":[38.958,40.292],\"threshold\":70.0}}}",
            "HistorySummary": "Alarm updated from ALARM to OK"
        },
        {
            "Timestamp": "2014-04-09T18:59:05.805Z",
            "HistoryItemType": "StateUpdate",
            "AlarmName": "myalarm",
            "HistoryData": "{\"version\":\"1.0\",\"oldState\":{\"stateValue
\":\"OK\",\"stateReason\":\"Threshold Crossed: 2 datapoints were
 not greater than the threshold (70.0). The most recent datapoints:
 [38.839999999999996, 39.714].\",\"stateReasonData\":{\"version\":
\"1.0\",\"queryDate\":\"2014-03-11T22:45:41.569+0000\",\"startDate\":
\"2014-03-11T22:30:00.000+0000\",\"statistic\":\"Average\",\"period\":300,
\"recentDatapoints\":[38.839999999999996,39.714],\"threshold\":70.0}},\"newState\":
{\"stateValue\":\"ALARM\",\"stateReason\":\"testing purposes\"}}",
            "HistorySummary": "Alarm updated from OK to ALARM"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAlarmHistory。

describe-alarms-for-metric

以下代码示例演示了如何使用 describe-alarms-for-metric。

Amazon CLI

显示与指标关联的警报的相关信息

以下示例使用 describe-alarms-for-metric 命令显示与 Amazon EC2 CPUUtilization 指标和 ID 为 i-0c986c72 的实例关联的任何警报的相关信息：

```
aws cloudwatch describe-alarms-for-metric --metric-name CPUUtilization --
namespace AWS/EC2 --dimensions Name=InstanceId,Value=i-0c986c72
```

输出：


```
{
    "MetricAlarms": [
        {
            "EvaluationPeriods": 10,
            "AlarmArn": "arn:aws:cloudwatch:us-
east-1:111122223333:alarm:myHighCpuAlarm2",
            "StateUpdatedTimestamp": "2013-10-30T03:03:51.479Z",
            "AlarmConfigurationUpdatedTimestamp": "2013-10-30T03:03:50.865Z",
            "ComparisonOperator": "GreaterThanOrEqualToThreshold",
            "AlarmActions": [
                "arn:aws:sns:us-east-1:111122223333:NotifyMe"
            ],
            "Namespace": "AWS/EC2",
            "AlarmDescription": "CPU usage exceeds 70 percent",
            "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":
\"2013-10-30T03:03:51.479+0000\",\"startDate\":\"2013-10-30T02:08:00.000+0000\",
\"statistic\":\"Average\",\"period\":300,\"recentDatapoints\":
[40.698,39.612,42.432,39.796,38.816,42.28,42.854,40.088,40.760000000000005,41.316],
\"threshold\":70.0}",
            "Period": 300,
            "StateValue": "OK",
            "Threshold": 70.0,
            "AlarmName": "myHighCpuAlarm2",
            "Dimensions": [
                {
                    "Name": "InstanceId",
                    "Value": "i-0c986c72"
                }
            ],
            "Statistic": "Average",
            "StateReason": "Threshold Crossed: 10 datapoints were not greater than
 or equal to the threshold (70.0). The most recent datapoints: [40.760000000000005,
 41.316].",
            "InsufficientDataActions": [],
            "OKActions": [],
            "ActionsEnabled": true,
            "MetricName": "CPUUtilization"
        },
        {
            "EvaluationPeriods": 2,
            "AlarmArn": "arn:aws:cloudwatch:us-
east-1:111122223333:alarm:myHighCpuAlarm",
```

```
            "StateUpdatedTimestamp": "2014-04-09T18:59:06.442Z",
            "AlarmConfigurationUpdatedTimestamp": "2014-04-09T22:26:05.958Z",
            "ComparisonOperator": "GreaterThanThreshold",
            "AlarmActions": [
                "arn:aws:sns:us-east-1:111122223333:HighCPUAlarm"
            ],
            "Namespace": "AWS/EC2",
            "AlarmDescription": "CPU usage exceeds 70 percent",
            "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":
\"2014-04-09T18:59:06.419+0000\",\"startDate\":\"2014-04-09T18:44:00.000+0000\",
\"statistic\":\"Average\",\"period\":300,\"recentDatapoints\":[38.958,40.292],
\"threshold\":70.0}",
            "Period": 300,
            "StateValue": "OK",
            "Threshold": 70.0,
            "AlarmName": "myHighCpuAlarm",
            "Dimensions": [
                {
                    "Name": "InstanceId",
                    "Value": "i-0c986c72"
                }
            ],
            "Statistic": "Average",
            "StateReason": "Threshold Crossed: 2 datapoints were not greater than
 the threshold (70.0). The most recent datapoints: [38.958, 40.292].",
            "InsufficientDataActions": [],
            "OKActions": [],
            "ActionsEnabled": false,
            "MetricName": "CPUUtilization"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAlarmsForMetric。

`describe-alarms`

以下代码示例演示了如何使用 `describe-alarms`。

Amazon CLI

列出有关警报的信息

以下示例使用 describe-alarms 命令提供名为“myalarm”的警报的相关信息：

```
aws cloudwatch describe-alarms --alarm-names "myalarm"
```

输出：

```
{
    "MetricAlarms": [
        {
            "EvaluationPeriods": 2,
            "AlarmArn": "arn:aws:cloudwatch:us-east-1:123456789012:alarm:myalarm",
            "StateUpdatedTimestamp": "2014-04-09T18:59:06.442Z",
            "AlarmConfigurationUpdatedTimestamp": "2012-12-27T00:49:54.032Z",
            "ComparisonOperator": "GreaterThanThreshold",
            "AlarmActions": [
                "arn:aws:sns:us-east-1:123456789012:myHighCpuAlarm"
            ],
            "Namespace": "AWS/EC2",
            "AlarmDescription": "CPU usage exceeds 70 percent",
            "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":
\"2014-04-09T18:59:06.419+0000\",\"startDate\":\"2014-04-09T18:44:00.000+0000\",
\"statistic\":\"Average\",\"period\":300,\"recentDatapoints\":[38.958,40.292],
\"threshold\":70.0}",
            "Period": 300,
            "StateValue": "OK",
            "Threshold": 70.0,
            "AlarmName": "myalarm",
            "Dimensions": [
                {
                    "Name": "InstanceId",
                    "Value": "i-0c986c72"
                }
            ],
            "Statistic": "Average",
            "StateReason": "Threshold Crossed: 2 datapoints were not greater than
 the threshold (70.0). The most recent datapoints: [38.958, 40.292].",
            "InsufficientDataActions": [],
            "OKActions": [],
            "ActionsEnabled": true,
            "MetricName": "CPUUtilization"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAlarms。

describe-anomaly-detectors

以下代码示例演示了如何使用 describe-anomaly-detectors。

Amazon CLI

检索异常检测模型的列表

以下 describe-anomaly-detectors 示例显示了有关与指定账户中 AWS/Logs 命名空间关联的异常检测器模型的信息。

```
aws cloudwatch describe-anomaly-detectors \
    --namespace AWS/Logs
```

输出：

```
{
    "AnomalyDetectors": [
        {
            "Namespace": "AWS/Logs",
            "MetricName": "IncomingBytes",
            "Dimensions": [],
            "Stat": "SampleCount",
            "Configuration": {
                "ExcludedTimeRanges": []
            },
            "StateValue": "TRAINED",
            "SingleMetricAnomalyDetector": {
                "AccountId": "123456789012",
                "Namespace": "AWS/Logs",
                "MetricName": "IncomingBytes",
                "Dimensions": [],
                "Stat": "SampleCount"
            }
        },
        {
            "Namespace": "AWS/Logs",
            "MetricName": "IncomingBytes",
            "Dimensions": [
                {
```

```
                    "Name": "LogGroupName",
                    "Value": "demo"
                }
            ],
            "Stat": "Average",
            "Configuration": {
                "ExcludedTimeRanges": []
            },
            "StateValue": "PENDING_TRAINING",
            "SingleMetricAnomalyDetector": {
                "AccountId": "123456789012",
                "Namespace": "AWS/Logs",
                "MetricName": "IncomingBytes",
                "Dimensions": [
                    {
                        "Name": "LogGroupName",
                        "Value": "demo"
                    }
                ],
                "Stat": "Average"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南中》的使用 CloudWatch 异常检测。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAnomalyDetectors。

`describe-insight-rules`

以下代码示例演示了如何使用 `describe-insight-rules`。

Amazon CLI

检索 Contributor Insights 规则列表

以下 `describe-insight-rules` 示例显示指定账户中的所有 Contributor Insights 规则。

```
aws cloudwatch describe-insight-rules
```

输出：

```
{
    "InsightRules": [
        {
            "Name": "Rule-A",
            "State": "ENABLED",
            "Schema": "CloudWatchLogRule/1",
            "Definition": "{\n\t\"AggregateOn\": \"Count\",\n\t\"Contribution\":
 {\n\t\t\"Filters\": [],\n\t\t\"Keys\": [\n\t\t\t\"$.requestId\"\n\t\t]\n\t},\n
\t\"LogFormat\": \"JSON\",\n\t\"Schema\": {\n\t\t\"Name\": \"CloudWatchLogRule
\",\n\t\t\"Version\": 1\n\t},\n\t\"LogGroupARNs\": [\n\t\t\"arn:aws:logs:us-
east-1:123456789012:log-group:demo\"\n\t]\n}",
            "ManagedRule": false
        },
        {
            "Name": "Rule-B",
            "State": "ENABLED",
            "Schema": "CloudWatchLogRule/1",
            "Definition": "{\n\t\"AggregateOn\": \"Count\",\n\t\"Contribution\":
 {\n\t\t\"Filters\": [],\n\t\t\"Keys\": [\n\t\t\t\"$.requestId\"\n\t\t]\n\t},\n
\t\"LogFormat\": \"JSON\",\n\t\"Schema\": {\n\t\t\"Name\": \"CloudWatchLogRule
\",\n\t\t\"Version\": 1\n\t},\n\t\"LogGroupARNs\": [\n\t\t\"arn:aws:logs:us-
east-1:123456789012:log-group:demo-1\"\n\t]\n}",
            "ManagedRule": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Contributor Insights 分析高基数数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInsightRules。

`disable-alarm-actions`

以下代码示例演示了如何使用 `disable-alarm-actions`。

Amazon CLI

禁用警报的操作

以下示例使用 `disable-alarm-actions` 命令禁用名为 myalarm 的警报的所有操作：

```
aws cloudwatch disable-alarm-actions --alarm-names myalarm
```

如果成功，该命令将返回到提示符。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableAlarmActions。

disable-insight-rules

以下代码示例演示了如何使用 disable-insight-rules。

Amazon CLI

禁用指定的 Contributor Insights 规则

以下 disable-insight-rules 示例禁用指定账户中两个分别名为 Rule-A 和 Rule-B 的 Contributor Insights 规则。

```
aws cloudwatch disable-insight-rules \
    --rule-names Rule-A Rule-B
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Contributor Insights 分析高基数数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableInsightRules。

enable-alarm-actions

以下代码示例演示了如何使用 enable-alarm-actions。

Amazon CLI

启用警报的所有操作

以下示例使用 enable-alarm-actions 命令启用名为 myalarm 的警报的所有操作：

```
aws cloudwatch enable-alarm-actions --alarm-names myalarm
```

如果成功，该命令将返回到提示符。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableAlarmActions。

enable-insight-rules

以下代码示例演示了如何使用 enable-insight-rules。

Amazon CLI

启用指定的 Contributor Insights 规则

以下 enable-insight-rules 示例启用指定账户中两个分别名为 Rule-A 和 Rule-B 的 Contributor Insights 规则。

```
aws cloudwatch enable-insight-rules \
    --rule-names Rule-A Rule-B
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Contributor Insights 分析高基数数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 EnableInsightRules。

get-dashboard

以下代码示例演示了如何使用 get-dashboard。

Amazon CLI

检索有关控制面板的信息

以下 get-dashboard 示例显示了有关指定账户中名为 Dashboard-A 的控制面板的信息。

```
aws cloudwatch get-dashboard \
```

```
    --dashboard-name Dashboard-A
```

输出：

```
{
    "DashboardArn": "arn:aws:cloudwatch::123456789012:dashboard/Dashboard-A",
    "DashboardBody": "{\"widgets\":[{\"type\":\"metric\",\"x\":0,\"y\":0,\"width
\":6,\"height\":6,\"properties\":{\"view\":\"timeSeries\",\"stacked\":false,
\"metrics\":[[\"AWS/EC2\",\"NetworkIn\",\"InstanceId\",\"i-0131f062232ade043\"],[\".
\",\"NetworkOut\",\".\",\".\"]],\"region\":\"us-east-1\"}}]}",
    "DashboardName": "Dashboard-A"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Amazon CloudWatch 控制面板。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDashboard。

get-insight-rule-report

以下代码示例演示了如何使用 get-insight-rule-report。

Amazon CLI

检索 Contributor Insights 规则收集的时间序列数据

以下 get-insight-rule-report 示例返回 Contributor Insights 规则收集的时间序列数据。

```
aws cloudwatch get-insight-rule-report \
    --rule-name Rule-A \
    --start-time 2024-10-13T20:15:00Z \
    --end-time 2024-10-13T20:30:00Z \
    --period 300
```

输出：

```
{
    "KeyLabels": [
        "PartitionKey"
    ],
    "AggregationStatistic": "Sum",
    "AggregateValue": 0.5,
```

```
    "ApproximateUniqueCount": 1,
    "Contributors": [
        {
            "Keys": [
                "RequestID"
            ],
            "ApproximateAggregateValue": 0.5,
            "Datapoints": [
                {
                    "Timestamp": "2024-10-13T21:00:00+00:00",
                    "ApproximateValue": 0.5
                }
            ]
        }
    ],
    "RuleAttributes": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Contributor Insights 分析高基数数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetInsightRuleReport。

get-metric-data

以下代码示例演示了如何使用 get-metric-data。

Amazon CLI

示例 1：使用数学表达式获取指定 EC2 的平均总 IOPS

以下 get-metric-data 示例使用组合 EBSReadOps 和 EBSWriteOps 指标的指标数学表达式检索 InstanceID 为 i-abcdef 的 EC2 实例 CloudWatch 指标值。

```
aws cloudwatch get-metric-data \
    --metric-data-queries file://file.json \
    --start-time 2024-09-29T22:10:00Z \
    --end-time 2024-09-29T22:15:00Z
```

的内容file.json：

```
[
    {
```

```
        "Id": "m3",
        "Expression": "(m1+m2)/300",
        "Label": "Avg Total IOPS"
    },
    {
        "Id": "m1",
        "MetricStat": {
            "Metric": {
                "Namespace": "AWS/EC2",
                "MetricName": "EBSReadOps",
                "Dimensions": [
                    {
                        "Name": "InstanceId",
                        "Value": "i-abcdef"
                    }
                ]
            },
            "Period": 300,
            "Stat": "Sum",
            "Unit": "Count"
        },
        "ReturnData": false
    },
    {
        "Id": "m2",
        "MetricStat": {
            "Metric": {
                "Namespace": "AWS/EC2",
                "MetricName": "EBSWriteOps",
                "Dimensions": [
                    {
                        "Name": "InstanceId",
                        "Value": "i-abcdef"
                    }
                ]
            },
            "Period": 300,
            "Stat": "Sum",
            "Unit": "Count"
        },
        "ReturnData": false
    }
]
```

输出：

```
{
    "MetricDataResults": [
        {
            "Id": "m3",
            "Label": "Avg Total IOPS",
            "Timestamps": [
                "2024-09-29T22:10:00+00:00"
            ],
            "Values": [
                96.85
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

示例 2：使用 CloudWatch 账单指标监控预估 Amazon 费用

以下 `get-metric-data` 示例从 Amazon/Billing 命名空间检索 `EstimatedCharges` CloudWatch 指标。

```
aws cloudwatch get-metric-data \
    --metric-data-queries '[{"Id":"m1","MetricStat":{"Metric":
{"Namespace":"AWS/Billing","MetricName":"EstimatedCharges","Dimensions":
[{"Name":"Currency","Value":"USD"}]},"Period":21600,"Stat":"Maximum"}}]' \
    --start-time 2024-09-26T12:00:00Z \
    --end-time 2024-09-26T18:00:00Z \
    --region us-east-1
```

输出：

```
{
    "MetricDataResults": [
        {
            "Id": "m1",
            "Label": "EstimatedCharges",
            "Timestamps": [
                "2024-09-26T12:00:00+00:00"
            ],
            "Values": [
```

```
                542.38
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的将数学表达式与 CloudWatch 指标结合使用。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMetricData。

get-metric-statistics

以下代码示例演示了如何使用 get-metric-statistics。

Amazon CLI

获取每个 EC2 实例的 CPU 利用率

以下示例使用 get-metric-statistics 命令获取 ID 为 i-abcdef 的 EC2 实例的 CPU 利用率。

```
aws cloudwatch get-metric-statistics --metric-name CPUUtilization --start-
time 2014-04-08T23:18:00Z --end-time 2014-04-09T23:18:00Z --period 3600 --
namespace AWS/EC2 --statistics Maximum --dimensions Name=InstanceId,Value=i-abcdef
```

输出：

```
{
    "Datapoints": [
        {
            "Timestamp": "2014-04-09T11:18:00Z",
            "Maximum": 44.79,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T20:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T19:18:00Z",
```

```
            "Maximum": 50.85,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T09:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T03:18:00Z",
            "Maximum": 76.84,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T21:18:00Z",
            "Maximum": 48.96,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T14:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T08:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T16:18:00Z",
            "Maximum": 45.55,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T06:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T13:18:00Z",
            "Maximum": 45.08,
            "Unit": "Percent"
        },
        {
```

```
            "Timestamp": "2014-04-09T05:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T18:18:00Z",
            "Maximum": 46.88,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T17:18:00Z",
            "Maximum": 52.08,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T07:18:00Z",
            "Maximum": 47.92,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T02:18:00Z",
            "Maximum": 51.23,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T12:18:00Z",
            "Maximum": 47.67,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-08T23:18:00Z",
            "Maximum": 46.88,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T10:18:00Z",
            "Maximum": 51.91,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T04:18:00Z",
            "Maximum": 47.13,
            "Unit": "Percent"
        },
```

```
        {
            "Timestamp": "2014-04-09T15:18:00Z",
            "Maximum": 48.96,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T00:18:00Z",
            "Maximum": 48.16,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2014-04-09T01:18:00Z",
            "Maximum": 49.18,
            "Unit": "Percent"
        }
    ],
    "Label": "CPUUtilization"
}
```

指定多个维度

以下示例说明如何指定多个维度。每个维度都指定为一个“名称/值”对，名称和值之间用逗号隔开。多个维度之间用空格隔开。如果单个指标包含多个维度，则必须为每个已定义的维度指定一个值。

有关使用 `get-metric-statistics` 命令的更多示例，请参阅《Amazon CloudWatch 开发人员指南》中的“获取指标的统计数据”。

```
aws cloudwatch get-metric-statistics --metric-name Buffers --namespace MyNameSpace
 --dimensions Name=InstanceID,Value=i-abcdef Name=InstanceType,Value=m1.small --
start-time 2016-10-15T04:00:00Z --end-time 2016-10-19T07:00:00Z --statistics Average
 --period 60
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMetricStatistics。

get-metric-stream

以下代码示例演示了如何使用 `get-metric-stream`。

Amazon CLI

检索有关指标流的信息

以下 get-metric-stream 示例显示有关指定账户中名为 QuickFull-GuaFbs 的指标流的信息。

```
aws cloudwatch get-metric-stream \
    --name QuickFull-GuaFbs
```

输出：

```
{
    "Arn": "arn:aws:cloudwatch:us-east-1:123456789012:metric-stream/QuickFull-GuaFbs",
    "Name": "QuickFull-GuaFbs",
    "FirehoseArn": "arn:aws:firehose:us-east-1:123456789012:deliverystream/MetricStreams-QuickFull-GuaFbs-WnySbECG",
    "RoleArn": "arn:aws:iam::123456789012:role/service-role/MetricStreams-FirehosePutRecords-JN10W9B3",
    "State": "running",
    "CreationDate": "2024-10-11T18:48:59.187000+00:00",
    "LastUpdateDate": "2024-10-11T18:48:59.187000+00:00",
    "OutputFormat": "json",
    "IncludeLinkedAccountsMetrics": false
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMetricStream。

get-metric-widget-image

以下代码示例演示了如何使用 get-metric-widget-image。

Amazon CLI

检索 CPUUtilization 的快照图

以下 get-metric-widget-image 示例检索 ID 为 i-abcde 的 EC2 实例的指标 CPUUtilization 快照图，并将检索到的图像保存为本地计算机上名为“image.png”的文件。

```
aws cloudwatch get-metric-widget-image \
    --metric-widget '{"metrics":[["AWS/EC2","CPUUtilization","InstanceId","i-abcde"]]}' \
    --output-format png \
```

```
    --output text | base64 --decode > image.png
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMetricWidgetImage。

list-dashboards

以下代码示例演示了如何使用 list-dashboards。

Amazon CLI

检索控制面板的列表

以下 list-dashboards 示例列出了指定账户中的所有控制面板。

```
aws cloudwatch list-dashboards
```

输出：

```
{
    "DashboardEntries": [
        {
            "DashboardName": "Dashboard-A",
            "DashboardArn": "arn:aws:cloudwatch::123456789012:dashboard/Dashboard-
A",
            "LastModified": "2024-10-11T18:40:11+00:00",
            "Size": 271
        },
        {
            "DashboardName": "Dashboard-B",
            "DashboardArn": "arn:aws:cloudwatch::123456789012:dashboard/Dashboard-
B",
            "LastModified": "2024-10-11T18:44:41+00:00",
            "Size": 522
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Amazon CloudWatch 控制面板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDashboards。

list-metric-streams

以下代码示例演示了如何使用 list-metric-streams。

Amazon CLI

检索指标流列表

以下 list-metric-streams 示例列出指定账户中的所有指标流。

```
aws cloudwatch list-metric-streams
```

输出：

```
{
    "Entries": [
        {
            "Arn": "arn:aws:cloudwatch:us-east-1:123456789012:metric-stream/
QuickFull-GuaFbs",
            "CreationDate": "2024-10-11T18:48:59.187000+00:00",
            "LastUpdateDate": "2024-10-11T18:48:59.187000+00:00",
            "Name": "QuickFull-GuaFbs",
            "FirehoseArn": "arn:aws:firehose:us-east-1:123456789012:deliverystream/
MetricStreams-QuickFull-GuaFbs-WnySbECG",
            "State": "running",
            "OutputFormat": "json"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListMetricStreams。

list-metrics

以下代码示例演示了如何使用 list-metrics。

Amazon CLI

列出 Amazon SNS 的指标

以下 `list-metrics` 示例显示 Amazon SNS 的指标。

```
aws cloudwatch list-metrics \
    --namespace "AWS/SNS"
```

输出：

```
{
    "Metrics": [
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "NotifyMe"
                }
            ],
            "MetricName": "PublishSize"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "CFO"
                }
            ],
            "MetricName": "PublishSize"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "NotifyMe"
                }
            ],
            "MetricName": "NumberOfNotificationsFailed"
        },
```

```
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "NotifyMe"
                }
            ],
            "MetricName": "NumberOfNotificationsDelivered"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "NotifyMe"
                }
            ],
            "MetricName": "NumberOfMessagesPublished"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "CFO"
                }
            ],
            "MetricName": "NumberOfMessagesPublished"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
                    "Name": "TopicName",
                    "Value": "CFO"
                }
            ],
            "MetricName": "NumberOfNotificationsDelivered"
        },
        {
            "Namespace": "AWS/SNS",
            "Dimensions": [
                {
```

```
                    "Name": "TopicName",
                    "Value": "CFO"
                }
            ],
            "MetricName": "NumberOfNotificationsFailed"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMetrics。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出与现有警报关联的标签*

以下 list-tags-for-resource 示例列出与指定账户中名为 demo 的警报关联的所有标签。

```
aws cloudwatch list-tags-for-resource \
    --resource-arn arn:aws:cloudwatch:us-east-1:123456789012:alarm:demo
```

输出：

```
{
    "Tags": [
        {
            "Key": "stack",
            "Value": "Production"
        },
        {
            "Key": "team",
            "Value": "Devops"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的警报和标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-anomaly-detector

以下代码示例演示了如何使用 put-anomaly-detector。

Amazon CLI

创建异常检测模型

以下 put-anomaly-detector 示例为 CloudWatch 指标创建了异常检测模型。

```
aws cloudwatch put-anomaly-detector \
    --namespace AWS/Logs \
    --metric-name IncomingBytes \
    --stat SampleCount
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南中》的使用 CloudWatch 异常检测。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutAnomalyDetector。

put-composite-alarm

以下代码示例演示了如何使用 put-composite-alarm。

Amazon CLI

创建复合 CloudWatch 警报

以下 put-composite-alarm 示例在指定账户中创建名为 ProdAlarm 的复合警报。

```
aws cloudwatch put-composite-alarm \
    --alarm-name ProdAlarm \
    --alarm-rule "ALARM(CPUUtilizationTooHigh) AND ALARM(MemUsageTooHigh)" \
    --alarm-actions arn:aws:sns:us-east-1:123456789012:demo \
    --actions-enabled
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的创建复合警报。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutCompositeAlarm。

put-dashboard

以下代码示例演示了如何使用 put-dashboard。

Amazon CLI

创建仪表板

以下 put-dashboard 示例在指定账户中创建了名为 Dashboard-A 的控制面板。

```
aws cloudwatch put-dashboard \
    --dashboard-name Dashboard-A \
    --dashboard-body '{"widgets":
[{"height":6,"width":6,"y":0,"x":0,"type":"metric","properties":
{"view":"timeSeries","stacked":false,"metrics":
[["Namespace","CPUUtilization","Environment","Prod","Type","App"]],"region":"us-
east-1"}}]}'
```

输出：

```
{
    "DashboardValidationMessages": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的创建 Amazon CloudWatch 控制面板。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutDashboard。

put-insight-rule

以下代码示例演示了如何使用 put-insight-rule。

Amazon CLI

创建 Contributor Insights 规则

以下 put-insight-rule 示例在指定账户中创建名为 VPCFlowLogsContributorInsights 的 Contributor Insights 规则。

```
aws cloudwatch put-insight-rule \
    --rule-name VPCFlowLogsContributorInsights \
```

```
    --rule-definition file://insight-rule.json \
    --rule-state ENABLED
```

的内容insight-rule.json：

```
{
    "Schema": {
        "Name": "CloudWatchLogRule",
        "Version": 1
    },
    "AggregateOn": "Count",
    "Contribution": {
        "Filters": [],
        "Keys": [
            "tcp-flag"
        ]
    },
    "LogFormat": "CLF",
    "LogGroupNames": [
        "/vpc/flowlogs/*"
    ],
    "Fields": {
        "23": "tcp-flag"
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的在 CloudWatch 中创建 Contributor Insights 规则。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutInsightRule。

put-metric-alarm

以下代码示例演示了如何使用 put-metric-alarm。

Amazon CLI

在 CPU 利用率超过 70% 时发送 Amazon Simple Notification Service 电子邮件消息

以下示例使用 put-metric-alarm 命令在 CPU 利用率超过 70% 时发送 Amazon Simple Notification Service 电子邮件消息：

```
aws cloudwatch put-metric-alarm --alarm-name cpu-mon --alarm-description "Alarm
 when CPU exceeds 70 percent" --metric-name CPUUtilization --namespace AWS/
EC2 --statistic Average --period 300 --threshold 70 --comparison-
operator GreaterThanThreshold  --dimensions "Name=InstanceId,Value=i-12345678" --
evaluation-periods 2 --alarm-actions arn:aws:sns:us-east-1:111122223333:MyTopic --
unit Percent
```

如果成功，该命令将返回到提示符。如果已存在同名警报，则该警报将被新警报覆盖。

指定多个维度

以下示例说明了如何指定多个维度。每个维度都指定为一个“名称/值”对，名称和值之间用逗号隔开。多个维度之间用空格隔开：

```
aws cloudwatch put-metric-alarm --alarm-name "Default_Test_Alarm3" --alarm-
description "The default example alarm" --namespace "CW EXAMPLE METRICS"
 --metric-name Default_Test --statistic Average --period 60 --evaluation-
periods 3 --threshold 50 --comparison-operator GreaterThanOrEqualToThreshold --
dimensions Name=key1,Value=value1 Name=key2,Value=value2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMetricAlarm。

put-metric-data

以下代码示例演示了如何使用 put-metric-data。

Amazon CLI

向 Amazon CloudWatch 发布自定义指标

以下示例使用 put-metric-data 命令向 Amazon CloudWatch 发布自定义指标：

```
aws cloudwatch put-metric-data --namespace "Usage Metrics" --metric-data file://
metric.json
```

指标本身的值存储在 JSON 文件 metric.json 中。

以下是该文件的内容：

```
[
```

```
  {
    "MetricName": "New Posts",
    "Timestamp": "Wednesday, June 12, 2013 8:28:20 PM",
    "Value": 0.50,
    "Unit": "Count"
  }
]
```

有关更多信息，请参阅《Amazon CloudWatch 开发人员指南》中的“发布自定义指标”。

指定多个维度

以下示例说明了如何指定多个维度。每个维度都指定为一个 Name=Value 对。多个维度之间用逗号隔开：

```
aws cloudwatch put-metric-data --metric-name Buffers --
namespace MyNameSpace --unit Bytes --value 231434333 --
dimensions InstanceID=1-23456789,InstanceType=m1.small
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMetricData。

put-metric-stream

以下代码示例演示了如何使用 put-metric-stream。

Amazon CLI

创建指标流

以下 put-metric-stream 示例在指定账户中创建名为 QuickFull-GuaFb 的指标流。

```
aws cloudwatch put-metric-stream \
    --name QuickFull-GuaFbs \
    --firehose-arn arn:aws:firehose:us-east-1:123456789012:deliverystream/
MetricStreams-QuickFull-GuaFbs-WnySbECG \
    --role-arn arn:aws:iam::123456789012:role/service-role/MetricStreams-
FirehosePutRecords-JN10W9B3 \
    --output-format json \
    --no-include-linked-accounts-metrics
```

输出：

```
{
    "Arn": "arn:aws:cloudwatch:us-east-1:123456789012:metric-stream/QuickFull-
GuaFbs"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的设置指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutMetricStream。

set-alarm-state

以下代码示例演示了如何使用 set-alarm-state。

Amazon CLI

暂时更改警报的状态

以下示例使用 set-alarm-state 命令暂时更改名为“myalarm”的 Amazon CloudWatch 警报的状态，并将其设置为 ALARM 状态以进行测试：

```
aws cloudwatch set-alarm-state --alarm-name "myalarm" --state-value ALARM --state-
reason "testing purposes"
```

如果成功，该命令将返回到提示符。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetAlarmState。

start-metric-streams

以下代码示例演示了如何使用 start-metric-streams。

Amazon CLI

启动指定的指标流

以下 start-metric-streams 示例启动指定账户中名为 QuickFull-GuaFbs 的指标流。

```
aws cloudwatch start-metric-streams \
    --names QuickFull-GuaFbs
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartMetricStreams。

stop-metric-streams

以下代码示例演示了如何使用 stop-metric-streams。

Amazon CLI

停止指定的指标流

以下 stop-metric-streams 示例停止指定账户中名为 QuickFull-GuaFbs 的指标流。

```
aws cloudwatch stop-metric-streams \
    --names QuickFull-GuaFbs
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用指标流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopMetricStreams。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为指定资源添加一个或多个标签

以下 tag-resource 示例为指定账户中名为 demo 的 CloudWatch 警报添加 2 个标签。

```
aws cloudwatch tag-resource \
    --resource-arn arn:aws:cloudwatch:us-east-1:123456789012:alarm:demo \
    --tags Key=stack,Value=Production Key=team,Value=Devops
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的标记 Amazon CloudWatch 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从指定资源中移除一个或多个标签

以下 untag-resource 示例从指定账户中名为 demo 的 CloudWatch 警报中移除 2 个标签。

```
aws cloudwatch untag-resource \
    --resource-arn arn:aws:cloudwatch:us-east-1:123456789012:alarm:demo \
    --tag-keys stack team
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的标记 Amazon CloudWatch 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 CloudWatch Logs 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CloudWatch Logs 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

create-log-group

以下代码示例演示了如何使用 create-log-group。

Amazon CLI

以下命令创建名为 my-logs 的日志组：

```
aws logs create-log-group --log-group-name my-logs
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLogGroup。

create-log-stream

以下代码示例演示了如何使用 create-log-stream。

Amazon CLI

以下命令在日志组 my-logs 中创建一个名为 20150601 的日志流：

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLogStream。

delete-log-group

以下代码示例演示了如何使用 delete-log-group。

Amazon CLI

以下命令删除名为 my-logs 的日志组：

```
aws logs delete-log-group --log-group-name my-logs
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLogGroup。

delete-log-stream

以下代码示例演示了如何使用 delete-log-stream。

Amazon CLI

以下命令从名为 my-logs 的日志组中删除名为 20150531 的日志流：

```
aws logs delete-log-stream --log-group-name my-logs --log-stream-name 20150531
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLogStream。

delete-retention-policy

以下代码示例演示了如何使用 delete-retention-policy。

Amazon CLI

以下命令移除以前应用于名为 my-logs 的日志组的留存策略：

```
aws logs delete-retention-policy --log-group-name my-logs
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRetentionPolicy。

describe-log-groups

以下代码示例演示了如何使用 describe-log-groups。

Amazon CLI

以下命令描述名为 my-logs 的日志组：

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

输出：

```
{
    "logGroups": [
        {
            "storedBytes": 0,
            "metricFilterCount": 0,
            "creationTime": 1433189500783,
            "logGroupName": "my-logs",
            "retentionInDays": 5,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLogGroups。

describe-log-streams

以下代码示例演示了如何使用 describe-log-streams。

Amazon CLI

以下命令显示日志组 my-logs 中以前缀 2015 开头的所有日志流：

```
aws logs describe-log-streams --log-group-name my-logs --log-stream-name-prefix 2015
```

输出：

```
{
    "logStreams": [
        {
            "creationTime": 1433189871774,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-
stream:20150531",
            "logStreamName": "20150531",
            "storedBytes": 0
        },
        {
            "creationTime": 1433189873898,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-
stream:20150601",
            "logStreamName": "20150601",
            "storedBytes": 0
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLogStreams。

get-log-events

以下代码示例演示了如何使用 get-log-events。

Amazon CLI

以下命令从日志组 my-logs 中名为 20150601 的日志流中检索日志事件：

```
aws logs get-log-events --log-group-name my-logs --log-stream-name 20150601
```

输出：

```
{
    "nextForwardToken":
 "f/31961209122447488583055879464742346735121166569214640130",
    "events": [
        {
            "ingestionTime": 1433190494190,
            "timestamp": 1433190184356,
            "message": "Example Event 1"
        },
        {
            "ingestionTime": 1433190516679,
            "timestamp": 1433190184356,
            "message": "Example Event 1"
        },
        {
            "ingestionTime": 1433190494190,
            "timestamp": 1433190184358,
            "message": "Example Event 2"
        }
    ],
    "nextBackwardToken":
 "b/31961209122358285602261756944988674324553373268216709120"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLogEvents。

put-log-events

以下代码示例演示了如何使用 put-log-events。

Amazon CLI

以下命令在日志组 my-logs 中名为 20150601 的日志流中添加日志事件：

```
aws logs put-log-events --log-group-name my-logs --log-stream-name 20150601 --log-
events file://events
```

输出：

```
{
```

```
    "nextSequenceToken": "495426724868310740095796045676567882148068632824696073 46"
}
```

上面的示例将从当前目录中名为 events 的文件中读取一个 JSON 事件数组：

```
[
  {
    "timestamp": 1433190184356,
    "message": "Example Event 1"
  },
  {
    "timestamp": 1433190184358,
    "message": "Example Event 2"
  },
  {
    "timestamp": 1433190184360,
    "message": "Example Event 3"
  }
]
```

每个后续调用都需要使用序列令牌选项指定上一个调用提供的下一个序列令牌：

```
aws logs put-log-events --log-group-name my-logs --log-
stream-name 20150601 --log-events file://events2 --sequence-
token "495426724868310740095796045676567882148068632824696073 46"
```

输出：

```
{
    "nextSequenceToken": "495426724868310740095796045679009912303690199563082198 26"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLogEvents。

put-retention-policy

以下代码示例演示了如何使用 put-retention-policy。

Amazon CLI

以下命令向名为 my-logs 的日志组中添加 5 天留存策略：

```
aws logs put-retention-policy --log-group-name my-logs --retention-in-days 5
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRetentionPolicy。

使用 Amazon CLI 的 CloudWatch 网络监控示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CloudWatch 网络监控结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-monitor`

以下代码示例演示了如何使用 `create-monitor`。

Amazon CLI

示例 1：创建具有聚合周期的网络监测仪

以下 `create-monitor` 示例创建一个名为 `Example_NetworkMonitor` 的监测仪，其中 `aggregationPeriod` 设置为 `30` 秒。该监测仪的初始 `state` 将是 `INACTIVE`，因为没有探测器与之关联。只有在添加探测器后，状态才会变成 `ACTIVE`。您可以使用 update-monitor 或 create-probe 命令将探测器添加到此监测仪中。

```
aws networkmonitor create-monitor \
     --monitor-name Example_NetworkMonitor \
     --aggregation-period 30
```

输出：

```
{
```

```
    "monitorArn": "arn:aws:networkmonitor:region:111122223333:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "INACTIVE",
    "aggregationPeriod": 30,
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

示例 2：创建带有探测器的网络监测仪，探测器使用 TCP 并包括标签

以下 `create-monitor` 示例创建一个名为 `Example_NetworkMonitor` 的监测仪。该命令还会创建一个使用 ICMP 协议并包括标签的探测器。因为请求中没有传入 `aggregationPeriod`，所以 60 秒设置为默认值。带有探测器的监测仪的 `state` 将为 `PENDING`，直到监测仪为 `ACTIVE`。这可能需要几分钟，当 `state` 变为 `ACTIVE` 时，您就可以开始查看 CloudWatch 指标了。

```
aws networkmonitor create-monitor \
    --monitor-name Example_NetworkMonitor \
    --probes sourceArn=arn:aws:ec2:region:111122223333:subnet/subnet-
id,destination=10.0.0.100,destinationPort=80,protocol=TCP,packetSize=56,probeTags={Name=Prob
 \
    --tags Monitor=Monitor1
```

输出：

```
{
    "monitorArn": "arn:aws:networkmonitor:region111122223333:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "PENDING",
    "aggregationPeriod": 60,
    "tags": {
        "Monitor": "Monitor1"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

示例 3：创建带有探测器的网络监测仪，探测器使用 ICMP 并包括标签

以下 create-monitor 示例创建一个名为 Example_NetworkMonitor 的监测仪，其中 aggregationPeriod 设置为 30 秒。该命令还会创建一个使用 ICMP 协议并包括标签的探测器。因为请求中没有传入 aggregationPeriod，所以 60 秒设置为默认值。带有探测器的监测仪的 state 将为 PENDING，直到监测仪为 ACTIVE。这可能需要几分钟，当 state 变为 ACTIVE 时，您就可以开始查看 CloudWatch 指标了。

```
aws networkmonitor create-monitor \
     --monitor-name Example_NetworkMonitor \
     --aggregation-period 30 \
     --probes sourceArn=arn:aws:ec2:region111122223333:subnet/subnet-
id,destination=10.0.0.100,protocol=ICMP,packetSize=56,probeTags={Name=Probe1} \
     --tags Monitor=Monitor1
```

输出：

```
{
    "monitorArn": "arn:aws:networkmonitor:region:111122223333:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "PENDING",
    "aggregationPeriod": 30,
    "tags": {
        "Monitor": "Monitor1"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMonitor。

create-probe

以下代码示例演示了如何使用 create-probe。

Amazon CLI

示例 1：创建使用 TCP 的探测器并将其添加到网络监测仪中

以下 create-probe 示例创建一个使用 TCP protocol 的探测器并将该探测器添加到名为 Example_NetworkMonitor 的监测仪中。创建后，带有探测器的监测仪的 state 将为

PENDING，直到监测仪为 ACTIVE。这可能需要几分钟，当 state 变为 ACTIVE 时，您就可以开始查看 CloudWatch 指标了。

```
aws networkmonitor create-probe \
    --monitor-name Example_NetworkMonitor \
    --probe sourceArn=arn:aws:ec2:region:111122223333:subnet/subnet-id,destination=10.0.0.100,destinationPort=80,protocol=TCP,packetSize=56,tags={Name=Probe1}
```

输出：

```
{
    "probeId": "probe-12345",
    "probeArn": "arn:aws:networkmonitor:region:111122223333:probe/probe-12345",
    "destination": "10.0.0.100",
    "destinationPort": 80,
    "packetSize": 56,
    "addressFamily": "IPV4",
    "vpcId": "vpc-12345",
    "state": "PENDING",
    "createdAt": "2024-03-29T12:41:57.314000-04:00",
    "modifiedAt": "2024-03-29T12:41:57.314000-04:00",
    "tags": {
        "Name": "Probe1"
    }
}
```

示例 2：创建使用 ICMP 的探测器并将其添加到网络监测仪中

以下 create-probe 示例创建一个使用 ICMP protocol 的探测器并将该探测器添加到名为 Example_NetworkMonitor 的监测仪中。创建后，带有探测器的监测仪的 state 将为 PENDING，直到监测仪为 ACTIVE。这可能需要几分钟，当 state 变为 ACTIVE 时，您就可以开始查看 CloudWatch 指标了。

```
aws networkmonitor create-probe \
    --monitor-name Example_NetworkMonitor \
    --probe sourceArn=arn:aws:ec2:region:012345678910:subnet/subnet-id,destination=10.0.0.100,protocol=ICMP,packetSize=56,tags={Name=Probe1}
```

输出：

```
{
```

```
    "probeId": "probe-12345",
    "probeArn": "arn:aws:networkmonitor:region:111122223333:probe/probe-12345",
    "destination": "10.0.0.100",
    "packetSize": 56,
    "addressFamily": "IPV4",
    "vpcId": "vpc-12345",
    "state": "PENDING",
    "createdAt": "2024-03-29T12:44:02.452000-04:00",
    "modifiedAt": "2024-03-29T12:44:02.452000-04:00",
    "tags": {
        "Name": "Probe1"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProbe。

delete-monitor

以下代码示例演示了如何使用 delete-monitor。

Amazon CLI

删除监测仪

以下 delete-monitor 示例删除名为 Example_NetworkMonitor 的监测仪。

```
aws networkmonitor delete-monitor \
    --monitor-name Example_NetworkMonitor
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMonitor。

delete-probe

以下代码示例演示了如何使用 delete-probe。

Amazon CLI

删除探测器

以下 delete-probe 示例从名为 Example_NetworkMonitor 的网络监测仪中删除 ID 为 probe-12345 的探测器。

```
aws networkmonitor delete-probe \
    --monitor-name Example_NetworkMonitor \
    --probe-id probe-12345
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProbe。

get-monitor

以下代码示例演示了如何使用 get-monitor。

Amazon CLI

获取监测仪信息

以下 get-monitor 示例获取有关名为 Example_NetworkMonitor 的监测仪的信息。

```
aws networkmonitor get-monitor \
    --monitor-name Example_NetworkMonitor
```

输出：

```
{
    "monitorArn": "arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "ACTIVE",
    "aggregationPeriod": 60,
    "tags": {},
    "probes": [],
```

```
    "createdAt": "2024-04-01T17:58:07.211000-04:00",
    "modifiedAt": "2024-04-01T17:58:07.211000-04:00"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMonitor。

get-probe

以下代码示例演示了如何使用 get-probe。

Amazon CLI

查看探测器的详细信息

以下 get-probe 示例返回与名为 Example_NetworkMonitor 的监测仪相关且 probeID 为 probe-12345 的探测器的详细信息。

```
aws networkmonitor get-probe \
    --monitor-name Example_NetworkMonitor \
    --probe-id probe-12345
```

输出：

```
{
    "probeId": "probe-12345",
    "probeArn": "arn:aws:networkmonitor:region:012345678910:probe/probe-12345",
    "sourceArn": "arn:aws:ec2:region:012345678910:subnet/subnet-12345",
    "destination": "10.0.0.100",
    "destinationPort": 80,
    "protocol": "TCP",
    "packetSize": 56,
    "addressFamily": "IPV4",
    "vpcId": "vpc-12345",
    "state": "ACTIVE",
    "createdAt": "2024-03-29T12:41:57.314000-04:00",
    "modifiedAt": "2024-03-29T12:42:28.610000-04:00",
    "tags": {
        "Name": "Probe1"
    }
```

```
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProbe。

`list-monitors`

以下代码示例演示了如何使用 `list-monitors`。

Amazon CLI

示例 1：列出所有监测仪（单个监测仪）

以下 `list-monitors` 示例返回仅包含单个监测仪的列表。该监测仪的 `state` 为 `ACTIVE`，`aggregationPeriod` 为 60 秒。

```
aws networkmonitor list-monitors
```

输出：

```
{
    "monitors": [{
            "monitorArn": "arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor",
            "monitorName": "Example_NetworkMonitor",
            "state": "ACTIVE",
            "aggregationPeriod": 60,
            "tags": {
                "Monitor": "Monitor1"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

示例 2：列出所有监测仪（多个监测仪）

以下 list-monitors 示例返回包含三个监测仪的列表。其中一个监测仪的 state 为 ACTIVE 并且生成 CloudWatch 指标。另两个监测仪的状态为 INACTIVE 且未生成 CloudWatch 指标。所有三个监测仪使用的 aggregationPeriod 均为 60 秒。

```
aws networkmonitor list-monitors
```

输出：

```
{
    "monitors": [
        {
            "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
Example_NetworkMonitor",
            "monitorName": "Example_NetworkMonitor",
            "state": "INACTIVE",
            "aggregationPeriod": 60,
            "tags": {}
        },
        {
            "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
Example_NetworkMonitor2",
            "monitorName": "Example_NetworkMonitor2",
            "state": "ACTIVE",
            "aggregationPeriod": 60,
            "tags": {
                "Monitor": "Monitor1"
            }
        },
        {
            "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
TestNetworkMonitor_CLI",
            "monitorName": "TestNetworkMonitor_CLI",
            "state": "INACTIVE",
            "aggregationPeriod": 60,
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMonitors。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例返回名为 Example_NetworkMonitor 的监测仪的标签列表。

```
aws networkmonitor list-tags-for-resource \
    --resource-arn arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor
```

输出：

```
{
    "tags": {
        "Environment": "Dev",
        "Application": "PetStore"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例使用 Environment=Dev 和 Application=PetStore 标签标记名为 Example_NetworkMonitor 的监测仪。

```
aws networkmonitor tag-resource \
```

```
    --resource-arn arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor \
    --tags Environment=Dev,Application=PetStore
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

取消标记资源

以下 untag-resource 示例将键-值对为 Environment Application 的 tag-keys 参数从它与名为 Example_NetworkMonitor 的监测仪的关联中移除。

```
aws networkmonitor untag-resource \
    --resource-arn arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor \
    --tag-keys Environment Application
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-monitor

以下代码示例演示了如何使用 update-monitor。

Amazon CLI

更新监测仪

以下 update-monitor 示例将监测仪的 aggregationPeriod 从 60 秒改为 30 秒。

```
aws networkmonitor update-monitor \
    --monitor-name Example_NetworkMonitor \
    --aggregation-period 30
```

输出：

```
{
    "monitorArn": "arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "PENDING",
    "aggregationPeriod": 30,
    "tags": {
        "Monitor": "Monitor1"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMonitor。

update-probe

以下代码示例演示了如何使用 update-probe。

Amazon CLI

更新探测器

以下 update-probe 示例更新探测器的原始 destination IP 地址，并将 packetSize 更新为 60。

```
aws networkmonitor update-probe \
    --monitor-name Example_NetworkMonitor \
    --probe-id probe-12345 \
    --destination 10.0.0.150 \
    --packet-size 60
```

输出：

```
{
```

```
    "probeId": "probe-12345",
    "probeArn": "arn:aws:networkmonitor:region:012345678910:probe/probe-12345",
    "sourceArn": "arn:aws:ec2:region:012345678910:subnet/subnet-12345",
    "destination": "10.0.0.150",
    "destinationPort": 80,
    "protocol": "TCP",
    "packetSize": 60,
    "addressFamily": "IPV4",
    "vpcId": "vpc-12345",
    "state": "PENDING",
    "createdAt": "2024-03-29T12:41:57.314000-04:00",
    "modifiedAt": "2024-03-29T13:52:23.115000-04:00",
    "tags": {
        "Name": "Probe1"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Amazon CloudWatch 网络状况监测仪的工作方式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProbe。

使用 Amazon CLI 的 CloudWatch Observability Access Monitor 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 CloudWatch Observability Access Monitor 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-link`

以下代码示例演示了如何使用 `create-link`。

Amazon CLI

创建链接

以下 create-link 示例在源账户和您在监控账户中创建的接收器之间创建链接。

```
aws oam create-link \
    --label-template sourceAccount \
    --resource-types AWS::CloudWatch::Metric \
    --sink-identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

输出：

```
{
    "Arn": "arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-example11111",
    "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
    "Label": "sourceAccount",
    "LabelTemplate": "sourceAccount",
    "ResourceTypes": [
        "AWS::CloudWatch::Metric"
    ],
    "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLink。

create-sink

以下代码示例演示了如何使用 create-sink。

Amazon CLI

创建接收器

以下 create-sink 示例在当前账户中创建接收器，以便该账户可用作 CloudWatch 跨账户可观测性中的监控账户。

```
aws oam create-sink \
    --name DemoSink
```

输出：

```
{
    "Arn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
    "Id": "a1b2c3d4-5678-90ab-cdef-example12345",
    "Name": "DemoSink",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateSink。

delete-link

以下代码示例演示了如何使用 delete-link。

Amazon CLI

删除链接

以下 delete-link 示例删除监控账户接收器和源账户之间的链接。

```
aws oam delete-link \
    --identifier arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-
example11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLink。

delete-sink

以下代码示例演示了如何使用 delete-sink。

Amazon CLI

删除接收器

以下 delete-sink 示例删除接收器。您必须先删除所有指向接收器的链接，之后才能删除接收器。

```
aws oam delete-sink \
    --identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteSink。

get-link

以下代码示例演示了如何使用 get-link。

Amazon CLI

返回有关某个链接的完整信息

以下 get-link 示例返回有关某个链接的完整信息。

```
aws oam get-link \
    --identifier arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-example11111
```

输出：

```
{
    "Arn": "arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-example11111",
    "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
    "Label": "sourceAccount",
    "LabelTemplate": "sourceAccount",
    "ResourceTypes": [
        "AWS::CloudWatch::Metric"
```

```
    ],
    "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetLink。

get-sink-policy

以下代码示例演示了如何使用 get-sink-policy。

Amazon CLI

返回附加到接收器的当前接收器策略

以下 get-sink-policy 示例返回附加到接收器的当前接收器策略。

```
aws oam get-sink-policy \
    --sink-identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-
cdef-example12345
```

输出：

```
{
    "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
    "SinkId": "a1b2c3d4-5678-90ab-cdef-example12345",
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789111:root\"},
\"Action\":[\"oam:CreateLink\",\"oam:UpdateLink\"],\"Resource\":\"*\",
\"Condition\":{\"ForAllValues:StringEquals\":{\"oam:ResourceTypes\":
[\"AWS::Logs::LogGroup\",\"AWS::CloudWatch::Metric\",\"AWS::XRay::Trace\",
\"AWS::ApplicationInsights::Application\"]}}}]}"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetSinkPolicy。

get-sink

以下代码示例演示了如何使用 get-sink。

Amazon CLI

返回有关某个监控账户接收器的完整信息

以下 get-sink 示例返回有关某个监控账户接收器的完整信息。

```
aws oam get-sink \
    --identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

输出：

```
{
    "Arn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345",
    "Id": "a1b2c3d4-5678-90ab-cdef-example12345",
    "Name": "DemoSink",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetSink。

list-attached-links

以下代码示例演示了如何使用 list-attached-links。

Amazon CLI

返回链接到该监控账户接收器的源账户链接的列表

以下 list-attached-links 示例返回链接到该监控账户接收器的源账户链接的列表。

```
aws oam list-attached-links \
    --sink-identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

输出：

```
{
    "Items": [{
        "Label": "Monitoring account",
        "LinkArn": "arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-
example11111",
        "ResourceTypes": [
            "AWS::ApplicationInsights::Application",
            "AWS::Logs::LogGroup",
            "AWS::CloudWatch::Metric",
            "AWS::XRay::Trace"
        ]
    }]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAttachedLinks。

list-links

以下代码示例演示了如何使用 `list-links`。

Amazon CLI

返回某个监控账户接收器的链接列表

以下 `list-links` 示例返回某个监控账户接收器的链接列表。在源账户中运行此操作可返回指向该源账户拥有的监控账户接收器的链接列表。

```
aws oam list-links
```

输出：

```
{
    "Items": [{
        "Arn": "arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-
example11111",
        "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
        "Label": "sourceAccount",
        "ResourceTypes": [
            "AWS::CloudWatch::Metric"
        ],
```

```
        "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345"
    }]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListLinks。

`list-sinks`

以下代码示例演示了如何使用 `list-sinks`。

Amazon CLI

返回在监控账户中创建的接收器的列表

以下 `list-sinks` 示例返回在监控账户中创建的接收器的列表。在监控账户中运行此操作。

```
aws oam list-sinks
```

输出：

```
{
    "Items": [
        {
            "Arn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
            "Id": "a1b2c3d4-5678-90ab-cdef-example12345",
            "Name": "DemoSink"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListSinks。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

显示与资源关联的标签

以下 `list-tags-for-resource` 示例显示与接收器关联的标签。

```
aws oam list-tags-for-resource \
    --resource-arn arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

输出：

```
{
    "Tags": {
        "Team": "Devops"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-sink-policy

以下代码示例演示了如何使用 `put-sink-policy`。

Amazon CLI

创建或更新资源策略

以下 `put-sink-policy` 示例创建资源策略，该策略向源账户授予链接到监控账户接收器的权限。

```
aws oam put-sink-policy \
    --policy '{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":{"AWS":"arn:aws:iam::123456789111:root"},"Action":
["oam:CreateLink","oam:UpdateLink"],"Resource":"*","Condition":
{"ForAllValues:StringEquals":{"oam:ResourceTypes":
["AWS::Logs::LogGroup","AWS::CloudWatch::Metric","AWS::XRay::Trace","AWS::ApplicationInsight
 \
    --sink-identifier arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345
```

输出：

```
{
    "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
    "SinkId": "a1b2c3d4-5678-90ab-cdef-example12345",
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789111:root\"},
\"Action\":[\"oam:CreateLink\",\"oam:UpdateLink\"],\"Resource\":\"*\",
\"Condition\":{\"ForAllValues:StringEquals\":{\"oam:ResourceTypes\":
[\"AWS::Logs::LogGroup\",\"AWS::CloudWatch::Metric\",\"AWS::XRay::Trace\",
\"AWS::ApplicationInsights::Application\"]}}}]}"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutSinkPolicy。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为指定资源分配一个或多个标签

以下 tag-resource 示例标记接收器 arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345。

```
aws oam tag-resource \
    --resource-arn arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345 \
    --tags team=Devops
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从指定资源中移除一个或多个标签

以下 untag-resource 示例从接收器 arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-example12345 中移除键为 team 的标签。

```
aws oam untag-resource \
    --resource-arn arn:aws:oam:us-east-2:123456789012:sink/f3f42f60-f0f2-425c-1234-12347bdd821f \
    --tag-keys team
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-link

以下代码示例演示了如何使用 update-link。

Amazon CLI

更改从源账户共享到其链接的监控账户接收器的数据类型

以下 update-link 示例使用资源类型 AWS::CloudWatch::Metric 和 AWS::Logs::LogGroup 更新链接 arn:aws:oam:us-east-2:123456789111:link/0123e691-e7ef-43fa-1234-c57c837fced0。

```
aws oam update-link \
    --identifier arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-example11111 \
    --resource-types "AWS::CloudWatch::Metric" "AWS::Logs::LogGroup"
```

输出：

```
{
    "Arn": "arn:aws:oam:us-east-2:123456789111:link/a1b2c3d4-5678-90ab-cdef-example11111",
    "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
    "Label": "sourceAccount",
    "LabelTemplate": "sourceAccount",
```

```
    "ResourceTypes": [
        "AWS::CloudWatch::Metric",
        "AWS::Logs::LogGroup"
    ],
    "SinkArn": "arn:aws:oam:us-east-2:123456789012:sink/a1b2c3d4-5678-90ab-cdef-
example12345",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 跨账户可观测性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLink。

使用 Amazon CLI 的 CloudWatch Observability Admin 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 CloudWatch Observability Admin 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`get-telemetry-evaluation-status-for-organization`

以下代码示例演示了如何使用 `get-telemetry-evaluation-status-for-organization`。

Amazon CLI

获取组织的遥测载入状态

以下 `get-telemetry-evaluation-status-for-organization` 示例返回组织的遥测配置功能的当前载入状态。

```
aws observabilityadmin get-telemetry-evaluation-status-for-organization
```

输出：

```
{
    "Status": "RUNNING"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的审计 CloudWatch 遥测配置。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetTelemetryEvaluationStatusForOrganization。

get-telemetry-evaluation-status

以下代码示例演示了如何使用 get-telemetry-evaluation-status。

Amazon CLI

获取账户的遥测载入状态

以下 get-telemetry-evaluation-status 示例返回指定账户中遥测配置功能的当前载入状态。

```
aws observabilityadmin get-telemetry-evaluation-status
```

输出：

```
{
    "Status": "RUNNING"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的审计 CloudWatch 遥测配置。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetTelemetryEvaluationStatus。

list-resource-telemetry-for-organization

以下代码示例演示了如何使用 list-resource-telemetry-for-organization。

Amazon CLI

检索组织的遥测配置

以下 `list-resource-telemetry-for-organization` 示例返回组织中用于遥测配置支持的 Amazon 资源的遥测配置列表。

```
aws observabilityadmin list-resource-telemetry-for-organization \
    --resource-types  AWS::EC2::Instance
```

输出：

```
{
    "TelemetryConfigurations": [
        {
            "AccountIdentifier": "111111111111",
            "TelemetryConfigurationState": {
                "Logs": "NotApplicable",
                "Metrics": "Disabled",
                "Traces": "NotApplicable"
            },
            "ResourceType": "AWS::EC2::Instance",
            "ResourceIdentifier": "i-a166400b",
            "ResourceTags": {
                "Name": "dev"
            },
            "LastUpdateTimeStamp": 1733168548521
        },
        {
            "AccountIdentifier": "222222222222",
            "TelemetryConfigurationState": {
                "Logs": "NotApplicable",
                "Metrics": "Disabled",
                "Traces": "NotApplicable"
            },
            "ResourceType": "AWS::EC2::Instance",
            "ResourceIdentifier": "i-b188560f",
            "ResourceTags": {
                "Name": "apache"
            },
            "LastUpdateTimeStamp": 1732744260182
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的审计 CloudWatch 遥测配置。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceTelemetryForOrganization。

`list-resource-telemetry`

以下代码示例演示了如何使用 `list-resource-telemetry`。

Amazon CLI

检索账户的遥测配置

以下 `list-resource-telemetry` 示例返回指定账户中用于遥测配置支持的 Amazon 资源的遥测配置列表。

```
aws observabilityadmin list-resource-telemetry \
    --resource-types  AWS::EC2::Instance
```

输出：

```
{
    "TelemetryConfigurations": [
        {
            "AccountIdentifier": "111111111111",
            "TelemetryConfigurationState": {
                "Logs": "NotApplicable",
                "Metrics": "Disabled",
                "Traces": "NotApplicable"
            },
            "ResourceType": "AWS::EC2::Instance",
            "ResourceIdentifier": "i-0e979d278b040f856",
            "ResourceTags": {
                "Name": "apache"
            },
            "LastUpdateTimeStamp": 1732744260182
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的审计 CloudWatch 遥测配置。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceTelemetry。

start-telemetry-evaluation-for-organization

以下代码示例演示了如何使用 start-telemetry-evaluation-for-organization。

Amazon CLI

启用遥测配置功能

以下 start-telemetry-evaluation-for-organization 示例为组织启用遥测配置功能。

```
aws observabilityadmin start-telemetry-evaluation-for-organization
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的开启 CloudWatch 遥测审计。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartTelemetryEvaluationForOrganization。

start-telemetry-evaluation

以下代码示例演示了如何使用 start-telemetry-evaluation。

Amazon CLI

启用遥测配置功能

以下 start-telemetry-evaluation 示例在指定账户中启用遥测配置功能。

```
aws observabilityadmin start-telemetry-evaluation
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的开启 CloudWatch 遥测审计。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartTelemetryEvaluation。

stop-telemetry-evaluation-for-organization

以下代码示例演示了如何使用 stop-telemetry-evaluation-for-organization。

Amazon CLI

禁用遥测配置功能

以下 `stop-telemetry-evaluation-for-organization` 示例为组织禁用遥测配置功能。

```
aws observabilityadmin stop-telemetry-evaluation-for-organization
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的关闭 CloudWatch 遥测审计。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopTelemetryEvaluationForOrganization。

`stop-telemetry-evaluation`

以下代码示例演示了如何使用 `stop-telemetry-evaluation`。

Amazon CLI

禁用遥测配置功能

以下 `stop-telemetry-evaluation` 示例在指定账户中禁用遥测配置功能。

```
aws observabilityadmin stop-telemetry-evaluation
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的关闭 CloudWatch 遥测审计。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopTelemetryEvaluation。

使用 Amazon CLI 的 CloudWatch Synthetics 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 CloudWatch Synthetics 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-resource`

以下代码示例演示了如何使用 `associate-resource`。

Amazon CLI

将金丝雀部署与组相关联

以下 `associate-resource` 示例将金丝雀部署与名为 demo_group 的组相关联。

```
aws synthetics associate-resource \
    --group-identifier demo_group \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateResource。

`create-canary`

以下代码示例演示了如何使用 `create-canary`。

Amazon CLI

创建金丝雀部署

以下 `create-canary` 示例创建名为 demo_canary 的金丝雀部署。

```
aws synthetics create-canary \
    --name demo_canary \
    --code '{"S3Bucket": "artifacts3bucket", "S3Key":"demo_canary.zip", "Handler":
 "index.lambda_handler"}' \
```

```
    --artifact-s3-location s3://amzn-s3-demo-bucket/demo_canary.zip \
    --execution-role-arn arn:aws:iam::123456789012:role/demo_canary_role \
    --schedule Expression="rate(10 minutes)" \
    --runtime-version syn-nodejs-puppeteer-9.1
```

输出：

```
{
    "Canary": {
        "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
        "Name": "demo_canary",
        "Code": {
            "Handler": "index.lambda_handler"
        },
        "ExecutionRoleArn": "arn:aws:iam::123456789012:role/demo_canary_role",
        "Schedule": {
            "Expression": "rate(10 minutes)",
            "DurationInSeconds": 0
        },
        "RunConfig": {
            "TimeoutInSeconds": 600,
            "MemoryInMB": 1000,
            "ActiveTracing": false
        },
        "SuccessRetentionPeriodInDays": 31,
        "FailureRetentionPeriodInDays": 31,
        "Status": {
            "State": "CREATING",
            "StateReasonCode": "CREATE_PENDING"
        },
        "Timeline": {
            "Created": "2024-10-15T19:03:08.826000+05:30",
            "LastModified": "2024-10-15T19:03:08.826000+05:30"
        },
        "ArtifactS3Location": "amzn-s3-demo-bucket/demo_canary.zip",
        "RuntimeVersion": "syn-nodejs-puppeteer-9.1",
        "Tags": {}
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateCanary。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

创建组

以下 create-group 示例创建名为 demo_group 的组。

```
aws synthetics create-group \
    --name demo_group
```

输出：

```
{
    "Group": {
        "Id": "example123",
        "Name": "demo_group",
        "Arn": "arn:aws:synthetics:us-east-1:123456789012:group:example123",
        "Tags": {},
        "CreatedTime": "2024-10-15T14:47:23.811000+05:30",
        "LastModifiedTime": "2024-10-15T14:47:23.811000+05:30"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

delete-canary

以下代码示例演示了如何使用 delete-canary。

Amazon CLI

永久删除金丝雀部署

以下 delete-canary 示例删除名为 demo_canary 的金丝雀部署。

```
aws synthetics delete-canary \
    --name demo_canary
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteCanary。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除组

以下 delete-group 示例删除名为 demo_group 的组。

```
aws synthetics delete-group \
    --group-identifier demo_group
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

describe-canaries-last-run

以下代码示例演示了如何使用 describe-canaries-last-run。

Amazon CLI

查看每个金丝雀部署的最近运行的相关信息

以下 describe-canaries-last-run 示例返回您创建的每个金丝雀部署的最近运行的相关信息。

```
aws synthetics describe-canaries-last-run
```

输出：

```
{
    "CanariesLastRun": [
```

```
        {
            "CanaryName": "demo_canary",
            "LastRun": {
                "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
                "Name": "demo_canary",
                "Status": {
                    "State": "PASSED",
                    "StateReason": "",
                    "StateReasonCode": ""
                },
                "Timeline": {
                    "Started": "2024-10-15T19:20:39.691000+05:30",
                    "Completed": "2024-10-15T19:20:58.211000+05:30"
                },
                "ArtifactS3Location": "cw-syn-results-123456789012-us-east-1/canary/
us-east-1/demo_canary-abc-example1234/2024/10/15/13/50-39-690"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeCanariesLastRun。

describe-canaries

以下代码示例演示了如何使用 describe-canaries。

Amazon CLI

列出您账户中的金丝雀部署

以下 describe-canaries 示例列出您账户中的金丝雀部署的详细信息。

```
aws synthetics describe-canaries
```

输出：

```
{
    "Canaries": [
        {
```

```
            "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
            "Name": "demo_canary",
            "Code": {
                "SourceLocationArn": "arn:aws:lambda:us-
east-1:123456789012:layer:cwsyn-demo_canary-a1b2c3d4-5678-90ab-cdef-
example11111b8:1",
                "Handler": "pageLoadBlueprint.handler"
            },
            "ExecutionRoleArn": "arn:aws:iam::123456789012:role/service-role/
CloudWatchSyntheticsRole-demo_canary-a12-a123bc456789",
            "Schedule": {
                "Expression": "rate(5 minutes)",
                "DurationInSeconds": 0
            },
            "RunConfig": {
                "TimeoutInSeconds": 300,
                "MemoryInMB": 1000,
                "ActiveTracing": false
            },
            "SuccessRetentionPeriodInDays": 31,
            "FailureRetentionPeriodInDays": 31,
            "Status": {
            "State": "RUNNING"
            },
            "Timeline": {
                "Created": "2024-10-15T18:55:15.168000+05:30",
                "LastModified": "2024-10-15T18:55:40.540000+05:30",
                "LastStarted": "2024-10-15T18:55:40.540000+05:30"
            },
            "ArtifactS3Location": "cw-syn-results-123456789012-us-east-1/canary/us-
east-1/demo_canary-a12-a123bc456789",
            "EngineArn": "arn:aws:lambda:us-east-1:123456789012:function:cwsyn-
demo_canary-a1b2c3d4-5678-90ab-cdef-example111118:1",
            "RuntimeVersion": "syn-nodejs-puppeteer-9.1",
            "Tags": {
                "blueprint": "heartbeat"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeCanaries。

describe-runtime-versions

以下代码示例演示了如何使用 describe-runtime-versions。

Amazon CLI

返回 Synthetics 金丝雀部署运行时版本的列表

以下 describe-runtime-versions 示例返回 Synthetics 金丝雀部署运行时版本的列表。

```
aws synthetics describe-runtime-versions
```

输出：

```
{
    "RuntimeVersions": [
        {
            "VersionName": "syn-nodejs-puppeteer-9.1",
            "Description": "Security fixes and bug fix for date range error in har.
 Dependencies: Node JS 20.x, Puppeteer-core 22.12.1, Chromium 126.0.6478.126",
            "ReleaseDate": "2024-10-02T05:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-9.0",
            "Description": "Upgraded Chromium and Puppeteer. Dependencies: Node JS
 20.x, Puppeteer-core 22.12.1, Chromium 126.0.6478.126",
            "ReleaseDate": "2024-07-22T05:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-8.0",
            "Description": "Upgraded Chromium and Puppeteer. Dependencies: Node JS
 20.x, Puppeteer-core 22.10.0, Chromium 125.0.6422.112",
            "ReleaseDate": "2024-06-21T05:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-7.0",
            "Description": "Upgraded Chromium and Puppeteer. Dependencies: Node JS
 18.x, Puppeteer-core 21.9.0, Chromium 121.0.6167.139",
            "ReleaseDate": "2024-03-08T05:30:00+05:30"
            },
        {
            "VersionName": "syn-nodejs-puppeteer-6.2",
```

```
            "Description": "Updated shared libraries for Chromium and added
ephemeral storage monitoring. Dependencies: Node JS 18.x, Puppeteer-core 19.7.0,
Chromium 111.0.5563.146",
            "ReleaseDate": "2024-02-02T05:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-6.1",
            "Description": "Added puppeteer launch retry. Dependencies: Node JS
18.x, Puppeteer-core 19.7.0, Chromium 111.0.5563.146",
            "ReleaseDate": "2023-11-13T05:30:00+05:30",
            "DeprecationDate": "2024-03-08T13:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-6.0",
            "Description": "Reduced X-Ray traces of a canary run, improved duration
metric and upgraded to NodeJS 18.x. Dependencies: Node JS 18.x, Puppeteer-core
19.7.0, Chromium 111.0.5563.146",
            "ReleaseDate": "2023-09-15T05:30:00+05:30",
            "DeprecationDate": "2024-03-08T13:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-5.2",
            "Description": "Updated shared libraries for Chromium. Dependencies:
Node JS 16.x, Puppeteer-core 19.7.0, Chromium 111.0.5563.146",
            "ReleaseDate": "2024-02-01T05:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-5.1",
            "Description": "Fixes a bug about missing request headers in har.
Dependencies: Node JS 16.x, Puppeteer-core 19.7.0, Chromium 111.0.5563.146",
            "ReleaseDate": "2023-08-09T05:30:00+05:30",
            "DeprecationDate": "2024-03-08T13:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-5.0",
            "Description": "Upgraded Puppeteer and Chromium. Dependencies: Node JS
16.x, Puppeteer-core 19.7.0, Chromium 111.0.5563.146",
            "ReleaseDate": "2023-07-21T05:30:00+05:30",
            "DeprecationDate": "2024-03-08T13:30:00+05:30"
        },
        {
            "VersionName": "syn-nodejs-puppeteer-4.0",
            "Description": "Upgraded to NodeJS 16.x. Dependencies: Node JS 16.x,
Puppeteer-core 5.5.0, Chromium 92.0.4512.0",
```

```
            "ReleaseDate": "2023-05-01T05:30:00+05:30",
            "DeprecationDate": "2024-03-08T13:30:00+05:30"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeRuntimeVersions。

disassociate-resource

以下代码示例演示了如何使用 disassociate-resource。

Amazon CLI

从组中移除金丝雀部署

以下 disassociate-resource 示例从名为 demo_group 的组中移除金丝雀部署。

```
aws synthetics disassociate-resource \
    --group-identifier demo_group \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateResource。

get-canary-runs

以下代码示例演示了如何使用 get-canary-runs。

Amazon CLI

检索指定金丝雀部署的运行列表

以下 get-canary-runs 示例检索名为 demo_canary 的金丝雀部署的运行列表。

```
aws synthetics get-canary-runs \
```

```
    --name demo_canary
```

输出：

```
{
    "CanaryRuns": [
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
            "Name": "demo_canary",
            "Status": {
                "State": "PASSED",
                "StateReason": "",
                "StateReasonCode": ""
            },
            "Timeline": {
                "Started": "2024-10-16T10:38:57.013000+05:30",
                "Completed": "2024-10-16T10:39:25.793000+05:30"
            },
            "ArtifactS3Location": "cw-syn-results-123456789012-us-east-1/canary/us-
east-1/demo_canary-abc-example1234/2024/10/15/13/50-39-690"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetCanaryRuns。

get-canary

以下代码示例演示了如何使用 get-canary。

Amazon CLI

检索有关某个金丝雀部署的完整信息

以下 get-canary 示例检索有关名为 demo_canary 的金丝雀部署的完整信息。

```
aws synthetics get-canary \
    --name demo_canary
```

输出：

```
{
    "Canary": {
        "Id": "a1b2c3d4-5678-90ab-cdef-example11111",
        "Name": "demo_canary",
        "Code": {
            "SourceLocationArn": "arn:aws:lambda:us-east-1:123456789012:layer:cwsyn-
demo_canary-a1b2c3d4-5678-90ab-cdef-example111118:1",
            "Handler": "pageLoadBlueprint.handler"
        },
        "ExecutionRoleArn": "arn:aws:iam::123456789012:role/demo_canary_role",
        "Schedule": {
            "Expression": "rate(10 minutes)",
            "DurationInSeconds": 0
        },
        "RunConfig": {
            "TimeoutInSeconds": 300,
            "MemoryInMB": 1000,
            "ActiveTracing": false
        },
        "SuccessRetentionPeriodInDays": 31,
        "FailureRetentionPeriodInDays": 31,
        "Status": {
            "State": "RUNNING"
        },
        "Timeline": {
            "Created": "2024-10-15T18:55:15.168000+05:30",
            "LastModified": "2024-10-15T18:55:40.540000+05:30",
            "LastStarted": "2024-10-15T18:55:40.540000+05:30"
        },
        "ArtifactS3Location": "cw-syn-results-123456789012-us-east-1/canary/us-
east-1/demo_canary-a12-a123bc456789",
        "EngineArn": "arn:aws:lambda:us-east-1:123456789012:function:cwsyn-
demo_canary-a1b2c3d4-5678-90ab-cdef-example111118:1",
        "RuntimeVersion": "syn-nodejs-puppeteer-9.1",
        "Tags": {
            "blueprint": "heartbeat"
        }
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetCanary。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

返回有关某个组的信息

以下 get-group 示例返回有关名为 demo_group 的组的信息。

```
aws synthetics get-group \
    --group-identifier demo_group
```

输出：

```
{
    "Group": {
        "Id": "example123",
        "Name": "demo_group",
        "Arn": "arn:aws:synthetics:us-east-1:123456789012:group:example123",
        "Tags": {},
        "CreatedTime": "2024-10-15T14:47:23.811000+05:30",
        "LastModifiedTime": "2024-10-15T14:47:23.811000+05:30"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

list-associated-groups

以下代码示例演示了如何使用 list-associated-groups。

Amazon CLI

返回组列表

以下 list-associated-groups 示例返回与名为 demo_canary 的金丝雀部署关联的组的列表。

```
aws synthetics list-associated-groups \
```

```
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary
```

输出：

```
{
    "Groups": [
        {
            "Id": "example123",
            "Name": "demo_group",
            "Arn": "arn:aws:synthetics:us-east-1:123456789012:group:example123"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAssociatedGroups。

`list-group-resources`

以下代码示例演示了如何使用 `list-group-resources`。

Amazon CLI

返回与指定组关联的金丝雀部署的 ARN 列表

以下 `list-group-resources` 示例返回与名为 demo_group 的组关联的金丝雀部署的 ARN 列表。

```
aws synthetics list-group-resources \
    --group-identifier demo_group
```

输出：

```
{
    "Resources": [
        "arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary"
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupResources。

list-groups

以下代码示例演示了如何使用 list-groups。

Amazon CLI

返回账户中所有组的列表

以下 list-groups 示例返回账户中所有组的列表。

```
aws synthetics list-groups
```

输出：

```
{
    "Groups": [
        {
            "Id": "example123",
            "Name": "demo_group",
            "Arn": "arn:aws:synthetics:us-east-1:123456789012:group:example123"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

示例 1：显示与金丝雀部署关联的标签

以下 list-tags-for-resource 示例返回与名为 demo_canary 的金丝雀部署关联的标签。

```
aws synthetics list-tags-for-resource \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary
```

输出：

```
{
    "Tags": {
        "blueprint": "heartbeat"
    }
}
```

示例 2：显示与组关联的标签

以下 list-tags-for-resource 示例返回与名为 demo_group 的组关联的标签。

```
aws  synthetics list-tags-for-resource \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:group:example123
```

输出：

```
{
    "Tags": {
        "team": "Devops"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-canary

以下代码示例演示了如何使用 start-canary。

Amazon CLI

运行金丝雀部署

以下 start-canary 示例运行名为 demo_canary 的金丝雀部署。

```
aws synthetics start-canary \
    --name demo_canary
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartCanary。

stop-canary

以下代码示例演示了如何使用 stop-canary。

Amazon CLI

停止金丝雀部署

以下 stop-canary 示例停止名为 demo_canary 的金丝雀部署。

```
aws synthetics stop-canary \
    --name demo_canary
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopCanary。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

示例 1：为金丝雀部署分配标签

以下 tag-resource 示例为名为 demo_canary 的金丝雀部署分配标签。

```
aws synthetics tag-resource \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary \
    --tags blueprint=heartbeat
```

此命令不生成任何输出。

示例 2：为组分配标签

以下 tag-resource 示例为名为 demo_group 的组分配标签。

```
aws synthetics tag-resource \
```

```
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:group:example123 \
    --tags team=Devops
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：从金丝雀部署中移除标签

以下 untag-resource 示例从名为 demo_canary 的金丝雀部署中移除标签。

```
aws synthetics untag-resource \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:canary:demo_canary \
    --tag-keys blueprint
```

此命令不生成任何输出。

示例 2：从组中移除标签

以下 untag-resource 示例从名为 demo_group 的组中移除标签。

```
aws synthetics untag-resource \
    --resource-arn arn:aws:synthetics:us-east-1:123456789012:group:example123 \
    --tag-keys team
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-canary

以下代码示例演示了如何使用 update-canary。

Amazon CLI

更新金丝雀部署

以下 update-canary 示例更新名为 demo_canary 的金丝雀部署的配置。

```
aws synthetics update-canary \
    --name demo_canary \
    --schedule Expression="rate(15 minutes)"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的综合监控（金丝雀部署）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateCanary。

使用 Amazon CLI 的 CodeArtifact 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeArtifact 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-external-connection`

以下代码示例演示了如何使用 associate-external-connection。

Amazon CLI

向存储库添加外部连接

以下 associate-external-connection 示例会将指向 npmjs.com 的外部连接添加到名为 test-repo 的存储库中。

```
aws codeartifact associate-external-connection \
    --repository test-repo \
    --domain test-domain \
    --external-connection public:npmjs
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "upstreams": [],
        "externalConnections": [
            {
                "externalConnectionName": "public:npmjs",
                "packageFormat": "npm",
                "status": "AVAILABLE"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的添加外部连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateExternalConnection。

copy-package-versions

以下代码示例演示了如何使用 copy-package-versions。

Amazon CLI

将软件包版本从一个存储库复制到另一个存储库

以下 copy-package-versions 会将名为 test-package 的软件包的 4.0.0 和 5.0.0 版本从 my-repo 移动至 test-repo。

```
aws codeartifact copy-package-versions \
```

```
    --domain test-domain \
    --source-repository my-repo \
    --destination-repository test-repo \
    --format npm \
    --package test-package \
    --versions '["4.0.0", "5.0.0"]'
```

输出：

```
{
    "format": "npm",
    "package": "test-package",
    "versions": [
        {
        "version": "5.0.0",
        "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
        "status": "Published"
        },
        {
        "version": "4.0.0",
        "revision": "REVISION-2-SAMPLE-55C752BEE772FC",
        "status": "Published"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的在存储库之间复制软件包。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyPackageVersions。

create-domain

以下代码示例演示了如何使用 create-domain。

Amazon CLI

创建域

以下 create-domain 示例创建一个名为 test-domain 的域。

```
aws codeartifact create-domain \
    --domain test-domain
```

输出：

```
{
    "domain": {
        "name": "test-domain",
        "owner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/test-domain",
        "status": "Active",
        "createdTime": "2020-10-20T13:16:48.559000-04:00",
        "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "repositoryCount": 0,
        "assetSizeBytes": 0
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的创建域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomain。

create-repository

以下代码示例演示了如何使用 create-repository。

Amazon CLI

创建存储库 ()

以下 create-repository 示例在名为 test-domain 的域中创建一个名为 test-repo 的存储库。

```
aws codeartifact create-repository \
    --domain test-domain \
    --domain-owner 111122223333 \
    --repository test-repo \
    --description "This is a test repository."
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
```

```
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "description": "This is a test repository.",
        "upstreams": [],
        "externalConnections": []
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的创建域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRepository。

delete-domain-permissions-policy

以下代码示例演示了如何使用 delete-domain-permissions-policy。

Amazon CLI

从域中删除权限策略文档

以下 delete-domain-permissions-policy 示例从名为 test-domain 的域中删除权限策略。

```
aws codeartifact delete-domain-permissions-policy \
    --domain test-domain
```

输出：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "BasicDomainPolicy",
            "Action": [
                "codeartifact:GetDomainPermissionsPolicy",
                "codeartifact:ListRepositoriesInDomain",
                "codeartifact:GetAuthorizationToken",
                "codeartifact:CreateRepository"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::111122223333:root"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的删除域策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomainPermissionsPolicy。

`delete-domain`

以下代码示例演示了如何使用 `delete-domain`。

Amazon CLI

删除域

以下 `delete-domain` 示例删除名为 `test-domain` 的域。

```
aws codeartifact delete-domain \
    --domain test-domain
```

输出：

```
{
    "domain": {
        "name": "test-domain",
        "owner": "417498243647",
        "arn": "arn:aws:codeartifact:us-west-2:417498243647:domain/test-domain",
        "status": "Deleted",
        "createdTime": "2020-10-20T13:16:48.559000-04:00",
        "encryptionKey": "arn:aws:kms:us-west-2:417498243647:key/c9fe2447-0795-4fda-
afbe-8464574ae162",
        "repositoryCount": 0,
        "assetSizeBytes": 0
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的删除域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomain。

delete-package-versions

以下代码示例演示了如何使用 delete-package-versions。

Amazon CLI

删除软件包版本

以下 delete-package-versions 示例删除名为 test-package 的软件包的 4.0.0 版本。

```
aws codeartifact delete-package-versions \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --versions 4.0.0
```

输出：

```
{
    "successfulVersions": {
        "4.0.0": {
            "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
            "status": "Deleted"
        }
    },
    "failedVersions": {}
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的删除软件包版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePackageVersions。

delete-repository-permissions-policy

以下代码示例演示了如何使用 delete-repository-permissions-policy。

Amazon CLI

从存储库中删除权限策略

以下 delete-repository-permissions-policy 示例从名为 test-repo 的存储库中删除权限策略。

```
aws codeartifact delete-repository-permissions-policy \
    --domain test-domain \
    --repository test-repo
```

输出：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": [
                "codeartifact:DescribePackageVersion",
                "codeartifact:DescribeRepository",
                "codeartifact:GetPackageVersionReadme",
                "codeartifact:GetRepositoryEndpoint",
                "codeartifact:ListPackages",
                "codeartifact:ListPackageVersions",
                "codeartifact:ListPackageVersionAssets",
                "codeartifact:ListPackageVersionDependencies",
                "codeartifact:ReadFromRepository"
            ],
            "Resource": "*"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的删除策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepositoryPermissionsPolicy。

delete-repository

以下代码示例演示了如何使用 delete-repository。

Amazon CLI

删除存储库（ ）

以下 delete-repository 示例在名为 test-domain 的域中删除名为 test-repo 的存储库。

```
aws codeartifact delete-repository \
    --domain test-domain \
    --repository test-repo
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "description": "This is a test repository",
        "upstreams": [],
        "externalConnections": []
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的删除存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepository。

describe-domain

以下代码示例演示了如何使用 describe-domain。

Amazon CLI

获取有关域的信息

以下 describe-domain 示例返回名为 test-domain 的域的 DomainDescription 对象。

```
aws codeartifact describe-domain \
    --domain test-domain
```

输出：

```
{
```

```
    "domain": {
        "name": "test-domain",
        "owner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/test-domain",
        "status": "Active",
        "createdTime": "2020-10-20T13:16:48.559000-04:00",
        "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "repositoryCount": 2,
        "assetSizeBytes": 0,
        "s3BucketArn": "arn:aws:s3:::assets-111122223333-us-west-2"
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的域概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDomain。

describe-repository

以下代码示例演示了如何使用 describe-repository。

Amazon CLI

获取有关存储库的信息

以下 describe-repository 示例返回名为 test-repo 的存储库的 RepositoryDescription 对象。

```
aws codeartifact describe-repository \
    --domain test-domain \
    --repository test-repo
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "description": "This is a test repository.",
```

```
        "upstreams": [],
        "externalConnections": []
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的创建域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRepository。

`disassociate-external-connection`

以下代码示例演示了如何使用 `disassociate-external-connection`。

Amazon CLI

从存储库移除外部连接

以下 `disassociate-external-connection` 示例从名为 test-repo 的存储库中移除指向 npmjs.com 的外部连接。

```
aws codeartifact disassociate-external-connection \
    --repository test-repo \
    --domain test-domain \
    --external-connection public:npmjs
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "upstreams": [],
        "externalConnections": []
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的移除外部连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateExternalConnection。

dispose-package-versions

以下代码示例演示了如何使用 dispose-package-versions。

Amazon CLI

删除软件包版本的资产并将其状态设置为“已处置”

以下 dispose-package-versions 示例删除测试软件包 4.0.0 版本的资产并将其状态设置为“已处置”。

```
aws codeartifact dispose-package-versions \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --versions 4.0.0
```

输出：

```
{
    "successfulVersions": {
        "4.0.0": {
            "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
            "status": "Disposed"
        }
    },
    "failedVersions": {}
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的在 CodeArtifact 中使用软件包。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisposePackageVersions。

get-authorization-token

以下代码示例演示了如何使用 get-authorization-token。

Amazon CLI

获取授权令牌

以下 get-authorization-token 示例检索 CodeArtifact 授权令牌。

```
aws codeartifact get-authorization-token \
    --domain test-domain \
    --query authorizationToken \
    --output text
```

输出：

```
This command will return the authorization token. You can store the output in an
 environment variable when calling the command.
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的不使用登录命令配置 pip。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizationToken。

get-domain-permissions-policy

以下代码示例演示了如何使用 get-domain-permissions-policy。

Amazon CLI

获取域的权限策略文档

以下 get-domain-permissions-policy 示例获取名为 test-domain 的域的权限策略。

```
aws codeartifact get-domain-permissions-policy \
    --domain test-domain
```

输出：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "BasicDomainPolicy",
            "Action": [
                "codeartifact:GetDomainPermissionsPolicy",
                "codeartifact:ListRepositoriesInDomain",
                "codeartifact:GetAuthorizationToken",
                "codeartifact:CreateRepository"
            ],
```

```
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的读取域策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainPermissionsPolicy。

get-package-version-asset

以下代码示例演示了如何使用 get-package-version-asset。

Amazon CLI

从软件包版本中获取资产

以下 get-package-version-asset 示例检索名为 test-package 的 npm 软件包 4.0.0 版的 package.tgz 资产。

```
aws codeartifact get-package-version-asset \
    --domain test-domain \
    --repository test-repo \
    --format npm \
    --package test-package \
    --package-version 4.0.0 \
    --asset 'package.tgz' \
    outfileName
```

输出：

```
The output for this command will also store the raw asset in the file provided in
 place of outfileName.

{
    "assetName": "package.tgz",
    "packageVersion": "4.0.0",
    "packageVersionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs="
```

```
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出软件包版本资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPackageVersionAsset。

get-package-version-readme

以下代码示例演示了如何使用 get-package-version-readme。

Amazon CLI

获取软件包版本的自述文件

以下 get-package-version-readme 示例检索名为 test-package 的 npm 软件包 4.0.0 版的自述文件。

```
aws codeartifact get-package-version-readme \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --package-version 4.0.0
```

输出：

```
{
    "format": "npm",
    "package": "test-package",
    "version": "4.0.0",
    "readme": "<div align=\"center\">\n   <a href=\https://github.com/test-package/
testpack\"> ... more content ... \n",
    "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs="
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的查看软件包版本的自述文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPackageVersionReadme。

get-repository-endpoint

以下代码示例演示了如何使用 get-repository-endpoint。

Amazon CLI

获取存储库的 URL 端点

以下 `get-repository-endpoint` 示例返回 test-repo 存储库的 npm 端点。

```
aws codeartifact get-repository-endpoint \
    --domain test-domain \
    --repository test-repo \
    --format npm
```

输出：

```
{
    "repositoryEndpoint": "https://test-domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/npm/test-repo/"
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的连接到存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepositoryEndpoint。

get-repository-permissions-policy

以下代码示例演示了如何使用 `get-repository-permissions-policy`。

Amazon CLI

获取存储库的权限策略文档

以下 `get-repository-permissions-policy` 示例获取名为 test-repo 的存储库的权限策略。

```
aws codeartifact get-repository-permissions-policy \
    --domain test-domain \
    --repository test-repo
```

输出：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": [
                "codeartifact:DescribePackageVersion",
                "codeartifact:DescribeRepository",
                "codeartifact:GetPackageVersionReadme",
                "codeartifact:GetRepositoryEndpoint",
                "codeartifact:ListPackages",
                "codeartifact:ListPackageVersions",
                "codeartifact:ListPackageVersionAssets",
                "codeartifact:ListPackageVersionDependencies",
                "codeartifact:ReadFromRepository"
            ],
            "Resource": "*"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的读取策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepositoryPermissionsPolicy。

`list-domains`

以下代码示例演示了如何使用 `list-domains`。

Amazon CLI

列出域

以下 `list-domains` 示例返回发起调用的 Amazon 账户拥有的所有域的摘要。

```
aws codeartifact list-domains
```

输出：

```
{
    "domains": [
        {
            "name": "my-domain",
            "owner": "111122223333",
```

```
            "status": "Active",
            "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        },
        {
            "name": "test-domain",
            "owner": "111122223333",
            "status": "Active",
            "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的在 CodeArtifact 中使用域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDomains。

`list-package-version-assets`

以下代码示例演示了如何使用 `list-package-version-assets`。

Amazon CLI

查看软件包版本的资产

以下 `list-package-version-assets` 示例检索名为 test-package 的 npm 软件包 4.0.0 版的资产。

```
aws codeartifact list-package-version-assets \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --package-version 4.0.0
```

输出：

```
{
    "format": "npm",
    "package": "test-package",
    "version": "4.0.0",
    "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
```

```
    "assets": [
        {
            "name": "package.tgz",
            "size": 316680,
            "hashes": {
                "MD5": "60078ec6d9e76b89fb55c860832742b2",
                "SHA-1": "b44a9b6297bcb698f1c51a3545a2b3b368d59c52",
                "SHA-256":
 "d2aa8c6afc3c8591765785a37d1c5acae482a8eb3ab9729ed28922692454f2e2",
                "SHA-512":
 "3e585d15c8a594e20d7de57b362ea81754c011acb2641a19f1b72c8531ea39825896bab344ae616a0a5a824cb9
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出软件包版本资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackageVersionAssets。

`list-package-version-dependencies`

以下代码示例演示了如何使用 `list-package-version-dependencies`。

Amazon CLI

查看软件包版本的依赖项

以下 `list-package-version-dependencies` 示例检索名为 test-package 的 npm 软件包 4.0.0 版的依赖项。

```
aws codeartifact list-package-version-dependencies \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --package-version 4.0.0
```

输出：

```
{
    "format": "npm",
    "package": "test-package",
```

```
    "version": "4.0.0",
    "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
    "dependencies": [
        {
            "namespace": "testns",
            "package": "testdep1",
            "dependencyType": "regular",
            "versionRequirement": "1.8.5"
        },
        {
            "namespace": "testns",
            "package": "testdep2",
            "dependencyType": "regular",
            "versionRequirement": "1.8.5"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的查看并更新软件包版本的详细信息和依赖项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackageVersionDependencies。

`list-package-versions`

以下代码示例演示了如何使用 `list-package-versions`。

Amazon CLI

列出软件包的软件包版本

以下 `list-package-versions` 示例返回名为 `kind-of` 的软件包的软件包版本列表。

```
aws codeartifact list-package-versions \
    --package kind-of \
    --domain test-domain \
    --repository test-repo \
    --format npm
```

输出：

```
{
```

```
    "defaultDisplayVersion": "1.0.1",
    "format": "npm",
    "package": "kind-of",
    "versions": [
        {
            "version": "1.0.1",
            "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
            "status": "Published"
        },
        {
            "version": "1.0.0",
            "revision": "REVISION-SAMPLE-2-C752BEEF6D2CFC",
            "status": "Published"
        },
        {
            "version": "0.1.2",
            "revision": "REVISION-SAMPLE-3-654S65A5C5E1FC",
            "status": "Published"
        },
        {
            "version": "0.1.1",
            "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC"",
            "status": "Published"
        },
        {
            "version": "0.1.0",
            "revision": "REVISION-SAMPLE-4-AF669139B772FC",
            "status": "Published"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出软件包版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackageVersions。

`list-packages`

以下代码示例演示了如何使用 `list-packages`。

Amazon CLI

列出存储库中的软件包

以下 list-packages 示例列出名为 test-domain 的域中名为 test-repo 的存储库中的软件包。

```
aws codeartifact list-packages \
    --domain test-domain \
    --repository test-repo
```

输出：

```
{
    "packages": [
        {
            "format": "npm",
            "package": "lodash"
        }
        {
            "format": "python",
            "package": "test-package"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出软件包名称。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackages。

list-repositories-in-domain

以下代码示例演示了如何使用 list-repositories-in-domain。

Amazon CLI

列出域中的存储库

以下 list-repositories-in-domain 示例返回 test-domain 域中所有存储库的摘要。

```
aws codeartifact list-repositories-in-domain \
    --domain test-domain
```

输出：

```
{
```

```
    "repositories": [
        {
            "name": "test-repo",
            "administratorAccount": "111122223333",
            "domainName": "test-domain",
            "domainOwner": "111122223333",
            "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-
domain/test-repo",
            "description": "This is a test repository."
        },
        {
            "name": "test-repo2",
            "administratorAccount": "111122223333",
            "domainName": "test-domain",
            "domainOwner": "111122223333",
            "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-
domain/test-repo2",
            "description": "This is a test repository."
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRepositoriesInDomain。

`list-repositories`

以下代码示例演示了如何使用 `list-repositories`。

Amazon CLI

列出存储库

以下 `list-repositories` 示例返回发起调用的 Amazon 账户拥有的所有域中存储库的摘要。

```
aws codeartifact list-repositories
```

输出：

```
{
    "repositories": [
        {
```

```
            "name": "npm-store",
            "administratorAccount": "111122223333",
            "domainName": "my-domain",
            "domainOwner": "111122223333",
            "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-
domain/npm-store",
            "description": "Provides npm artifacts from npm, Inc."
        },
        {
            "name": "target-repo",
            "administratorAccount": "111122223333",
            "domainName": "my-domain",
            "domainOwner": "111122223333",
            "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-
domain/target-repo",
            "description": "test target repo"
        },
        {
            "name": "test-repo2",
            "administratorAccount": "111122223333",
            "domainName": "test-domain",
            "domainOwner": "111122223333",
            "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-
domain/test-repo2",
            "description": "This is a test repository."
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的列出存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRepositories。

`login`

以下代码示例演示了如何使用 `login`。

Amazon CLI

使用登录命令配置对存储库的身份验证

以下 `login` 示例在名为 test-domain 的域中配置一个带有名为 test-repo 的存储库的 npm 软件包管理器。

```
aws codeartifact login \
    --domain test-domain \
    --repository test-repo \
    --tool npm
```

输出：

```
Successfully configured npm to use AWS CodeArtifact repository https://test-
domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/test-repo/
Login expires in 12 hours at 2020-11-12 01:53:16-05:00
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Login。

put-domain-permissions-policy

以下代码示例演示了如何使用 put-domain-permissions-policy。

Amazon CLI

将权限策略附加到域

以下 put-domain-permissions-policy 示例会将 policy.json 文件中定义的权限策略附加到名为 test-domain 的域。

```
aws codeartifact put-domain-permissions-policy \
    --domain test-domain \
    --policy-document file://PATH/TO/policy.json
```

输出：

```
{
    "policy": {
        "resourceArn": "arn:aws:codeartifact:region-id:111122223333:domain/test-
domain",
        "document": "{ ...policy document content...}",
        "revision": "MQlyyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxx="
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的设置域策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutDomainPermissionsPolicy。

put-repository-permissions-policy

以下代码示例演示了如何使用 put-repository-permissions-policy。

Amazon CLI

将权限策略附加到存储库

以下 put-repository-permissions-policy 示例会将 policy.json 文件中定义的权限策略附加到名为 test-repo 的存储库。

```
aws codeartifact put-repository-permissions-policy \
    --domain test-domain \
    --repository test-repo \
    --policy-document file://PATH/TO/policy.json
```

输出：

```
{
    "policy": {
        "resourceArn": "arn:aws:codeartifact:region-id:111122223333:repository/test-
domain/test-repo",
        "document": "{ ...policy document content...}",
        "revision": "MQlyyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxx="
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的设置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRepositoryPermissionsPolicy。

update-package-versions-status

以下代码示例演示了如何使用 update-package-versions-status。

Amazon CLI

更新软件包版本状态

以下 update-package-versions-status 示例会将 test-package 软件包 4.0.0 版本的状态更新为 Archived。

```
aws codeartifact update-package-versions-status \
    --domain test-domain \
    --repo test-repo \
    --format npm \
    --package test-package \
    --versions 4.0.0 \
    --target-status Archived
```

输出：

```
{
    "successfulVersions": {
        "4.0.0": {
            "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
            "status": "Archived"
        }
    },
    "failedVersions": {}
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的更新软件包版本状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePackageVersionsStatus。

update-repository

以下代码示例演示了如何使用 update-repository。

Amazon CLI

更新存储库

以下 update-repository 示例会将名为 test-domain 的域中名为 test-repo 的存储库的描述更新为“这是更新的描述”。

```
aws codeartifact update-repository \
    --domain test-domain \
    --repository test-repo \
```

```
    --description "this is an updated description"
```

输出：

```
{
    "repository": {
        "name": "test-repo",
        "administratorAccount": "111122223333",
        "domainName": "test-domain",
        "domainOwner": "111122223333",
        "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
        "description": "this is an updated description",
        "upstreams": [],
        "externalConnections": []
    }
}
```

有关更多信息，请参阅《Amazon CodeArtifact 用户指南》中的查看或修改存储库配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRepository。

使用 Amazon CLI 的 CodeBuild 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeBuild 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-delete-builds`

以下代码示例演示了如何使用 `batch-delete-builds`。

Amazon CLI

在 Amazon CodeBuild 中删除构建

以下 batch-delete-builds 示例删除 CodeBuild 中具有指定 ID 的构建。

```
aws codebuild batch-delete-builds --ids my-build-project-one:a1b2c3d4-5678-9012-abcd-11111EXAMPLE my-build-project-two:a1b2c3d4-5678-9012-abcd-22222EXAMPLE
```

输出：

```
{
    "buildsNotDeleted": [
        {
            "id": "arn:aws:codebuild:us-west-2:123456789012:build/my-build-project-one:a1b2c3d4-5678-9012-abcd-11111EXAMPLE",
            "statusCode": "BUILD_IN_PROGRESS"
        }
    ],
    "buildsDeleted": [
        "arn:aws:codebuild:us-west-2:123456789012:build/my-build-project-two:a1b2c3d4-5678-9012-abcd-22222EXAMPLE"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的删除构建（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteBuilds。

batch-get-build-batches

以下代码示例演示了如何使用 batch-get-build-batches。

Amazon CLI

在 Amazon CodeBuild 中查看构建的详细信息

以下 batch-get-build-batches 示例获取有关使用指定 ID 在 CodeBuild 中批量构建的信息。

```
aws codebuild batch-get-build-batches \
    --ids codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE
```

输出：

```
{
    "buildBatches": [
        {
            "id": "codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
            "arn": "arn:aws:codebuild:us-west-2:123456789012:build-batch/codebuild-
demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
            "startTime": "2020-11-03T21:52:20.775000+00:00",
            "endTime": "2020-11-03T21:56:59.784000+00:00",
            "currentPhase": "SUCCEEDED",
            "buildBatchStatus": "SUCCEEDED",
            "resolvedSourceVersion": "0a6546f68309560d08a310daac92314c4d378f6b",
            "projectName": "codebuild-demo-project",
            "phases": [
                {
                    "phaseType": "SUBMITTED",
                    "phaseStatus": "SUCCEEDED",
                    "startTime": "2020-11-03T21:52:20.775000+00:00",
                    "endTime": "2020-11-03T21:52:20.976000+00:00",
                    "durationInSeconds": 0
                },
                {
                    "phaseType": "DOWNLOAD_BATCHSPEC",
                    "phaseStatus": "SUCCEEDED",
                    "startTime": "2020-11-03T21:52:20.976000+00:00",
                    "endTime": "2020-11-03T21:52:57.401000+00:00",
                    "durationInSeconds": 36
                },
                {
                    "phaseType": "IN_PROGRESS",
                    "phaseStatus": "SUCCEEDED",
                    "startTime": "2020-11-03T21:52:57.401000+00:00",
                    "endTime": "2020-11-03T21:56:59.751000+00:00",
                    "durationInSeconds": 242
                },
                {
                    "phaseType": "COMBINE_ARTIFACTS",
                    "phaseStatus": "SUCCEEDED",
                    "startTime": "2020-11-03T21:56:59.751000+00:00",
                    "endTime": "2020-11-03T21:56:59.784000+00:00",
                    "durationInSeconds": 0
                },
                {
```

```
            "phaseType": "SUCCEEDED",
            "startTime": "2020-11-03T21:56:59.784000+00:00"
        }
    ],
    "source": {
        "type": "GITHUB",
        "location": "https://github.com/my-repo/codebuild-demo-project.git",
        "gitCloneDepth": 1,
        "gitSubmodulesConfig": {
            "fetchSubmodules": false
        },
        "reportBuildStatus": false,
        "insecureSsl": false
    },
    "secondarySources": [],
    "secondarySourceVersions": [],
    "artifacts": {
        "location": ""
    },
    "secondaryArtifacts": [],
    "cache": {
        "type": "NO_CACHE"
    },
    "environment": {
        "type": "LINUX_CONTAINER",
        "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
        "computeType": "BUILD_GENERAL1_SMALL",
        "environmentVariables": [],
        "privilegedMode": false,
        "imagePullCredentialsType": "CODEBUILD"
    },
    "logConfig": {
        "cloudWatchLogs": {
            "status": "ENABLED"
        },
        "s3Logs": {
            "status": "DISABLED",
            "encryptionDisabled": false
        }
    },
    "buildTimeoutInMinutes": 60,
    "queuedTimeoutInMinutes": 480,
    "complete": true,
    "initiator": "Strohm",
```

```
            "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
            "buildBatchNumber": 6,
            "buildBatchConfig": {
                "serviceRole": "arn:aws:iam::123456789012:role/service-role/
codebuild-demo-project",
                "restrictions": {
                    "maximumBuildsAllowed": 100
                },
                "timeoutInMins": 480
            },
            "buildGroups": [
                {
                    "identifier": "DOWNLOAD_SOURCE",
                    "ignoreFailure": false,
                    "currentBuildSummary": {
                        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:379737d8-bc35-48ec-97fd-776d27545315",
                        "requestedOn": "2020-11-03T21:52:21.394000+00:00",
                        "buildStatus": "SUCCEEDED",
                        "primaryArtifact": {
                            "type": "no_artifacts",
                            "identifier": "DOWNLOAD_SOURCE"
                        },
                        "secondaryArtifacts": []
                    }
                },
                {
                    "identifier": "linux_small",
                    "dependsOn": [],
                    "ignoreFailure": false,
                    "currentBuildSummary": {
                        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:dd785171-ed84-4bb6-8ede-ceeb86e54bdb",
                        "requestedOn": "2020-11-03T21:52:57.604000+00:00",
                        "buildStatus": "SUCCEEDED",
                        "primaryArtifact": {
                            "type": "no_artifacts",
                            "identifier": "linux_small"
                        },
                        "secondaryArtifacts": []
                    }
                },
                {
                    "identifier": "linux_medium",
```

```
                    "dependsOn": [
                        "linux_small"
                    ],
                    "ignoreFailure": false,
                    "currentBuildSummary": {
                        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:97cf7bd4-5313-4786-8243-4aef350a1267",
                        "requestedOn": "2020-11-03T21:54:18.474000+00:00",
                        "buildStatus": "SUCCEEDED",
                        "primaryArtifact": {
                            "type": "no_artifacts",
                            "identifier": "linux_medium"
                        },
                        "secondaryArtifacts": []
                    }
                },
                {
                    "identifier": "linux_large",
                    "dependsOn": [
                        "linux_medium"
                    ],
                    "ignoreFailure": false,
                    "currentBuildSummary": {
                        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:60a194cd-0d03-4337-9db1-d41476a17d27",
                        "requestedOn": "2020-11-03T21:55:39.203000+00:00",
                        "buildStatus": "SUCCEEDED",
                        "primaryArtifact": {
                            "type": "no_artifacts",
                            "identifier": "linux_large"
                        },
                        "secondaryArtifacts": []
                    }
                }
            ]
        }
    ],
    "buildBatchesNotFound": []
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的“在 Amazon CodeBuild 中批量构建 <https://docs.aws.amazon.com/codebuild/latest/userguide/batch-build.html>)__”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetBuildBatches。

batch-get-builds

以下代码示例演示了如何使用 batch-get-builds。

Amazon CLI

在 Amazon CodeBuild 中查看构建的详细信息

以下 batch-get-builds 示例获取 CodeBuild 中指定 ID 的构建信息。

```
aws codebuild batch-get-builds --ids codebuild-demo-project:e9c4f4df-3f43-41d2-
ab3a-60fe2EXAMPLE codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE
```

输出：

```
{
    "buildsNotFound": [],
    "builds": [
        {
            "artifacts": {
                "md5sum": "0e95edf915048a0c22efe6d139fff837",
                "location": "arn:aws:s3:::codepipeline-us-west-2-820783811474/
CodeBuild-Python-Pip/BuildArtif/6DJsqQa",
                "encryptionDisabled": false,
                "sha256sum":
 "cfa0df33a090966a737f64ae4fe498969fdc842a0c9aec540bf93c37ac0d05a2"
            },
            "logs": {
                "cloudWatchLogs": {
                    "status": "ENABLED"
                },
                "s3Logs": {
                    "status": "DISABLED"
                },
                "streamName": "46472baf-8f6b-43c2-9255-b3b963af2732",
                "groupName": "/aws/codebuild/codebuild-demo-project",
                "deepLink": "https://console.aws.amazon.com/cloudwatch/
home?region=us-west-2#logEvent:group=/aws/codebuild/codebuild-demo-
project;stream=46472baf-8f6b-43c2-9255-b3b963af2732"
            },
            "timeoutInMinutes": 60,
            "environment": {
                "privilegedMode": false,
                "computeType": "BUILD_GENERAL1_MEDIUM",
```

```
            "image": "aws/codebuild/windows-base:1.0",
            "environmentVariables": [],
            "type": "WINDOWS_CONTAINER"
        },
        "projectName": "codebuild-demo-project",
        "buildComplete": true,
        "source": {
            "gitCloneDepth": 1,
            "insecureSsl": false,
            "type": "CODEPIPELINE"
        },
        "buildStatus": "SUCCEEDED",
        "secondaryArtifacts": [],
        "phases": [
            {
                "durationInSeconds": 0,
                "startTime": 1548717462.122,
                "phaseType": "SUBMITTED",
                "endTime": 1548717462.484,
                "phaseStatus": "SUCCEEDED"
            },
            {
                "durationInSeconds": 0,
                "startTime": 1548717462.484,
                "phaseType": "QUEUED",
                "endTime": 1548717462.775,
                "phaseStatus": "SUCCEEDED"
            },
            {
                "durationInSeconds": 34,
                "endTime": 1548717496.909,
                "contexts": [
                    {
                        "statusCode": "",
                        "message": ""
                    }
                ],
                "startTime": 1548717462.775,
                "phaseType": "PROVISIONING",
                "phaseStatus": "SUCCEEDED"
            },
            {
                "durationInSeconds": 15,
                "endTime": 1548717512.555,
```

```
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548717496.909,
                    "phaseType": "DOWNLOAD_SOURCE",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 0,
                    "endTime": 1548717512.734,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548717512.555,
                    "phaseType": "INSTALL",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 0,
                    "endTime": 1548717512.924,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548717512.734,
                    "phaseType": "PRE_BUILD",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 9,
                    "endTime": 1548717522.254,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
```

```
            ],
            "startTime": 1548717512.924,
            "phaseType": "BUILD",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 3,
            "endTime": 1548717525.498,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548717522.254,
            "phaseType": "POST_BUILD",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 9,
            "endTime": 1548717534.646,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548717525.498,
            "phaseType": "UPLOAD_ARTIFACTS",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 2,
            "endTime": 1548717536.846,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548717534.646,
            "phaseType": "FINALIZING",
            "phaseStatus": "SUCCEEDED"
        },
```

```
                {
                    "startTime": 1548717536.846,
                    "phaseType": "COMPLETED"
                }
            ],
            "startTime": 1548717462.122,
            "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
            "initiator": "codepipeline/CodeBuild-Pipeline",
            "secondarySources": [],
            "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-
codebuild-service-role",
            "currentPhase": "COMPLETED",
            "id": "codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
            "cache": {
                "type": "NO_CACHE"
            },
            "sourceVersion": "arn:aws:s3:::codepipeline-us-west-2-820783811474/
CodeBuild-Python-Pip/SourceArti/1TspnN3.zip",
            "endTime": 1548717536.846,
            "arn": "arn:aws:codebuild:us-west-2:123456789012:build/codebuild-demo-
project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
            "queuedTimeoutInMinutes": 480,
            "resolvedSourceVersion": "f2194c1757bbdcb0f8f229254a4b3c8b27d43e0b"
        },
        {
            "artifacts": {
                "md5sum": "",
                "overrideArtifactName": false,
                "location": "arn:aws:s3:::my-artifacts/codebuild-demo-project",
                "encryptionDisabled": false,
                "sha256sum": ""
            },
            "logs": {
                "cloudWatchLogs": {
                    "status": "ENABLED"
                },
                "s3Logs": {
                    "status": "DISABLED"
                },
                "streamName": "4dea3ca4-20ec-4898-b22a-a9eb9292775d",
                "groupName": "/aws/codebuild/codebuild-demo-project",
                "deepLink": "https://console.aws.amazon.com/cloudwatch/
home?region=us-west-2#logEvent:group=/aws/codebuild/codebuild-demo-
project;stream=4dea3ca4-20ec-4898-b22a-a9eb9292775d"
```

```
            },
            "timeoutInMinutes": 60,
            "environment": {
                "privilegedMode": false,
                "computeType": "BUILD_GENERAL1_MEDIUM",
                "image": "aws/codebuild/windows-base:1.0",
                "environmentVariables": [],
                "type": "WINDOWS_CONTAINER"
            },
            "projectName": "codebuild-demo-project",
            "buildComplete": true,
            "source": {
                "gitCloneDepth": 1,
                "location": "https://github.com/my-repo/codebuild-demo-project.git",
                "insecureSsl": false,
                "reportBuildStatus": false,
                "type": "GITHUB"
            },
            "buildStatus": "SUCCEEDED",
            "secondaryArtifacts": [],
            "phases": [
                {
                    "durationInSeconds": 0,
                    "startTime": 1548716241.89,
                    "phaseType": "SUBMITTED",
                    "endTime": 1548716242.241,
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 0,
                    "startTime": 1548716242.241,
                    "phaseType": "QUEUED",
                    "endTime": 1548716242.536,
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 33,
                    "endTime": 1548716276.171,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
```

```
            "startTime": 1548716242.536,
            "phaseType": "PROVISIONING",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 15,
            "endTime": 1548716291.809,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548716276.171,
            "phaseType": "DOWNLOAD_SOURCE",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 0,
            "endTime": 1548716291.993,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548716291.809,
            "phaseType": "INSTALL",
            "phaseStatus": "SUCCEEDED"
        },
        {
            "durationInSeconds": 0,
            "endTime": 1548716292.191,
            "contexts": [
                {
                    "statusCode": "",
                    "message": ""
                }
            ],
            "startTime": 1548716291.993,
            "phaseType": "PRE_BUILD",
            "phaseStatus": "SUCCEEDED"
        },
        {
```

```
                    "durationInSeconds": 9,
                    "endTime": 1548716301.622,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548716292.191,
                    "phaseType": "BUILD",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 3,
                    "endTime": 1548716304.783,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548716301.622,
                    "phaseType": "POST_BUILD",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 8,
                    "endTime": 1548716313.775,
                    "contexts": [
                        {
                            "statusCode": "",
                            "message": ""
                        }
                    ],
                    "startTime": 1548716304.783,
                    "phaseType": "UPLOAD_ARTIFACTS",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "durationInSeconds": 2,
                    "endTime": 1548716315.935,
                    "contexts": [
                        {
                            "statusCode": "",
```

```
                            "message": ""
                        }
                    ],
                    "startTime": 1548716313.775,
                    "phaseType": "FINALIZING",
                    "phaseStatus": "SUCCEEDED"
                },
                {
                    "startTime": 1548716315.935,
                    "phaseType": "COMPLETED"
                }
            ],
            "startTime": 1548716241.89,
            "secondarySourceVersions": [],
            "initiator": "my-codebuild-project",
            "arn": "arn:aws:codebuild:us-west-2:123456789012:build/codebuild-demo-
project:815e755f-bade-4a7e-80f0-efe51EXAMPLE",
            "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
            "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-
codebuild-service-role",
            "currentPhase": "COMPLETED",
            "id": "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE",
            "cache": {
                "type": "NO_CACHE"
            },
            "endTime": 1548716315.935,
            "secondarySources": [],
            "queuedTimeoutInMinutes": 480,
            "resolvedSourceVersion": "f2194c1757bbdcb0f8f229254a4b3c8b27d43e0b"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的查看构建的详细信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetBuilds。

batch-get-projects

以下代码示例演示了如何使用 batch-get-projects。

Amazon CLI

获取 Amazon CodeBuild 构建项目名称的列表。

以下 batch-get-projects 示例按名称指定获取 CodeBuild 构建项目的列表。

```
aws codebuild batch-get-projects --names codebuild-demo-project codebuild-demo-project2 my-other-demo-project
```

在以下输出中，projectsNotFound 数组列出了已指定但未找到的所有构建项目名称。projects 数组列出了可找到相关信息的所有构建项目的详细信息。

```
{
    "projectsNotFound": [],
    "projects": [
        {
            "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
            "name": "codebuild-demo-project2",
            "queuedTimeoutInMinutes": 480,
            "timeoutInMinutes": 60,
            "source": {
                "buildspec": "version: 0.2\n\n#env:\n  #variables:\n     # key: \"value\"\n     # key: \"value\"\n  #parameter-store:\n     # key: \"value\"\n  # key:\"value\"\n\nphases:\n  #install:\n    #commands:\n      # - command\n  # - command\n  #pre_build:\n    #commands:\n      # - command\n      # - command\n  build:\n    commands:\n      # - command\n      # - command\n  #post_build:\n    #commands:\n      # - command\n      # - command\n#artifacts:\n  #files:\n    # - location\n    # - location\n  #name: $(date +%Y-%m-%d)\n  #discard-paths: yes\n  #base-directory: location\n#cache:\n  #paths:\n    # - paths",
                "type": "NO_SOURCE",
                "insecureSsl": false,
                "gitCloneDepth": 1
            },
            "artifacts": {
                "type": "NO_ARTIFACTS"
            },
            "badge": {
                "badgeEnabled": false
            },
            "lastModified": 1540588091.108,
            "created": 1540588091.108,
            "arn": "arn:aws:codebuild:us-west-2:123456789012:project/test-for-sample",
```

```
            "secondarySources": [],
            "secondaryArtifacts": [],
            "cache": {
                "type": "NO_CACHE"
            },
            "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-test-
role",
            "environment": {
                "image": "aws/codebuild/java:openjdk-8",
                "privilegedMode": true,
                "type": "LINUX_CONTAINER",
                "computeType": "BUILD_GENERAL1_SMALL",
                "environmentVariables": []
            },
            "tags": []
        },
        {
            "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
            "name": "my-other-demo-project",
            "queuedTimeoutInMinutes": 480,
            "timeoutInMinutes": 60,
            "source": {
                "location": "https://github.com/iversonic/codedeploy-sample.git",
                "reportBuildStatus": false,
                "buildspec": "buildspec.yml",
                "insecureSsl": false,
                "gitCloneDepth": 1,
                "type": "GITHUB",
                "auth": {
                    "type": "OAUTH"
                }
            },
            "artifacts": {
                "type": "NO_ARTIFACTS"
            },
            "badge": {
                "badgeEnabled": false
            },
            "lastModified": 1523401711.73,
            "created": 1523401711.73,
            "arn": "arn:aws:codebuild:us-west-2:123456789012:project/Project2",
            "cache": {
                "type": "NO_CACHE"
            },
```

```
            "serviceRole": "arn:aws:iam::123456789012:role/service-role/codebuild-
Project2-service-role",
            "environment": {
                "image": "aws/codebuild/nodejs:4.4.7",
                "privilegedMode": false,
                "type": "LINUX_CONTAINER",
                "computeType": "BUILD_GENERAL1_SMALL",
                "environmentVariables": []
            },
            "tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的查看构建项目的详细信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetProjects。

batch-get-report-groups

以下代码示例演示了如何使用 batch-get-report-groups。

Amazon CLI

获取 Amazon CodeBuild 中一个或多个报告组的信息。

以下 batch-get-report-groups 示例检索有关带指定 ARN 的报告组的信息。

```
aws codebuild batch-get-report-groups \
    --report-group-arns arn:aws:codebuild:<region-ID>:<user-ID>:report-group/
<report-group-name>
```

输出：

```
{
    "reportGroups": [
        {
            "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-
group-name>",
            "name": "report-group-name",
            "type": "TEST",
            "exportConfig": {
```

```
                "exportConfigType": "NO_EXPORT"
            },
            "created": "2020-10-01T18:04:08.466000+00:00",
            "lastModified": "2020-10-01T18:04:08.466000+00:00",
            "tags": []
        }
    ],
    "reportGroupsNotFound": []
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetReportGroups。

batch-get-reports

以下代码示例演示了如何使用 batch-get-reports。

Amazon CLI

获取 Amazon CodeBuild 中一个或多个报告的信息。

以下 batch-get-reports 示例检索有关带指定 ARN 的报告的信息。

```
aws codebuild batch-get-reports \
    --report-arns arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report 1 ID> arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report 2 ID>
```

输出：

```
{
    "reports": [
        {
            "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report 1 ID>",
            "type": "TEST",
            "name": "<report-group-name>",
            "reportGroupArn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-group-name>",
            "executionId": "arn:aws:codebuild:<region-ID>:<user-ID>:build/test-reports:<ID>",
            "status": "FAILED",
```

```
            "created": "2020-10-01T11:25:22.531000-07:00",
            "expired": "2020-10-31T11:25:22-07:00",
            "exportConfig": {
                "exportConfigType": "NO_EXPORT"
            },
            "truncated": false,
            "testSummary": {
                "total": 28,
                "statusCounts": {
                    "ERROR": 5,
                    "FAILED": 1,
                    "SKIPPED": 4,
                    "SUCCEEDED": 18,
                    "UNKNOWN": 0
                },
                "durationInNanoSeconds": 94000000
            }
        },
        {
            "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-
name>:<report 2 ID>",
            "type": "TEST",
            "name": "<report-group-name>",
            "reportGroupArn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/
<report-group-name>",
            "executionId": "arn:aws:codebuild:<region-ID>:<user-ID>:build/test-
reports:<ID>",
            "status": "FAILED",
            "created": "2020-10-01T11:13:05.816000-07:00",
            "expired": "2020-10-31T11:13:05-07:00",
            "exportConfig": {
                "exportConfigType": "NO_EXPORT"
            },
            "truncated": false,
            "testSummary": {
                "total": 28,
                "statusCounts": {
                    "ERROR": 5,
                    "FAILED": 1,
                    "SKIPPED": 4,
                    "SUCCEEDED": 18,
                    "UNKNOWN": 0
                },
                "durationInNanoSeconds": 94000000
```

```
            }
        }
    ],
    "reportsNotFound": []
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetReports。

create-project

以下代码示例演示了如何使用 create-project。

Amazon CLI

示例 1：创建 Amazon CodeBuild 构建项目

以下 create-project 示例使用 S3 存储桶中的源文件创建 CodeBuild 构建项目

```
aws codebuild create-project \
    --name "my-demo-project" \
    --source "{\"type\": \"S3\",\"location\": \"codebuild-us-west-2-123456789012-
input-bucket/my-source.zip\"}" \
    --artifacts {"\"type\": \"S3\",\"location\": \"codebuild-us-west-2-123456789012-
output-bucket\""} \
    --environment "{\"type\": \"LINUX_CONTAINER\",\"image\": \"aws/codebuild/
standard:1.0\",\"computeType\": \"BUILD_GENERAL1_SMALL\"}" \
    --service-role "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role"
```

输出：

```
{
    "project": {
        "arn": "arn:aws:codebuild:us-west-2:123456789012:project/my-demo-project",
        "name": "my-cli-demo-project",
        "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
        "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
        "lastModified": 1556839783.274,
        "badge": {
            "badgeEnabled": false
```

```
        },
        "queuedTimeoutInMinutes": 480,
        "environment": {
            "image": "aws/codebuild/standard:1.0",
            "computeType": "BUILD_GENERAL1_SMALL",
            "type": "LINUX_CONTAINER",
            "imagePullCredentialsType": "CODEBUILD",
            "privilegedMode": false,
            "environmentVariables": []
        },
        "artifacts": {
            "location": "codebuild-us-west-2-123456789012-output-bucket",
            "name": "my-cli-demo-project",
            "namespaceType": "NONE",
            "type": "S3",
            "packaging": "NONE",
            "encryptionDisabled": false
        },
        "source": {
            "type": "S3",
            "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source.zip",
            "insecureSsl": false
        },
        "timeoutInMinutes": 60,
        "cache": {
            "type": "NO_CACHE"
        },
        "created": 1556839783.274
    }
}
```

示例 2：使用 JSON 输入文件作为参数创建 Amazon CodeBuild 构建项目

以下 `create-project` 示例通过在 JSON 输入文件中传递所有必需的参数来创建 CodeBuild 构建项目。通过仅使用 `--generate-cli-skeleton parameter` 运行命令来创建输入文件模板。

```
aws codebuild create-project --cli-input-json file://create-project.json
```

输入 JSON 文件 `create-project.json` 包含以下内容：

```
{
    "name": "codebuild-demo-project",
```

```
    "source": {
        "type": "S3",
        "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
    },
    "artifacts": {
        "type": "S3",
        "location": "codebuild-region-ID-account-ID-output-bucket"
    },
    "environment": {
        "type": "LINUX_CONTAINER",
        "image": "aws/codebuild/standard:1.0",
        "computeType": "BUILD_GENERAL1_SMALL"
    },
    "serviceRole": "serviceIAMRole"
}
```

输出：

```
{
    "project": {
        "name": "codebuild-demo-project",
        "serviceRole": "serviceIAMRole",
        "tags": [],
        "artifacts": {
            "packaging": "NONE",
            "type": "S3",
            "location": "codebuild-region-ID-account-ID-output-bucket",
            "name": "message-util.zip"
        },
        "lastModified": 1472661575.244,
        "timeoutInMinutes": 60,
        "created": 1472661575.244,
        "environment": {
            "computeType": "BUILD_GENERAL1_SMALL",
            "image": "aws/codebuild/standard:1.0",
            "type": "LINUX_CONTAINER",
            "environmentVariables": []
        },
        "source": {
            "type": "S3",
            "location": "codebuild-region-ID-account-ID-input-bucket/
MessageUtil.zip"
        },
```

```
        "encryptionKey": "arn:aws:kms:region-ID:account-ID:alias/aws/s3",
        "arn": "arn:aws:codebuild:region-ID:account-ID:project/codebuild-demo-
project"
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的创建构建项目（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProject。

create-report-group

以下代码示例演示了如何使用 create-report-group。

Amazon CLI

在 Amazon CodeBuild 中创建报告组。

以下 create-report-group 示例创建一个新的报告组。

```
aws codebuild create-report-group \
    --cli-input-json file://create-report-group-source.json
```

create-report-group-source.json 的内容：

```
{
    "name": "cli-created-report-group",
    "type": "TEST",
    "exportConfig": {
        "exportConfigType": "S3",
        "s3Destination": {
            "bucket": "amzn-s3-demo-bucket",
            "path": "",
            "packaging": "ZIP",
            "encryptionDisabled": true
        }
    }
}
```

输出：

```
{
```

```
    "reportGroup": {
        "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-
report-group",
        "name": "cli-created-report-group",
        "type": "TEST",
        "exportConfig": {
            "exportConfigType": "S3",
            "s3Destination": {
                "bucket": "amzn-s3-demo-bucket",
                "path": "",
                "packaging": "ZIP",
                "encryptionDisabled": true
            }
        },
        "created": 1602020026.775,
        "lastModified": 1602020026.775
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReportGroup。

`create-webhook`

以下代码示例演示了如何使用 `create-webhook`。

Amazon CLI

为 Amazon CodeBuild 项目创建 webhook 筛选条件

以下 `create-webhook` 示例为名为 `my-project` 的 CodeBuild 项目创建带有两个筛选条件组的 webhook。第一个筛选条件组使用与正则表达式 `^refs/heads/master$` 匹配的 Git 引用名称以及与 `^refs/heads/myBranch$` 匹配的头部引用，指定在分支上创建、更新或重新打开的拉取请求。第二个筛选条件组使用与正则表达式 `^refs/heads/myBranch$` 不匹配的 Git 引用名称，指定分支上的推送请求。

```
aws codebuild create-webhook \
    --project-name my-project \
    --filter-groups "[[{\"type\":\"EVENT\",\"pattern\":\"PULL_REQUEST_CREATED,
 PULL_REQUEST_UPDATED, PULL_REQUEST_REOPENED\"},{\"type\":\"HEAD_REF\",\"pattern
\":\"^refs/heads/myBranch$\",\"excludeMatchedPattern\":true},{\"type\":\"BASE_REF
```

```
\",\"pattern\":\"^refs/heads/master$\",\"excludeMatchedPattern\":true}],[{\"type\":
\"EVENT\",\"pattern\":\"PUSH\"},{\"type\":\"HEAD_REF\",\"pattern\":\"^refs/heads/
myBranch$\",\"excludeMatchedPattern\":true}]]"
```

输出：

```
{
    "webhook": {
        "payloadUrl": "https://codebuild.us-west-2.amazonaws.com/webhooks?
t=eyJlbmNyeXB0ZWREYXRhIjoiVVl5MGtoeGRwSzZFRXl2Wnh4bld1Z0tKZ291TVpQNEtFamQ3RDlDYWpRaGIreVFrdm
        "url": "https://api.github.com/repos/iversonic/codedeploy-sample/
hooks/105190656",
        "lastModifiedSecret": 1556311319.069,
        "filterGroups": [
            [
                {
                    "type": "EVENT",
                    "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED",
                    "excludeMatchedPattern": false
                },
                {
                    "type": "HEAD_REF",
                    "pattern": "refs/heads/myBranch$",
                    "excludeMatchedPattern": true
                },
                {
                    "type": "BASE_REF",
                    "pattern": "refs/heads/master$",
                    "excludeMatchedPattern": true
                }
            ],
            [
                {
                    "type": "EVENT",
                    "pattern": "PUSH",
                    "excludeMatchedPattern": false
                },
                {
                    "type": "HEAD_REF",
                    "pattern": "refs/heads/myBranch$",
                    "excludeMatchedPattern": true
                }
```

```
            ]
        ]
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的 Filter GitHub Webhook Events（SDK）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWebhook。

delete-build-batch

以下代码示例演示了如何使用 delete-build-batch。

Amazon CLI

在 Amazon CodeBuild 中删除批量构建。

以下 delete-build-batch 示例删除指定批量构建。

```
aws codebuild delete-build-batch \
    --id <project-name>:<batch-ID>
```

输出：

```
{
    "statusCode": "BATCH_DELETED",
    "buildsDeleted": [
        "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-
ID>",
        "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-
ID>",
        "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-
ID>",
        "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-ID>"
    ],
    "buildsNotDeleted": []
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBuildBatch。

delete-project

以下代码示例演示了如何使用 delete-project。

Amazon CLI

删除 Amazon CodeBuild 构建项目

以下 delete-project 示例删除指定的 CodeBuild 构建项目。

```
aws codebuild delete-project --name my-project
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的删除构建项目（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProject。

delete-report-group

以下代码示例演示了如何使用 delete-report-group。

Amazon CLI

在 Amazon CodeBuild 中删除报告组。

以下 delete-report-group 示例删除具有指定 ARN 的报告组。

```
aws codebuild delete-report-group \
    --arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-group-name>
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReportGroup。

delete-report

以下代码示例演示了如何使用 delete-report。

Amazon CLI

在 Amazon CodeBuild 中删除报告。

以下 delete-report 示例删除指定报告。

```
aws codebuild delete-report \
    --arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-name>:<report-ID>
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReport。

delete-source-credentials

以下代码示例演示了如何使用 delete-source-credentials。

Amazon CLI

断开与源提供商的连接并删除其访问令牌。

以下 delete-source-credentials 示例断开与源提供商的连接并删除其令牌。用于连接源提供商的源凭证 ARN 决定了使用哪些源凭证。

```
aws codebuild delete-source-credentials --arn arn-of-your-credentials
```

输出：

```
{
    "arn": "arn:aws:codebuild:your-region:your-account-id:token/your-server-type"
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用访问令牌连接源提供商（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSourceCredentials。

delete-webhook

以下代码示例演示了如何使用 delete-webhook。

Amazon CLI

从 Amazon CodeBuild 项目中删除 webhook 筛选条件

以下 delete-webhook 示例删除指定 CodeBuild 项目的 webhook。

```
aws codebuild delete-webhook --project-name my-project
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的停止自动运行构建（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWebhook。

describe-code-coverages

以下代码示例演示了如何使用 describe-code-coverages。

Amazon CLI

获取有关 Amazon CodeBuild 中代码覆盖率测试结果的详细信息。

以下 describe-code-coverages 示例在指定报告中获取有关代码覆盖率测试结果的信息。

```
aws codebuild describe-code-coverages \
    --report-arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-name>:<report-ID>
```

输出：

```
{
    "codeCoverages": [
        {
            "id": "20a0adcc-db13-4b66-804b-ecaf9f852855",
            "reportARN": "arn:aws:codebuild:<region-ID>:972506530580:report/<report-group-name>:<report-ID>",
            "filePath": "<source-file-1-path>",
            "lineCoveragePercentage": 83.33,
            "linesCovered": 5,
            "linesMissed": 1,
            "branchCoveragePercentage": 50.0,
            "branchesCovered": 1,
            "branchesMissed": 1,
            "expired": "2020-11-20T21:22:45+00:00"
        },
        {
            "id": "0887162d-bf57-4cf1-a164-e432373d1a83",
```

```
            "reportARN": "arn:aws:codebuild:<region-ID>:972506530580:report/<report-
group-name>:<report-ID>",
            "filePath": "<source-file-2-path>",
            "lineCoveragePercentage": 90.9,
            "linesCovered": 10,
            "linesMissed": 1,
            "branchCoveragePercentage": 50.0,
            "branchesCovered": 1,
            "branchesMissed": 1,
            "expired": "2020-11-20T21:22:45+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的代码覆盖率报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCodeCoverages。

describe-test-cases

以下代码示例演示了如何使用 describe-test-cases。

Amazon CLI

获取有关 Amazon CodeBuild 中测试用例的详细信息。

以下 describe-test-cases 示例在指定报告中获取有关测试用例的信息。

```
aws codebuild describe-test-cases \
    --report-arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-
name>:<report-ID>
```

输出：

```
{
    "testCases": [
        {
            "reportArn": "arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-
group-name>:<report-ID>",
            "testRawDataPath": "<test-report-path>",
            "prefix": "NUnit.Tests.Assemblies.MockTestFixture",
            "name": "NUnit.Tests.Assemblies.MockTestFixture.NotRunnableTest",
            "status": "ERROR",
```

```
            "durationInNanoSeconds": 0,
            "message": "No arguments were provided\n",
            "expired": "2020-11-20T17:52:10+00:00"
        },
        {
            "reportArn": "arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-
group-name>:<report-ID>",
            "testRawDataPath": "<test-report-path>",
            "prefix": "NUnit.Tests.Assemblies.MockTestFixture",
            "name": "NUnit.Tests.Assemblies.MockTestFixture.TestWithException",
            "status": "ERROR",
            "durationInNanoSeconds": 0,
            "message": "System.ApplicationException : Intentional Exception
\nat NUnit.Tests.Assemblies.MockTestFixture.MethodThrowsException()\nat
 NUnit.Tests.Assemblies.MockTestFixture.TestWithException()\n\n",
            "expired": "2020-11-20T17:52:10+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中使用测试报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTestCases。

`import-source-credentials`

以下代码示例演示了如何使用 `import-source-credentials`。

Amazon CLI

通过导入源提供商的凭证，将 Amazon CodeBuild 用户与源提供商联系起来。

以下 `import-source-credentials` 示例为使用 BASIC_AUTH 作为其身份验证类型的 Bitbucket 存储库导入令牌。

```
aws codebuild import-source-credentials --server-type BITBUCKET --auth-
type BASIC_AUTH --token my-Bitbucket-password --username my-Bitbucket-username
```

输出：

```
{
```

```
    "arn": "arn:aws:codebuild:us-west-2:123456789012:token/bitbucket"
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用访问令牌连接源提供商（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportSourceCredentials。

`invalidate-project-cache`

以下代码示例演示了如何使用 `invalidate-project-cache`。

Amazon CLI

重置 Amazon CodeBuild 构建项目的缓存。

以下 `invalidate-project-cache` 示例为指定的 CodeBuild 项目重置缓存。

```
aws codebuild invalidate-project-cache --project-name my-project
```

此命令不生成任何输出。

有关详细信息，请参阅《Amazon CodeBuild 用户指南》中的在 CodeBuild 中构建缓存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InvalidateProjectCache。

`list-build-batches-for-project`

以下代码示例演示了如何使用 `list-build-batches-for-project`。

Amazon CLI

在 Amazon CodeBuild 中列出特定构建项目的批量构建。

以下 `list-build-batches-for-project` 示例列出指定项目的 CodeBuild 批量构建。

```
aws codebuild list-build-batches-for-project \
    --project-name "<project-name>"
```

输出：

```
{
    "ids": [
```

```
        "<project-name>:<batch-ID>",
        "<project-name>:<batch-ID>"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuildBatchesForProject。

`list-build-batches`

以下代码示例演示了如何使用 `list-build-batches`。

Amazon CLI

在 Amazon CodeBuild 中列出批量构建。

以下 `list-build-batches` 示例列出当前账户的 CodeBuild 批量构建。

```
aws codebuild list-build-batches
```

输出：

```
{
    "ids": [
        "<project-name>:<batch-ID>",
        "<project-name>:<batch-ID>"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的“在 Amazon CodeBuild 中批量构建 <https://docs.aws.amazon.com/codebuild/latest/userguide/batch-build.html>)__”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuildBatches。

`list-builds-for-project`

以下代码示例演示了如何使用 `list-builds-for-project`。

Amazon CLI

查看 Amazon CodeBuild 构建项目的构建列表。

以下 `list-builds-for-project` 示例按降序列出指定 CodeBuild 构建项目的构建 ID。

```
aws codebuild list-builds-for-project --project-name codebuild-demo-project --sort-order DESCENDING
```

输出：

```
{
    "ids": [
        "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-11111example",
        "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-22222example",
        "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-33333example",
        "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-44444example",
        "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-55555example"
    ]
}
```

有关详细信息，请参阅《Amazon CodeBuild 用户指南》中的查看构建项目的构建 ID 列表（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuildsForProject。

`list-builds`

以下代码示例演示了如何使用 `list-builds`。

Amazon CLI

获取 Amazon CodeBuild 构建 ID 的列表。

以下 `list-builds` 示例获取 CodeBuild ID 的列表，按升序排列。

```
aws codebuild list-builds --sort-order ASCENDING
```

输出包括一个 `nextToken` 值，该值表示还有更多可用的输出。

```
{
    "nextToken": "4AEA6u7J...The full token has been omitted for
 brevity...MzY2OA==",
    "ids": [
        "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE"
        "codebuild-demo-project:84a7f3d1-d40e-4956-b4cf-7a9d4EXAMPLE"
```

```
            ... The full list of build IDs has been omitted for brevity ...
        "codebuild-demo-project:931d0b72-bf6f-4040-a472-5c707EXAMPLE"
    ]
}
```

再次运行此命令并提供上一个响应中的 nextToken 值作为参数，从而获取输出的下一部分。重复此操作，直到在响应中不再收到 nextToken 值。

```
aws codebuild list-builds --sort-order ASCENDING --next-
token 4AEA6u7J...The full token has been omitted for brevity...MzY2OA==
```

输出的下一部分：

```
{
    "ids": [
        "codebuild-demo-project:49015049-21cf-4b50-9708-df115EXAMPLE",
        "codebuild-demo-project:543e7206-68a3-46d6-a4da-759abEXAMPLE",
            ... The full list of build IDs has been omitted for brevity ...
        "codebuild-demo-project:c282f198-4582-4b38-bdc0-26f96EXAMPLE"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的查看构建 ID 的列表（Amazon CLI）

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuilds。

list-curated-environment-images

以下代码示例演示了如何使用 list-curated-environment-images。

Amazon CLI

获取由 Amazon CodeBuild 管理的可用于构建的 Docker 映像的列表。

以下 list-curated-environment-images 示例列出由 CodeBuild 管理的可用于构建的 Docker 映像：

```
aws codebuild list-curated-environment-images
```

输出：

```
{
```

```
    "platforms": [
        {
            "platform": "AMAZON_LINUX",
            "languages": [
                {
                    "language": "JAVA",
                    "images": [
                        {
                            "description": "AWS ElasticBeanstalk - Java 7 Running on
 Amazon Linux 64bit v2.1.3",
                            "name": "aws/codebuild/eb-java-7-amazonlinux-64:2.1.3",
                            "versions": [
                                "aws/codebuild/eb-java-7-amazonlinux-64:2.1.3-1.0.0"
                            ]
                        },
                        {
                            "description": "AWS ElasticBeanstalk - Java 8 Running on
 Amazon Linux 64bit v2.1.3",
                            "name": "aws/codebuild/eb-java-8-amazonlinux-64:2.1.3",
                            "versions": [
                                "aws/codebuild/eb-java-8-amazonlinux-64:2.1.3-1.0.0"
                            ]
                        },
                        ... LIST TRUNCATED FOR BREVITY ...
                    ]
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的由 CodeBuild 提供的 Docker 映像。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCuratedEnvironmentImages。

`list-projects`

以下代码示例演示了如何使用 `list-projects`。

Amazon CLI

获取 Amazon CodeBuild 构建项目名称的列表。

以下 `list-projects` 示例获取 CodeBuild 构建项目的列表，按名称升序排列。

```
aws codebuild list-projects --sort-by NAME --sort-order ASCENDING
```

输出包括一个 nextToken 值，该值表示还有更多可用的输出。

```
{
    "nextToken": "Ci33ACF6...The full token has been omitted for brevity...U
+AkMx8=",
    "projects": [
        "codebuild-demo-project",
        "codebuild-demo-project2",
            ... The full list of build project names has been omitted for
 brevity ...
        "codebuild-demo-project99"
    ]
}
```

再次运行此命令并提供来自上一个响应的 nextToken 值作为参数，从而获取输出的下一部分。重复此操作，直到在响应中不再收到 nextToken 值。

```
aws codebuild list-projects  --sort-by NAME --sort-order ASCENDING --next-
token Ci33ACF6...The full token has been omitted for brevity...U+AkMx8=

{
    "projects": [
        "codebuild-demo-project100",
        "codebuild-demo-project101",

  ... The full list of build project names has been omitted for brevity ...
        "codebuild-demo-project122"

  ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的查看构建项目名称的列表（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProjects。

list-report-groups

以下代码示例演示了如何使用 list-report-groups。

Amazon CLI

获取 Amazon CodeBuild 中报告组 ARN 的列表。

以下 `list-report-groups` 示例检索区域内账户的报告组 ARN。

```
aws codebuild list-report-groups
```

输出：

```
{
    "reportGroups": [
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-1",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-2",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-3"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReportGroups。

`list-reports-for-report-group`

以下代码示例演示了如何使用 `list-reports-for-report-group`。

Amazon CLI

获取 Amazon CodeBuild 中报告组中报告的列表。

以下 `list-report-for-report-groups` 示例为区域内的账户检索指定报告组中的报告。

```
aws codebuild list-reports-for-report-group \
    --report-group-arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-group-name>
```

输出：

```
{
    "reports": [
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-1",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-2",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-3"
```

```
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReportsForReportGroup。

`list-reports`

以下代码示例演示了如何使用 `list-reports`。

Amazon CLI

获取 Amazon CodeBuild 中当前账户的报告列表。

以下 `list-reports` 示例检索当前账户报告的 ARN。

```
aws codebuild list-reports
```

输出：

```
{
    "reports": [
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report
 ID>",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report
 ID>",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report
 ID>"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReports。

`list-shared-projects`

以下代码示例演示了如何使用 `list-shared-projects`。

Amazon CLI

列出 Amazon CodeBuild 中的共享项目。

以下 list-shared-projects 示例列出可供当前账户使用的 CodeBuild 共享项目。

```
aws codebuild list-shared-projects
```

输出：

```
{
    "projects": [
        "arn:aws:codebuild:<region-ID>:<account-ID>:project/<shared-project-
name-1>",
        "arn:aws:codebuild:<region-ID>:<account-ID>:project/<shared-project-name-2>"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用共享的项目。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSharedProjects。

list-shared-report-groups

以下代码示例演示了如何使用 list-shared-report-groups。

Amazon CLI

获取 Amazon CodeBuild 中共享报告组 ARN 的列表。

以下 list-shared-report-groups 示例检索区域内账户的报告组 ARN。

```
aws codebuild list-shared-report-groups
```

输出：

```
{
    "reportGroups": [
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-1",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-2",
        "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-3"
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSharedReportGroups。

list-source-credentials

以下代码示例演示了如何使用 list-source-credentials。

Amazon CLI

查看 sourceCredentialsObjects 列表

以下 list-source-credentials 示例列出与一个 Bitbucket 账户和一个 GitHub 账户关联的 Amazon 账户的令牌。响应中的每个 sourceCredentialsInfos 对象都包含连接的源凭证信息。

```
aws codebuild list-source-credentials
```

输出：

```
{
    "sourceCredentialsInfos": [
        {
            "serverType": "BITBUCKET",
            "arn": "arn:aws:codebuild:us-west-2:123456789012:token/bitbucket",
            "authType": "BASIC_AUTH"
        },
        {
            "serverType": "GITHUB",
            "arn": "arn:aws:codebuild:us-west-2:123456789012:token/github",
            "authType": "OAUTH"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用访问令牌连接源提供商（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSourceCredentials。

retry-build-batch

以下代码示例演示了如何使用 retry-build-batch。

Amazon CLI

重试 Amazon CodeBuild 中失败的批量构建。

以下 retry-build-batch 示例重试指定的批量构建。

```
aws codebuild retry-build-batch \
    --id <project-name>:<batch-ID>
```

输出：

```
{
    "buildBatch": {
        "id": "<project-name>:<batch-ID>",
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-
name>:<batch-ID>",
        "startTime": "2020-10-21T17:26:23.099000+00:00",
        "currentPhase": "SUBMITTED",
        "buildBatchStatus": "IN_PROGRESS",
        "resolvedSourceVersion": "3a9e11cb419e8fff14b03883dc4e64f6155aaa7e",
        "projectName": "<project-name>",
        "phases": [
            {
                "phaseType": "SUBMITTED",
                "phaseStatus": "SUCCEEDED",
                "startTime": "2020-10-21T17:26:23.099000+00:00",
                "endTime": "2020-10-21T17:26:23.457000+00:00",
                "durationInSeconds": 0
            },
            {
                "phaseType": "DOWNLOAD_BATCHSPEC",
                "phaseStatus": "SUCCEEDED",
                "startTime": "2020-10-21T17:26:23.457000+00:00",
                "endTime": "2020-10-21T17:26:54.902000+00:00",
                "durationInSeconds": 31
            },
            {
                "phaseType": "IN_PROGRESS",
                "phaseStatus": "CLIENT_ERROR",
                "startTime": "2020-10-21T17:26:54.902000+00:00",
                "endTime": "2020-10-21T17:28:16.060000+00:00",
                "durationInSeconds": 81
            },
            {
                "phaseType": "FAILED",
                "phaseStatus": "RETRY",
                "startTime": "2020-10-21T17:28:16.060000+00:00",
```

```
                "endTime": "2020-10-21T17:29:39.709000+00:00",
                "durationInSeconds": 83
            },
            {
                "phaseType": "SUBMITTED",
                "startTime": "2020-10-21T17:29:39.709000+00:00"
            }
        ],
        "source": {
            "type": "GITHUB",
            "location": "https://github.com/strohm-a/<project-name>-graph.git",
            "gitCloneDepth": 1,
            "gitSubmodulesConfig": {
                "fetchSubmodules": false
            },
            "reportBuildStatus": false,
            "insecureSsl": false
        },
        "secondarySources": [],
        "secondarySourceVersions": [],
        "artifacts": {
            "location": ""
        },
        "secondaryArtifacts": [],
        "cache": {
            "type": "NO_CACHE"
        },
        "environment": {
            "type": "LINUX_CONTAINER",
            "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
            "computeType": "BUILD_GENERAL1_SMALL",
            "environmentVariables": [],
            "privilegedMode": false,
            "imagePullCredentialsType": "CODEBUILD"
        },
        "logConfig": {
            "cloudWatchLogs": {
                "status": "ENABLED"
            },
            "s3Logs": {
                "status": "DISABLED",
                "encryptionDisabled": false
            }
        },
```

```
        "buildTimeoutInMinutes": 60,
        "queuedTimeoutInMinutes": 480,
        "complete": false,
        "initiator": "<username>",
        "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
        "buildBatchNumber": 4,
        "buildBatchConfig": {
            "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<project-
name>",
            "restrictions": {
                "maximumBuildsAllowed": 100
            },
            "timeoutInMins": 480
        },
        "buildGroups": [
            {
                "identifier": "DOWNLOAD_SOURCE",
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T17:26:23.889000+00:00",
                    "buildStatus": "SUCCEEDED",
                    "primaryArtifact": {
                        "type": "no_artifacts",
                        "identifier": "DOWNLOAD_SOURCE"
                    },
                    "secondaryArtifacts": []
                }
            },
            {
                "identifier": "linux_small",
                "dependsOn": [],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T17:26:55.115000+00:00",
                    "buildStatus": "FAILED",
                    "primaryArtifact": {
                        "type": "no_artifacts",
                        "identifier": "linux_small"
                    },
                    "secondaryArtifacts": []
```

```
                }
            },
            {
                "identifier": "linux_medium",
                "dependsOn": [
                    "linux_small"
                ],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T17:26:54.594000+00:00",
                    "buildStatus": "STOPPED"
                }
            },
            {
                "identifier": "linux_large",
                "dependsOn": [
                    "linux_medium"
                ],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T17:26:54.701000+00:00",
                    "buildStatus": "STOPPED"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetryBuildBatch。

retry-build

以下代码示例演示了如何使用 retry-build。

Amazon CLI

重试 Amazon CodeBuild 中失败的构建。

以下 retry-build 示例重试指定的构建。

```
aws codebuild retry-build \
    --id <project-name>:<build-ID>
```

输出：

```
{
    "build": {
        "id": "<project-name>:<build-ID>",
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-
name>:<build-ID>",
        "buildNumber": 9,
        "startTime": "2020-10-21T17:51:38.161000+00:00",
        "currentPhase": "QUEUED",
        "buildStatus": "IN_PROGRESS",
        "projectName": "<project-name>",
        "phases": [
            {
                "phaseType": "SUBMITTED",
                "phaseStatus": "SUCCEEDED",
                "startTime": "2020-10-21T17:51:38.161000+00:00",
                "endTime": "2020-10-21T17:51:38.210000+00:00",
                "durationInSeconds": 0
            },
            {
                "phaseType": "QUEUED",
                "startTime": "2020-10-21T17:51:38.210000+00:00"
            }
        ],
        "source": {
            "type": "GITHUB",
            "location": "<GitHub-repo-URL>",
            "gitCloneDepth": 1,
            "gitSubmodulesConfig": {
                "fetchSubmodules": false
            },
            "reportBuildStatus": false,
            "insecureSsl": false
        },
        "secondarySources": [],
        "secondarySourceVersions": [],
        "artifacts": {
```

```
            "location": ""
        },
        "secondaryArtifacts": [],
        "cache": {
            "type": "NO_CACHE"
        },
        "environment": {
            "type": "LINUX_CONTAINER",
            "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
            "computeType": "BUILD_GENERAL1_SMALL",
            "environmentVariables": [],
            "privilegedMode": false,
            "imagePullCredentialsType": "CODEBUILD"
        },
        "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<service-role-
name>",
        "logs": {
            "deepLink": "https://console.aws.amazon.com/cloudwatch/home?
region=<region-ID>#logEvent:group=null;stream=null",
            "cloudWatchLogsArn": "arn:aws:logs:<region-ID>:<account-ID>:log-
group:null:log-stream:null",
            "cloudWatchLogs": {
                "status": "ENABLED"
            },
            "s3Logs": {
                "status": "DISABLED",
                "encryptionDisabled": false
            }
        },
        "timeoutInMinutes": 60,
        "queuedTimeoutInMinutes": 480,
        "buildComplete": false,
        "initiator": "<username>",
        "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3"
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetryBuild。

start-build-batch

以下代码示例演示了如何使用 start-build-batch。

Amazon CLI

启动 Amazon CodeBuild 中的批量构建。

以下 start-build-batch 示例启动指定项目的批量构建。

```
aws codebuild start-build-batch \
    --project-name <project-name>
```

输出：

```
{
    "buildBatch": {
        "id": "<project-name>:<batch-ID>",
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-
name>:<batch-ID>",
        "startTime": "2020-10-21T16:54:24.740000+00:00",
        "currentPhase": "SUBMITTED",
        "buildBatchStatus": "IN_PROGRESS",
        "projectName": "<project-name>",
        "source": {
            "type": "GITHUB",
            "location": "<GitHub-repo-URL>",
            "gitCloneDepth": 1,
            "gitSubmodulesConfig": {
                "fetchSubmodules": false
            },
            "reportBuildStatus": false,
            "insecureSsl": false
        },
        "secondarySources": [],
        "secondarySourceVersions": [],
        "artifacts": {
            "location": ""
        },
        "secondaryArtifacts": [],
        "cache": {
            "type": "NO_CACHE"
        },
        "environment": {
```

```
            "type": "LINUX_CONTAINER",
            "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
            "computeType": "BUILD_GENERAL1_SMALL",
            "environmentVariables": [],
            "privilegedMode": false,
            "imagePullCredentialsType": "CODEBUILD"
        },
        "logConfig": {
            "cloudWatchLogs": {
                "status": "ENABLED"
            },
            "s3Logs": {
                "status": "DISABLED",
                "encryptionDisabled": false
            }
        },
        "buildTimeoutInMinutes": 60,
        "queuedTimeoutInMinutes": 480,
        "complete": false,
        "initiator": "<username>",
        "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
        "buildBatchNumber": 3,
        "buildBatchConfig": {
            "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<service-
role-name>",
            "restrictions": {
                "maximumBuildsAllowed": 100
            },
            "timeoutInMins": 480
        }
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartBuildBatch。

start-build

以下代码示例演示了如何使用 start-build。

Amazon CLI

开始运行 Amazon CodeBuild 构建项目的构建。

以下 start-build 示例为指定的 CodeBuild 项目启动构建。构建会覆盖有关在超时前可在队列中等待的分钟数的项目设置，还会覆盖项目的构件设置。

```
aws codebuild start-build \
    --project-name "my-demo-project" \
    --queued-timeout-in-minutes-override 5 \
    --artifacts-override {"\"type\": \"S3\",\"location\": \"arn:aws:s3:::artifacts-
override\",\"overrideArtifactName\":true"}
```

输出：

```
{
    "build": {
        "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
        "buildStatus": "IN_PROGRESS",
        "buildComplete": false,
        "projectName": "my-demo-project",
        "timeoutInMinutes": 60,
        "source": {
            "insecureSsl": false,
            "type": "S3",
            "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source.zip"
        },
        "queuedTimeoutInMinutes": 5,
        "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
        "currentPhase": "QUEUED",
        "startTime": 1556905683.568,
        "environment": {
            "computeType": "BUILD_GENERAL1_MEDIUM",
            "environmentVariables": [],
            "type": "LINUX_CONTAINER",
            "privilegedMode": false,
            "image": "aws/codebuild/standard:1.0",
            "imagePullCredentialsType": "CODEBUILD"
        },
        "phases": [
            {
                "phaseStatus": "SUCCEEDED",
                "startTime": 1556905683.568,
                "phaseType": "SUBMITTED",
                "durationInSeconds": 0,
```

```
                "endTime": 1556905684.524
            },
            {
                "startTime": 1556905684.524,
                "phaseType": "QUEUED"
            }
        ],
        "logs": {
            "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=null;stream=null"
        },
        "artifacts": {
            "encryptionDisabled": false,
            "location": "arn:aws:s3:::artifacts-override/my-demo-project",
            "overrideArtifactName": true
        },
        "cache": {
            "type": "NO_CACHE"
        },
        "id": "my-demo-project::12345678-a1b2-c3d4-e5f6-11111EXAMPLE",
        "initiator": "my-aws-account-name",
        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/my-demo-
project::12345678-a1b2-c3d4-e5f6-11111EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的运行构建（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartBuild。

stop-build-batch

以下代码示例演示了如何使用 stop-build-batch。

Amazon CLI

停止 Amazon CodeBuild 中正在进行的批量构建。

以下 stop-build-batch 示例停止指定的批量构建。

```
aws codebuild stop-build-batch \
    --id <project-name>:<batch-ID>
```

输出：

```
{
    "buildBatch": {
        "id": "<project-name>:<batch-ID>",
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-
name>:<batch-ID>",
        "startTime": "2020-10-21T16:54:24.740000+00:00",
        "endTime": "2020-10-21T16:56:05.152000+00:00",
        "currentPhase": "STOPPED",
        "buildBatchStatus": "STOPPED",
        "resolvedSourceVersion": "aef7744ed069c51098e15c360f4102cd2cd1ad64",
        "projectName": "<project-name>",
        "phases": [
            {
                "phaseType": "SUBMITTED",
                "phaseStatus": "SUCCEEDED",
                "startTime": "2020-10-21T16:54:24.740000+00:00",
                "endTime": "2020-10-21T16:54:25.039000+00:00",
                "durationInSeconds": 0
            },
            {
                "phaseType": "DOWNLOAD_BATCHSPEC",
                "phaseStatus": "SUCCEEDED",
                "startTime": "2020-10-21T16:54:25.039000+00:00",
                "endTime": "2020-10-21T16:54:56.583000+00:00",
                "durationInSeconds": 31
            },
            {
                "phaseType": "IN_PROGRESS",
                "phaseStatus": "STOPPED",
                "startTime": "2020-10-21T16:54:56.583000+00:00",
                "endTime": "2020-10-21T16:56:05.152000+00:00",
                "durationInSeconds": 68
            },
            {
                "phaseType": "STOPPED",
                "startTime": "2020-10-21T16:56:05.152000+00:00"
            }
        ],
        "source": {
            "type": "GITHUB",
            "location": "<GitHub-repo-URL>",
            "gitCloneDepth": 1,
```

```
            "gitSubmodulesConfig": {
                "fetchSubmodules": false
            },
            "reportBuildStatus": false,
            "insecureSsl": false
        },
        "secondarySources": [],
        "secondarySourceVersions": [],
        "artifacts": {
            "location": ""
        },
        "secondaryArtifacts": [],
        "cache": {
            "type": "NO_CACHE"
        },
        "environment": {
            "type": "LINUX_CONTAINER",
            "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
            "computeType": "BUILD_GENERAL1_SMALL",
            "environmentVariables": [],
            "privilegedMode": false,
            "imagePullCredentialsType": "CODEBUILD"
        },
        "logConfig": {
            "cloudWatchLogs": {
                "status": "ENABLED"
            },
            "s3Logs": {
                "status": "DISABLED",
                "encryptionDisabled": false
            }
        },
        "buildTimeoutInMinutes": 60,
        "queuedTimeoutInMinutes": 480,
        "complete": true,
        "initiator": "Strohm",
        "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
        "buildBatchNumber": 3,
        "buildBatchConfig": {
            "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<project-
name>",
            "restrictions": {
                "maximumBuildsAllowed": 100
            },
```

```
            "timeoutInMins": 480
        },
        "buildGroups": [
            {
                "identifier": "DOWNLOAD_SOURCE",
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T16:54:25.468000+00:00",
                    "buildStatus": "SUCCEEDED",
                    "primaryArtifact": {
                        "type": "no_artifacts",
                        "identifier": "DOWNLOAD_SOURCE"
                    },
                    "secondaryArtifacts": []
                }
            },
            {
                "identifier": "linux_small",
                "dependsOn": [],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T16:54:56.833000+00:00",
                    "buildStatus": "IN_PROGRESS"
                }
            },
            {
                "identifier": "linux_medium",
                "dependsOn": [
                    "linux_small"
                ],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T16:54:56.211000+00:00",
                    "buildStatus": "PENDING"
                }
            },
            {
                "identifier": "linux_large",
```

```
                "dependsOn": [
                    "linux_medium"
                ],
                "ignoreFailure": false,
                "currentBuildSummary": {
                    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
                    "requestedOn": "2020-10-21T16:54:56.330000+00:00",
                    "buildStatus": "PENDING"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的在 Amazon CodeBuild 中批量构建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopBuildBatch。

stop-build

以下代码示例演示了如何使用 stop-build。

Amazon CLI

停止 Amazon CodeBuild 构建项目的构建。

以下 stop-build 示例停止指定的 CodeBuild 构建。

```
aws codebuild stop-build --id my-demo-project:12345678-a1b2-c3d4-e5f6-11111EXAMPLE
```

输出：

```
{
    "build": {
        "startTime": 1556906956.318,
        "initiator": "my-aws-account-name",
        "projectName": "my-demo-project",
        "currentPhase": "COMPLETED",
        "cache": {
            "type": "NO_CACHE"
        },
        "source": {
```

```
            "insecureSsl": false,
            "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source.zip",
            "type": "S3"
        },
        "id": "my-demo-project:1a2b3c4d-5678-90ab-cdef-11111EXAMPLE",
        "endTime": 1556906974.781,
        "phases": [
            {
                "durationInSeconds": 0,
                "phaseType": "SUBMITTED",
                "endTime": 1556906956.935,
                "phaseStatus": "SUCCEEDED",
                "startTime": 1556906956.318
            },
            {
                "durationInSeconds": 1,
                "phaseType": "QUEUED",
                "endTime": 1556906958.272,
                "phaseStatus": "SUCCEEDED",
                "startTime": 1556906956.935
            },
            {
                "phaseType": "PROVISIONING",
                "phaseStatus": "SUCCEEDED",
                "durationInSeconds": 14,
                "contexts": [
                    {
                        "message": "",
                        "statusCode": ""
                    }
                ],
                "endTime": 1556906972.847,
                "startTime": 1556906958.272
            },
            {
                "phaseType": "DOWNLOAD_SOURCE",
                "phaseStatus": "SUCCEEDED",
                "durationInSeconds": 0,
                "contexts": [
                        {
                        "message": "",
                        "statusCode": ""
                    }
```

```
                ],
                "endTime": 1556906973.552,
                "startTime": 1556906972.847
            },
            {
                "phaseType": "INSTALL",
                "phaseStatus": "SUCCEEDED",
                "durationInSeconds": 0,
                "contexts": [
                    {
                        "message": "",
                        "statusCode": ""
                    }
                ],
                "endTime": 1556906973.75,
                "startTime": 1556906973.552
            },
            {
                "phaseType": "PRE_BUILD",
                "phaseStatus": "SUCCEEDED",
                "durationInSeconds": 0,
                "contexts": [
                    {
                        "message": "",
                        "statusCode": ""
                    }
                ],
                "endTime": 1556906973.937,
                "startTime": 1556906973.75
            },
            {
                "durationInSeconds": 0,
                "phaseType": "BUILD",
                "endTime": 1556906974.781,
                "phaseStatus": "STOPPED",
                "startTime": 1556906973.937
            },
            {
                "phaseType": "COMPLETED",
                "startTime": 1556906974.781
            }
        ],
        "artifacts": {
            "location": "arn:aws:s3:::artifacts-override/my-demo-project",
```

```
            "encryptionDisabled": false,
            "overrideArtifactName": true
        },
        "buildComplete": true,
        "buildStatus": "STOPPED",
        "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
        "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
        "queuedTimeoutInMinutes": 5,
        "timeoutInMinutes": 60,
        "environment": {
            "type": "LINUX_CONTAINER",
            "environmentVariables": [],
            "computeType": "BUILD_GENERAL1_MEDIUM",
            "privilegedMode": false,
            "image": "aws/codebuild/standard:1.0",
            "imagePullCredentialsType": "CODEBUILD"
        },
        "logs": {
            "streamName": "1a2b3c4d-5678-90ab-cdef-11111EXAMPLE",
            "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=/aws/codebuild/my-demo-project;stream=1a2b3c4d-5678-90ab-
cdef-11111EXAMPLE",
            "groupName": "/aws/codebuild/my-demo-project"
        },
        "arn": "arn:aws:codebuild:us-west-2:123456789012:build/my-demo-
project:1a2b3c4d-5678-90ab-cdef-11111EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的停止构建（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopBuild。

update-project

以下代码示例演示了如何使用 update-project。

Amazon CLI

更改 Amazon CodeBuild 构建项目的设置。

以下 update-project 示例更改名为 my-demo-project 的指定 CodeBuild 构建项目的设置。

```
aws codebuild update-project --name "my-demo-project" \
    --description "This project is updated" \
    --source "{\"type\": \"S3\",\"location\": \"codebuild-us-west-2-123456789012-
input-bucket/my-source-2.zip\"}" \
    --artifacts {"\"type\": \"S3\",\"location\": \"codebuild-us-west-2-123456789012-
output-bucket-2\""} \
    --environment "{\"type\": \"LINUX_CONTAINER\",\"image\": \"aws/codebuild/
standard:1.0\",\"computeType\": \"BUILD_GENERAL1_MEDIUM\"}" \
    --service-role "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role"
```

输出显示了更新的设置。

```
{
    "project": {
        "arn": "arn:aws:codebuild:us-west-2:123456789012:project/my-demo-project",
        "environment": {
            "privilegedMode": false,
            "environmentVariables": [],
            "type": "LINUX_CONTAINER",
            "image": "aws/codebuild/standard:1.0",
            "computeType": "BUILD_GENERAL1_MEDIUM",
            "imagePullCredentialsType": "CODEBUILD"
        },
        "queuedTimeoutInMinutes": 480,
        "description": "This project is updated",
        "artifacts": {
            "packaging": "NONE",
            "name": "my-demo-project",
            "type": "S3",
            "namespaceType": "NONE",
            "encryptionDisabled": false,
            "location": "codebuild-us-west-2-123456789012-output-bucket-2"
        },
        "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
        "badge": {
            "badgeEnabled": false
        },
        "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
        "lastModified": 1556840545.967,
        "tags": [],
        "timeoutInMinutes": 60,
```

```
        "created": 1556839783.274,
        "name": "my-demo-project",
        "cache": {
            "type": "NO_CACHE"
        },
        "source": {
            "type": "S3",
            "insecureSsl": false,
            "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source-2.zip"
        }
    }
}
```

有关详细信息，请参阅《Amazon CodeBuild 用户指南》中的更改构建项目的设置（Amazon CLI）

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProject。

update-report-group

以下代码示例演示了如何使用 update-report-group。

Amazon CLI

在 Amazon CodeBuild 中更新报告组。

以下 update-report-group 示例将报告组的导出类型更改为“NO_EXPORT”。

```
aws codebuild update-report-group \
    --arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-report-
group \
    --export-config="exportConfigType=NO_EXPORT"
```

输出：

```
{
    "reportGroup": {
        "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-
report-group",
        "name": "cli-created-report-group",
        "type": "TEST",
        "exportConfig": {
            "exportConfigType": "NO_EXPORT"
```

```
        },
        "created": 1602020686.009,
        "lastModified": 1602021033.454,
        "tags": []
    }
}
```

有关更多信息，请参阅《Amazon CodeBuild 用户指南》中的使用报告组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateReportGroup。

update-webhook

以下代码示例演示了如何使用 update-webhook。

Amazon CLI

更新 Amazon CodeBuild 项目的 webhook

以下 update-webhook 示例使用两个筛选条件组为指定 CodeBuild 项目更新 webhook。该 --rotate-secret 参数指定，每次代码更改触发构建时，GitHub 都会轮换项目的密钥。第一个筛选条件组使用与正则表达式 ^refs/heads/master$ 匹配的 Git 引用名称以及与 ^refs/heads/myBranch$ 匹配的头部引用，指定在分支上创建、更新或重新打开的拉取请求。第二个筛选条件组使用与正则表达式 ^refs/heads/myBranch$ 不匹配的 Git 引用名称，指定分支上的推送请求。

```
aws codebuild update-webhook \
    --project-name Project2 \
    --rotate-secret \
    --filter-groups "[[{\"type\":\"EVENT\",\"pattern\":\"PULL_REQUEST_CREATED,
 PULL_REQUEST_UPDATED, PULL_REQUEST_REOPENED\"},{\"type\":\"HEAD_REF\",\"pattern
\":\"^refs/heads/myBranch$\",\"excludeMatchedPattern\":true},{\"type\":\"BASE_REF
\",\"pattern\":\"^refs/heads/master$\",\"excludeMatchedPattern\":true}],[{\"type\":
\"EVENT\",\"pattern\":\"PUSH\"},{\"type\":\"HEAD_REF\",\"pattern\":\"^refs/heads/
myBranch$\",\"excludeMatchedPattern\":true}]]"
```

输出：

```
{
    "webhook": {
        "filterGroups": [
            [
```

```
                {
                    "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
 PULL_REQUEST_REOPENED",
                    "type": "EVENT"
                },
                {
                    "excludeMatchedPattern": true,
                    "pattern": "refs/heads/myBranch$",
                    "type": "HEAD_REF"
                },
                {
                    "excludeMatchedPattern": true,
                    "pattern": "refs/heads/master$",
                    "type": "BASE_REF"
                }
            ],
            [
                {
                    "pattern": "PUSH",
                    "type": "EVENT"
                },
                {
                    "excludeMatchedPattern": true,
                    "pattern": "refs/heads/myBranch$",
                    "type": "HEAD_REF"
                }
            ]
        ],
        "lastModifiedSecret": 1556312220.133
    }
}
```

有关详细信息，请参阅《Amazon CodeBuild 用户指南》中的更改构建项目的设置（Amazon CLI）

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWebhook。

使用 Amazon CLI 的 CodeCommit 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeCommit 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-approval-rule-template-with-repository`

以下代码示例演示了如何使用 `associate-approval-rule-template-with-repository`。

Amazon CLI

将审批规则模板与存储库关联

以下 `associate-approval-rule-template-with-repository` 示例将指定的审批规则模板与名为 `MyDemoRepo` 的存储库关联。

```
aws codecommit associate-approval-rule-template-with-repository \
    --repository-name MyDemoRepo  \
    --approval-rule-template-name 2-approver-rule-for-main
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的将审批规则模板与存储库关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateApprovalRuleTemplateWithRepository。

`batch-associate-approval-rule-template-with-repositories`

以下代码示例演示了如何使用 `batch-associate-approval-rule-template-with-repositories`。

Amazon CLI

在单个操作中将一个审批规则模板与多个存储库关联

以下 `batch-associate-approval-rule-template-with-repositories` 示例将指定的审批规则模板与名为 `MyDemoRepo` 和 `MyOtherDemoRepo` 的存储库关联。

请注意：审批规则模板特定于创建模板时所处的 Amazon 区域。它们只能与该 Amazon 区域的存储库关联。

```
aws codecommit batch-associate-approval-rule-template-with-repositories \
    --repository-names MyDemoRepo, MyOtherDemoRepo  \
    --approval-rule-template-name 2-approver-rule-for-main
```

输出：

```
{
    "associatedRepositoryNames": [
        "MyDemoRepo",
        "MyOtherDemoRepo"
    ],
    "errors": []
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的将审批规则模板与存储库关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchAssociateApprovalRuleTemplateWithRepositories。

batch-describe-merge-conflicts

以下代码示例演示了如何使用 batch-describe-merge-conflicts。

Amazon CLI

获取有关在两个提交说明符之间的合并中所有文件或部分文件合并冲突的信息

以下 batch-describe-merge-conflicts 示例确定在名为 MyDemoRepo 的存储库中合并名为 feature-randomizationfeature 的源分支与使用 THREE_WAY_MERGE 策略的名为 main 的目标分支合并时产生的合并冲突。

```
aws codecommit batch-describe-merge-conflicts \
    --source-commit-specifier feature-randomizationfeature \
    --destination-commit-specifier main \
    --merge-option THREE_WAY_MERGE \
    --repository-name MyDemoRepo
```

输出：

```
{
    "conflicts": [
        {
            "conflictMetadata": {
                "filePath": "readme.md",
                "fileSizes": {
                    "source": 139,
                    "destination": 230,
                    "base": 85
                },
                "fileModes": {
                    "source": "NORMAL",
                    "destination": "NORMAL",
                    "base": "NORMAL"
                },
                "objectTypes": {
                    "source": "FILE",
                    "destination": "FILE",
                    "base": "FILE"
                },
                "numberOfConflicts": 1,
                "isBinaryFile": {
                    "source": false,
                    "destination": false,
                    "base": false
                },
                "contentConflict": true,
                "fileModeConflict": false,
                "objectTypeConflict": false,
                "mergeOperations": {
                    "source": "M",
                    "destination": "M"
                }
            },
            "mergeHunks": [
                {
                    "isConflict": true,
                    "source": {
                        "startLine": 0,
                        "endLine": 3,
                        "hunkContent": "VGhpcyBpEXAMPLE=="
                    },
                    "destination": {
```

```
                        "startLine": 0,
                        "endLine": 1,
                        "hunkContent": "VXNlIHRoEXAMPLE="
                    }
                }
            ]
        }
    ],
    "errors": [],
    "destinationCommitId": "86958e0aEXAMPLE",
    "sourceCommitId": "6ccd57fdEXAMPLE",
    "baseCommitId": "767b6958EXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解决拉取请求中的冲突。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDescribeMergeConflicts。

batch-disassociate-approval-rule-template-from-repositories

以下代码示例演示了如何使用 batch-disassociate-approval-rule-template-from-repositories。

Amazon CLI

在单个操作中解除审批规则模板与多个存储库的关联

以下 batch-disassociate-approval-rule-template-from-repositories 示例解除指定的审批规则模板与名为 MyDemoRepo 和 MyOtherDemoRepo 的存储库关联。

```
aws codecommit batch-disassociate-approval-rule-template-from-repositories \
    --repository-names MyDemoRepo, MyOtherDemoRepo  \
    --approval-rule-template-name 1-approval-rule-for-all pull requests
```

输出：

```
{
    "disassociatedRepositoryNames": [
        "MyDemoRepo",
        "MyOtherDemoRepo"
    ],
    "errors": []
```

```
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解除审批规则模板的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDisassociateApprovalRuleTemplateFromRepositories。

batch-get-commits

以下代码示例演示了如何使用 batch-get-commits。

Amazon CLI

查看有关多个提交的信息

以下 batch-get-commits 示例显示指定提交的详细信息。

```
aws codecommit batch-get-commits  \
    --repository-name MyDemoRepo  \
    --commit-ids 317f8570EXAMPLE 4c925148EXAMPLE
```

输出：

```
{
    "commits": [
      {
        "additionalData": "",
        "committer": {
            "date": "1508280564 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "author": {
            "date": "1508280564 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "commitId": "317f8570EXAMPLE",
        "treeId": "1f330709EXAMPLE",
        "parents": [
            "6e147360EXAMPLE"
        ],
```

```
        "message": "Change variable name and add new response element"
    },
    {
        "additionalData": "",
        "committer": {
            "date": "1508280542 -0800",
            "name": "Li Juan",
            "email": "li_juan@example.com"
        },
        "author": {
            "date": "1508280542 -0800",
            "name": "Li Juan",
            "email": "li_juan@example.com"
        },
        "commitId": "4c925148EXAMPLE",
        "treeId": "1f330709EXAMPLE",
        "parents": [
            "317f8570EXAMPLE"
        ],
        "message": "Added new class"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的查看提交的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetCommits。

batch-get-repositories

以下代码示例演示了如何使用 batch-get-repositories。

Amazon CLI

查看有关多个存储库的详细信息

此示例显示了有关多个 Amazon CodeCommit 存储库的详细信息。

```
aws codecommit batch-get-repositories \
    --repository-names MyDemoRepo MyOtherDemoRepo
```

输出：

```
{
```

```
    "repositoriesNotFound": [],
    "repositories": [
         {
            "creationDate": 1429203623.625,
            "defaultBranch": "main",
            "repositoryName": "MyDemoRepo",
            "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
            "lastModifiedDate": 1430783812.0869999,
            "repositoryDescription": "My demonstration repository",
            "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
            "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
            "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
            "accountId": "111111111111"
        },
        {
            "creationDate": 1429203623.627,
            "defaultBranch": "main",
            "repositoryName": "MyOtherDemoRepo",
            "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyOtherDemoRepo",
            "lastModifiedDate": 1430783812.0889999,
            "repositoryDescription": "My other demonstration repository",
            "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/
MyOtherDemoRepo",
            "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE",
            "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"
            "accountId": "111111111111"
        }
    ],
    "repositoriesNotFound": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetRepositories。

create-approval-rule-template

以下代码示例演示了如何使用 create-approval-rule-template。

Amazon CLI

创建审批规则模板

以下 create-approval-rule-template 示例创建一个名为 2-approver-rule-for-main ``. The template requires two users who assume the role of ``CodeCommitReview 的审批规则模板，用于批准任何拉取请求，然后再将其合并到 main 分支。

```
aws codecommit create-approval-rule-template \
    --approval-rule-template-name 2-approver-rule-for-main \
    --approval-rule-template-description  "Requires two developers from the team to
 approve the pull request if the destination branch is main" \
    --approval-rule-template-content "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

输出：

```
{
    "approvalRuleTemplate": {
        "approvalRuleTemplateName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "approvalRuleTemplateDescription": "Requires two developers from the team to
 approve the pull request if the destination branch is main",
        "lastModifiedDate": 1571356106.936,
        "ruleContentSha256": "4711b576EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApprovalRuleTemplate。

create-branch

以下代码示例演示了如何使用 create-branch。

Amazon CLI

创建分支

此示例在 Amazon CodeCommit 存储库中创建了一个分支。该命令只在出现错误时生成输出。

命令:

```
aws codecommit create-branch --repository-name MyDemoRepo --branch-name MyNewBranch
 --commit-id 317f8570EXAMPLE
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBranch。

create-commit

以下代码示例演示了如何使用 create-commit。

Amazon CLI

创建提交

以下 create-commit 示例演示了如何为存储库创建初始提交，以便将 readme.md 文件添加到 main 分支中名为 MyDemoRepo 的存储库。

```
aws codecommit create-commit \
    --repository-name MyDemoRepo \
    --branch-name main \
    --put-files "filePath=readme.md,fileContent='Welcome to our team repository.'"
```

输出：

```
{
    "filesAdded": [
        {
            "blobId": "5e1c309d-EXAMPLE",
            "absolutePath": "readme.md",
```

```
            "fileMode": "NORMAL"
        }
    ],
    "commitId": "4df8b524-EXAMPLE",
    "treeId": "55b57003-EXAMPLE",
    "filesDeleted": [],
    "filesUpdated": []
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的在 Amazon CodeCommit 中创建提交。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCommit。

create-pull-request-approval-rule

以下代码示例演示了如何使用 create-pull-request-approval-rule。

Amazon CLI

创建拉取请求的审批规则

以下 create-pull-request-approval-rule 示例为指定拉取请求创建名为 Require two approved approvers 的审批规则。该规则指定审批池中需要存在两个审批。池中包括通过在 123456789012 Amazon 账户中代入 CodeCommitReview 角色来访问 CodeCommit 的所有用户。还包括同一 Amazon 账户中的 IAM 用户或名为 Nikhil_Jayashankar 的联合用户。

```
aws codecommit create-pull-request-approval-rule  \
    --approval-rule-name "Require two approved approvers"  \
    --approval-rule-content "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
 \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

输出：

```
{
    "approvalRule": {
        "approvalRuleName": "Require two approved approvers",
        "lastModifiedDate": 1570752871.932,
        "ruleContentSha256": "7c44e6ebEXAMPLE",
        "creationDate": 1570752871.932,
```

```
        "approvalRuleId": "aac33506-EXAMPLE",
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
 \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建审批规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePullRequestApprovalRule。

`create-pull-request`

以下代码示例演示了如何使用 `create-pull-request`。

Amazon CLI

创建拉取请求

以下 `create-pull-request` 示例创建一个名为“发音难度分析器”的拉取请求，其描述为“请在周二之前查看这些更改”，目标是“jane-branch”源分支，并将合并到名为“MyDemoRepo”的 Amazon CodeCommit 存储库中的默认分支“main”中。

```
aws codecommit create-pull-request \
    --title "My Pull Request" \
    --description "Please review these changes by Tuesday" \
    --client-request-token 123Example \
    --targets repositoryName=MyDemoRepo,sourceReference=MyNewBranch
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
```

```
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd3d22fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::111111111111:user/Jane_Doe",
        "description": "Please review these changes by Tuesday",
        "title": "Pronunciation difficulty analyzer",
        "pullRequestTargets": [
            {
                "destinationCommit": "5d036259EXAMPLE",
                "destinationReference": "refs/heads/main",
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "317f8570EXAMPLE",
                "sourceReference": "refs/heads/jane-branch",
                "mergeMetadata": {
                    "isMerged": false
                }
            }
        ],
        "lastActivityDate": 1508962823.285,
        "pullRequestId": "42",
        "clientRequestToken": "123Example",
        "pullRequestStatus": "OPEN",
        "creationDate": 1508962823.285
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePullRequest。

`create-repository`

以下代码示例演示了如何使用 `create-repository`。

Amazon CLI

创建存储库 ()

此示例创建一个存储库并将其与用户的 Amazon 帐户相关联。

命令:

```
aws codecommit create-repository --repository-name MyDemoRepo --repository-
description "My demonstration repository"
```

输出：

```
{
    "repositoryMetadata": {
        "repositoryName": "MyDemoRepo",
                "cloneUrlSsh": "ssh://git-codecommit.us-east-1.amazonaws.com/v1/
repos/MyDemoRepo",
                "lastModifiedDate": 1444766838.027,
        "repositoryDescription": "My demonstration repository",
                "cloneUrlHttp": "https://git-codecommit.us-east-1.amazonaws.com/v1/
repos/MyDemoRepo",
        "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
                "Arn": "arn:aws:codecommit:us-
east-1:111111111111EXAMPLE:MyDemoRepo",
        "accountId": "111111111111"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRepository。

create-unreferenced-merge-commit

以下代码示例演示了如何使用 create-unreferenced-merge-commit。

Amazon CLI

创建表示两个提交说明符的合并结果的未引用提交

以下 create-unreferenced-merge-commit 示例创建一个提交，表示在名为 MyDemoRepo 的存储库中，使用 THREE_WAY_MERGE 策略在名为 bugfix-1234 的源分支与名为 main 的目标分支之间的合并结果。

```
aws codecommit create-unreferenced-merge-commit \
    --source-commit-specifier bugfix-1234 \
```

```
    --destination-commit-specifier main \
    --merge-option THREE_WAY_MERGE \
    --repository-name MyDemoRepo \
    --name "Maria Garcia" \
    --email "maria_garcia@example.com" \
    --commit-message "Testing the results of this merge."
```

输出：

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解决拉取请求中的冲突。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUnreferencedMergeCommit。

credential-helper

以下代码示例演示了如何使用 credential-helper。

Amazon CLI

使用 Amazon CodeCommit 设置 Amazon CLI 中包含的凭证助手

此 credential-helper 实用程序没有设计为直接从 Amazon CLI 调用。相反，它旨在用作设置本地计算机的 git config 命令的参数。每当 Git 需要通过向 Amazon 进行身份验证来与 CodeCommit 存储库进行交互时，它都允许 Git 使用 HTTPS 和 IAM 用户凭证或 Amazon EC2 实例角色的加密签名版本。

```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
```

输出：

```
[credential]
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的“使用其他方法设置 Amazon CodeCommit”。仔细阅读内容，然后按照以下主题之一中的步骤操作：《Amazon CodeCommit 用户指南》中的“在 Linux、macOS 或 Unix 上建立 HTTPS 连接”或“在 Windows 上建立 HTTPS 连接”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CredentialHelper。

`delete-approval-rule-template`

以下代码示例演示了如何使用 `delete-approval-rule-template`。

Amazon CLI

删除审批规则模板

以下 `delete-approval-rule-template` 示例删除指定审批规则模板。

```
aws codecommit delete-approval-rule-template  \
    --approval-rule-template-name 1-approver-for-all-pull-requests
```

输出：

```
{
    "approvalRuleTemplateId": "41de97b7-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的删除审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApprovalRuleTemplate。

`delete-branch`

以下代码示例演示了如何使用 `delete-branch`。

Amazon CLI

删除分支

此示例说明如何删除 Amazon CodeCommit 存储库中的分支。

命令:

```
aws codecommit delete-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

输出：

```
{
  "branch": {
      "commitId": "317f8570EXAMPLE",
      "branchName": "MyNewBranch"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBranch。

delete-comment-content

以下代码示例演示了如何使用 delete-comment-content。

Amazon CLI

删除评论的内容

如果您创建了评论，则只能删除该评论的内容。此示例演示如何删除系统生成的 ID 为 ff30b348EXAMPLEb9aa670f 的评论内容。

```
aws codecommit delete-comment-content \
    --comment-id ff30b348EXAMPLEb9aa670f
```

输出：

```
{
    "comment": {
        "creationDate": 1508369768.142,
        "deleted": true,
        "lastModifiedDate": 1508369842.278,
        "clientRequestToken": "123Example",
        "commentId": "ff30b348EXAMPLEb9aa670f",
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "callerReactions": [],
        "reactionCounts":
        {
```

```
            "CLAP" : 1
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCommentContent。

delete-file

以下代码示例演示了如何使用 delete-file。

Amazon CLI

删除文件

以下 delete-file 示例演示如何在名为 MyDemoRepo 的存储库中删除名为 main 的分支中名为 README.md 的文件，该分支的最新提交 ID 为 c5709475EXAMPLE。

```
aws codecommit delete-file \
    --repository-name MyDemoRepo \
    --branch-name main \
    --file-path README.md \
    --parent-commit-id c5709475EXAMPLE
```

输出：

```
{
    "blobId":"559b44fEXAMPLE",
    "commitId":"353cf655EXAMPLE",
    "filePath":"README.md",
    "treeId":"6bc824cEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit API 参考指南》中的在 Amazon CodeCommit 中编辑或删除文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFile。

delete-pull-request-approval-rule

以下代码示例演示了如何使用 delete-pull-request-approval-rule。

Amazon CLI

删除拉取请求的审批规则

以下 delete-pull-request-approval-rule 示例删除指定拉取请求的名为 My Approval Rule 的审批规则。

```
aws codecommit delete-pull-request-approval-rule  \
    --approval-rule-name "My Approval Rule"  \
    --pull-request-id 15
```

输出：

```
{
    "approvalRuleId": "077d8e8a8-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的编辑或删除审批规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePullRequestApprovalRule。

delete-repository

以下代码示例演示了如何使用 delete-repository。

Amazon CLI

删除存储库（ ）

此示例说明如何删除 Amazon CodeCommit 存储库。

命令:

```
aws codecommit delete-repository --repository-name MyDemoRepo
```

输出：

```
{
  "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepository。

describe-merge-conflicts

以下代码示例演示了如何使用 describe-merge-conflicts。

Amazon CLI

获取有关合并冲突的详细信息

以下 describe-merge-conflicts 示例使用 THREE_WAY_MERGE 策略确定指定源分支和目标分支中名为 readme.md 的文件的合并冲突。

```
aws codecommit describe-merge-conflicts \
    --source-commit-specifier feature-randomizationfeature \
    --destination-commit-specifier main \
    --merge-option THREE_WAY_MERGE \
    --file-path readme.md \
    --repository-name MyDemoRepo
```

输出：

```
{
    "conflictMetadata": {
        "filePath": "readme.md",
        "fileSizes": {
            "source": 139,
            "destination": 230,
            "base": 85
        },
        "fileModes": {
            "source": "NORMAL",
            "destination": "NORMAL",
            "base": "NORMAL"
        },
        "objectTypes": {
            "source": "FILE",
            "destination": "FILE",
            "base": "FILE"
        },
        "numberOfConflicts": 1,
        "isBinaryFile": {
```

```
            "source": false,
            "destination": false,
            "base": false
        },
        "contentConflict": true,
        "fileModeConflict": false,
        "objectTypeConflict": false,
        "mergeOperations": {
            "source": "M",
            "destination": "M"
        }
    },
    "mergeHunks": [
        {
            "isConflict": true,
            "source": {
                "startLine": 0,
                "endLine": 3,
                "hunkContent": "VGhpcyBpEXAMPLE="
            },
            "destination": {
                "startLine": 0,
                "endLine": 1,
                "hunkContent": "VXNlIHRoEXAMPLE="
            }
        }
    ],
    "destinationCommitId": "86958e0aEXAMPLE",
    "sourceCommitId": "6ccd57fdEXAMPLE",
    "baseCommitId": "767b69580EXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解决拉取请求中的冲突。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMergeConflicts。

`describe-pull-request-events`

以下代码示例演示了如何使用 `describe-pull-request-events`。

Amazon CLI

查看拉取请求中的事件

以下 describe-pull-request-events 示例检索 ID 为“8”的拉取请求的事件。

```
aws codecommit describe-pull-request-events --pull-request-id 8
```

输出：

```
{
    "pullRequestEvents": [
        {
            "pullRequestId": "8",
            "pullRequestEventType": "PULL_REQUEST_CREATED",
            "eventDate": 1510341779.53,
            "actor": "arn:aws:iam::111111111111:user/Zhang_Wei"
        },
        {
            "pullRequestStatusChangedEventMetadata": {
                "pullRequestStatus": "CLOSED"
            },
            "pullRequestId": "8",
            "pullRequestEventType": "PULL_REQUEST_STATUS_CHANGED",
            "eventDate": 1510341930.72,
            "actor": "arn:aws:iam::111111111111:user/Jane_Doe"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePullRequestEvents。

disassociate-approval-rule-template-from-repository

以下代码示例演示了如何使用 disassociate-approval-rule-template-from-repository。

Amazon CLI

解除审批规则模板与存储库的关联

以下 disassociate-approval-rule-template-from-repository 示例解除指定的审批规则模板与名为 MyDemoRepo 的存储库的关联。

```
aws codecommit disassociate-approval-rule-template-from-repository \
    --repository-name MyDemoRepo  \
```

```
    --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解除审批规则模板的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateApprovalRuleTemplateFromRepository。

evaluate-pull-request-approval-rules

以下代码示例演示了如何使用 evaluate-pull-request-approval-rules。

Amazon CLI

评估拉取请求是否满足其所有审批规则

以下 evaluate-pull-request-approval-rules 示例评估指定拉取请求的审批规则的状态。在此示例中，拉取请求未满足审批规则，因此命令输出显示 approved 值为 false。

```
aws codecommit evaluate-pull-request-approval-rules \
    --pull-request-id 27  \
    --revision-id 9f29d167EXAMPLE
```

输出：

```
{
    "evaluation": {
        "approved": false,
        "approvalRulesNotSatisfied": [
            "Require two approved approvers"
        ],
        "overridden": false,
        "approvalRulesSatisfied": []
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的合并拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EvaluatePullRequestApprovalRules。

get-approval-rule-template

以下代码示例演示了如何使用 get-approval-rule-template。

Amazon CLI

获取审批规则模板的内容

以下 get-approval-rule-template 示例获取名为 1-approver-rule-for-all-pull-requests 的审批规则模板的内容。

```
aws codecommit get-approval-rule-template \
    --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

输出：

```
{
    "approvalRuleTemplate": {
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "ruleContentSha256": "621181bbEXAMPLE",
        "lastModifiedDate": 1571356106.936,
        "creationDate": 1571356106.936,
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan",
        "approvalRuleTemplateId": "a29abb15-EXAMPLE",
        "approvalRuleTemplateDescription": "All pull requests must be approved by
 one developer on the team."
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApprovalRuleTemplate。

get-blob

以下代码示例演示了如何使用 get-blob。

Amazon CLI

查看有关 Git blob 对象的信息

以下 get-blob 示例检索名为“MyDemoRepo”的 Amazon CodeCommit 存储库中 ID 为“2eb4af3bEXAMPLE”的 Git blob 的相关信息。

```
aws codecommit get-blob  --repository-name MyDemoRepo  --blob-id 2eb4af3bEXAMPLE
```

输出：

```
{
    "content": "QSBCaW5hcnkgTGFyToEXAMPLE="
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBlob。

get-branch

以下代码示例演示了如何使用 get-branch。

Amazon CLI

查看有关分支的信息

此示例获取有关 Amazon CodeCommit 存储库中分支的信息。

命令:

```
aws codecommit get-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

输出：

```
{
  "BranchInfo": {
        "commitID": "317f8570EXAMPLE",
                "branchName": "MyNewBranch"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBranch。

get-comment-reactions

以下代码示例演示了如何使用 get-comment-reactions。

Amazon CLI

查看对评论的表情符号反应

以下 get-comment-reactions 示例列出对 ID 为 abcd1234EXAMPLEb5678efgh 的评论的所有表情符号反应。如果您的 shell 的字体支持显示表情符号版本 1.0，则会在 emoji 输出中显示表情符号。

```
aws codecommit get-comment-reactions \
    --comment-id abcd1234EXAMPLEb5678efgh
```

输出：

```
{
    "reactionsForComment": {
        [
           {
               "reaction": {
                    "emoji:"??",
                    "shortCode": "thumbsup",
                    "unicode": "U+1F44D"
                },
                "users": [
                    "arn:aws:iam::123456789012:user/Li_Juan",
                    "arn:aws:iam::123456789012:user/Mary_Major",
                    "arn:aws:iam::123456789012:user/Jorge_Souza"
                ]
            },
            {
                "reaction": {
                    "emoji": "??",
                    "shortCode": "thumbsdown",
                    "unicode": "U+1F44E"
                },
                "users": [
                    "arn:aws:iam::123456789012:user/Nikhil_Jayashankar"
                ]
            },
            {
                "reaction": {
                    "emoji": "??",
                    "shortCode": "confused",
                    "unicode": "U+1F615"
```

```
                },
                "users": [
                    "arn:aws:iam::123456789012:user/Saanvi_Sarkar"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的在 Amazon CodeCommit 中评论提交。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCommentReactions。

get-comment

以下代码示例演示了如何使用 get-comment。

Amazon CLI

查看有关评论的详细信息

此示例演示如何查看系统生成的评论 ID 为 ff30b348EXAMPLEb9aa670f 的评论的详细信息。

```
aws codecommit get-comment \
    --comment-id ff30b348EXAMPLEb9aa670f
```

输出：

```
{
    "comment": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "123Example",
        "commentId": "ff30b348EXAMPLEb9aa670f",
        "content": "Whoops - I meant to add this comment to the line, but I don't
 see how to delete it.",
        "creationDate": 1508369768.142,
        "deleted": false,
        "commentId": "",
        "lastModifiedDate": 1508369842.278,
        "callerReactions": [],
        "reactionCounts":
```

```
        {
            "SMILE" : 6,
            "THUMBSUP" : 1
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComment。

get-comments-for-compared-commit

以下代码示例演示了如何使用 get-comments-for-compared-commit。

Amazon CLI

查看对提交进行的评论

此示例演示了如何查看对名为 MyDemoRepo 的存储库中两个提交之间的比较做出的评论。

```
aws codecommit get-comments-for-compared-commit \
    --repository-name MyDemoRepo \
    --before-commit-ID 6e147360EXAMPLE \
    --after-commit-id 317f8570EXAMPLE
```

输出：

```
{
    "commentsForComparedCommitData": [
        {
            "afterBlobId": "1f330709EXAMPLE",
            "afterCommitId": "317f8570EXAMPLE",
            "beforeBlobId": "80906a4cEXAMPLE",
            "beforeCommitId": "6e147360EXAMPLE",
            "comments": [
                {
                    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
                    "clientRequestToken": "123Example",
                    "commentId": "ff30b348EXAMPLEb9aa670f",
                    "content": "Whoops - I meant to add this comment to the line,
 not the file, but I don't see how to delete it.",
                    "creationDate": 1508369768.142,
                    "deleted": false,
```

```
                    "CommentId": "123abc-EXAMPLE",
                    "lastModifiedDate": 1508369842.278,
                    "callerReactions": [],
                    "reactionCounts":
                    {
                        "SMILE" : 6,
                        "THUMBSUP" : 1
                    }
                },
                {
                    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
                    "clientRequestToken": "123Example",
                    "commentId": "553b509bEXAMPLE56198325",
                    "content": "Can you add a test case for this?",
                    "creationDate": 1508369612.240,
                    "deleted": false,
                    "commentId": "456def-EXAMPLE",
                    "lastModifiedDate": 1508369612.240,
                    "callerReactions": [],
                    "reactionCounts":
                    {
                        "THUMBSUP" : 2
                    }
                }
            ],
            "location": {
                "filePath": "cl_sample.js",
                "filePosition": 1232,
                "relativeFileVersion": "after"
            },
            "repositoryName": "MyDemoRepo"
        }
    ],
    "nextToken": "exampleToken"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCommentsForComparedCommit。

get-comments-for-pull-request

以下代码示例演示了如何使用 get-comments-for-pull-request。

Amazon CLI

查看拉取请求的评论

此示例演示了如何查看名为 MyDemoRepo 的存储库中的拉取请求的评论。

```
aws codecommit get-comments-for-pull-request \
    --repository-name MyDemoRepo \
    --before-commit-ID 317f8570EXAMPLE \
    --after-commit-id 5d036259EXAMPLE
```

输出：

```
{
    "commentsForPullRequestData": [
        {
            "afterBlobId": "1f330709EXAMPLE",
            "afterCommitId": "5d036259EXAMPLE",
            "beforeBlobId": "80906a4cEXAMPLE",
            "beforeCommitId": "317f8570EXAMPLE",
            "comments": [
                {
                    "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
                    "clientRequestToken": "",
                    "commentId": "abcd1234EXAMPLEb5678efgh",
                    "content": "These don't appear to be used anywhere. Can we
 remove them?",
                    "creationDate": 1508369622.123,
                    "deleted": false,
                    "lastModifiedDate": 1508369622.123,
                    "callerReactions": [],
                    "reactionCounts":
                    {
                        "THUMBSUP" : 6,
                        "CONFUSED" : 1
                    }
                },
                {
                    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
                    "clientRequestToken": "",
                    "commentId": "442b498bEXAMPLE5756813",
                    "content": "Good catch. I'll remove them.",
                    "creationDate": 1508369829.104,
```

```
                    "deleted": false,
                    "lastModifiedDate": 150836912.273,
                    "callerReactions": ["THUMBSUP"]
                    "reactionCounts":
                    {
                        "THUMBSUP" : 14
                    }
                }
            ],
            "location": {
                "filePath": "ahs_count.py",
                "filePosition": 367,
                "relativeFileVersion": "AFTER"
            },
            "repositoryName": "MyDemoRepo",
            "pullRequestId": "42"
        }
    ],
    "nextToken": "exampleToken"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCommentsForPullRequest。

get-commit

以下代码示例演示了如何使用 get-commit。

Amazon CLI

查看有关存储库中提交的信息

此示例显示 Amazon CodeCommit 存储库中名为“MyDemoRepo”的提交的详细信息，该提交的系统生成 ID 为“7e9fd3091thisisanexamplethisisanexample1”。

命令:

```
aws codecommit get-commit --repository-name MyDemoRepo --commit-
id 7e9fd3091thisisanexamplethisisanexample1
```

输出：

```
{
  "commit": {
```

```
      "additionalData": "",
      "committer": {
          "date": "1484167798 -0800",
          "name": "Mary Major",
          "email": "mary_major@example.com"
      },
      "author": {
          "date": "1484167798 -0800",
          "name": "Mary Major",
          "email": "mary_major@example.com"
      },
      "treeId": "347a3408thisisanexampletreeidexample",
      "parents": [
          "7aa87a031thisisanexamplethisisanexample1"
      ],
      "message": "Fix incorrect variable name"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCommit。

get-differences

以下代码示例演示了如何使用 get-differences。

Amazon CLI

获取有关存储库中提交说明符差异的信息

此示例显示了 Amazon CodeCommit 存储库中名为 MyDemoRepo 的重命名文件夹中查看两个提交说明符（分支、标记、HEAD 或其他完全限定参考，例如提交 ID）之间更改的元数据信息。该示例包括几个非必备选项，包括 --before-commit-specifier、--before-path 和 --after-path，以便更全面地说明如何使用这些选项来限制结果。响应包括文件模式权限。

命令:

```
aws codecommit get-differences --repository-name MyDemoRepo --before-
commit-specifier 955bba12thisisanexamplethisisanexample --after-commit-
specifier 14a95463thisisanexamplethisisanexample --before-path tmp/example-folder --
after-path tmp/renamed-folder
```

输出：

```
{
  "differences": [
      {
          "afterBlob": {
              "path": "blob.txt",
              "blobId": "2eb4af3b1thisisanexamplethisisanexample1",
              "mode": "100644"
          },
          "changeType": "M",
          "beforeBlob": {
              "path": "blob.txt",
              "blobId": "bf7fcf281thisisanexamplethisisanexample1",
              "mode": "100644"
          }
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDifferences。

get-file

以下代码示例演示了如何使用 get-file。

Amazon CLI

获取 Amazon CodeCommit 存储库中文件的 base-64 编码内容

以下 get-file 示例演示如何从名为 MyDemoRepo 的存储库中名为 main 的分支中获取名为 README.md 的文件的 base-64 编码内容。

```
aws codecommit get-file \
    --repository-name MyDemoRepo \
    --commit-specifier main \
    --file-path README.md
```

输出：

```
{
    "blobId":"559b44fEXAMPLE",
    "commitId":"c5709475EXAMPLE",
    "fileContent":"IyBQaHVzEXAMPLE",
```

```
    "filePath":"README.md",
    "fileMode":"NORMAL",
    "fileSize":1563
}
```

有关更多信息，请参阅《Amazon CodeCommit API 参考》指南中的 GetFile。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFile。

get-folder

以下代码示例演示了如何使用 get-folder。

Amazon CLI

获取 Amazon CodeCommit 存储库中某个文件夹的内容

以下 get-folder 示例演示如何从名为 MyDemoRepo 的存储库中获取顶层文件夹的内容。

```
aws codecommit get-folder --repository-name MyDemoRepo --folder-path ""
```

输出：

```
{
    "commitId":"c5709475EXAMPLE",
    "files":[
        {
            "absolutePath":".gitignore",
            "blobId":"74094e8bEXAMPLE",
            "fileMode":"NORMAL",
            "relativePath":".gitignore"
        },
        {
            "absolutePath":"Gemfile",
            "blobId":"9ceb72f6EXAMPLE",
            "fileMode":"NORMAL",
            "relativePath":"Gemfile"
        },
        {
            "absolutePath":"Gemfile.lock",
            "blobId":"795c4a2aEXAMPLE",
            "fileMode":"NORMAL",
            "relativePath":"Gemfile.lock"
```

```
        },
        {
            "absolutePath":"LICENSE.txt",
            "blobId":"0c7932c8EXAMPLE",
            "fileMode":"NORMAL",
            "relativePath":"LICENSE.txt"
        },
        {
            "absolutePath":"README.md",
            "blobId":"559b44feEXAMPLE",
            "fileMode":"NORMAL",
            "relativePath":"README.md"
        }
    ],
    "folderPath":"",
    "subFolders":[
        {
            "absolutePath":"public",
            "relativePath":"public",
            "treeId":"d5e92ae3aEXAMPLE"
        },
        {
            "absolutePath":"tmp",
            "relativePath":"tmp",
            "treeId":"d564d0bcEXAMPLE"
        }
    ],
    "subModules":[],
    "symbolicLinks":[],
    "treeId":"7b3c4dadEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit API 参考》指南中的 GetFolder。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFolder。

`get-merge-commit`

以下代码示例演示了如何使用 `get-merge-commit`。

Amazon CLI

获取有关合并提交的详细信息

以下 get-merge-commit 示例显示有关在名为 MyDemoRepo 的存储库中，名为 bugfix-bug1234 的源分支与名为 main 的目标分支合并提交的详细信息。

```
aws codecommit get-merge-commit \
    --source-commit-specifier bugfix-bug1234 \
    --destination-commit-specifier main \
    --repository-name MyDemoRepo
```

输出：

```
{
    "sourceCommitId": "c5709475EXAMPLE",
    "destinationCommitId": "317f8570EXAMPLE",
    "baseCommitId": "fb12a539EXAMPLE",
    "mergeCommitId": "ffc4d608eEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的查看提交的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMergeCommit。

get-merge-conflicts

以下代码示例演示了如何使用 get-merge-conflicts。

Amazon CLI

查看拉取请求是否存在合并冲突

以下 get-merge-conflicts 示例显示在名为 MyDemoRepo 的存储库中，名为 feature-randomizationfeature 的源分支的提示和名为“main”的目标分支的提示之间是否存在合并冲突。

```
aws codecommit get-merge-conflicts \
    --repository-name MyDemoRepo \
    --source-commit-specifier feature-randomizationfeature \
    --destination-commit-specifier main \
    --merge-option THREE_WAY_MERGE
```

输出：

```
{
    "mergeable": false,
    "destinationCommitId": "86958e0aEXAMPLE",
    "sourceCommitId": "6ccd57fdEXAMPLE",
    "baseCommitId": "767b6958EXAMPLE",
    "conflictMetadataList": [
        {
            "filePath": "readme.md",
            "fileSizes": {
                "source": 139,
                "destination": 230,
                "base": 85
            },
            "fileModes": {
                "source": "NORMAL",
                "destination": "NORMAL",
                "base": "NORMAL"
            },
            "objectTypes": {
                "source": "FILE",
                "destination": "FILE",
                "base": "FILE"
            },
            "numberOfConflicts": 1,
            "isBinaryFile": {
                "source": false,
                "destination": false,
                "base": false
            },
            "contentConflict": true,
            "fileModeConflict": false,
            "objectTypeConflict": false,
            "mergeOperations": {
                "source": "M",
                "destination": "M"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMergeConflicts。

get-merge-options

以下代码示例演示了如何使用 get-merge-options。

Amazon CLI

获取有关合并两个指定分支的合并选项的信息

以下 get-merge-options 示例确定在名为 MyDemoRepo 的存储库中，合并名为 bugfix-bug1234 的源分支与名为 main 的目标分支时可用的合并选项。

```
aws codecommit get-merge-options \
    --source-commit-specifier bugfix-bug1234 \
    --destination-commit-specifier main \
    --repository-name MyDemoRepo
```

输出：

```
{
    "mergeOptions": [
        "FAST_FORWARD_MERGE",
        "SQUASH_MERGE",
        "THREE_WAY_MERGE"
    ],
    "sourceCommitId": "18059494EXAMPLE",
    "destinationCommitId": "ffd3311dEXAMPLE",
    "baseCommitId": "ffd3311dEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的解决拉取请求中的冲突。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMergeOptions。

get-pull-request-approval-states

以下代码示例演示了如何使用 get-pull-request-approval-states。

Amazon CLI

查看拉取请求的审批

以下 get-pull-request-approval-states 示例返回指定拉取请求的审批。

```
aws codecommit get-pull-request-approval-states \
    --pull-request-id 8 \
    --revision-id 9f29d167EXAMPLE
```

输出：

```
{
    "approvals": [
        {
            "userArn": "arn:aws:iam::123456789012:user/Mary_Major",
            "approvalState": "APPROVE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的查看拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPullRequestApprovalStates。

get-pull-request-override-state

以下代码示例演示了如何使用 get-pull-request-override-state。

Amazon CLI

获取有关拉取请求覆盖状态的信息

以下 get-pull-request-override-state 示例返回指定拉取请求的覆盖状态。在此示例中，名为 Mary Major 的用户覆盖了拉取请求的审批规则，因此输出返回的值为 true：

```
aws codecommit get-pull-request-override-state \
    --pull-request-id 34  \
    --revision-id 9f29d167EXAMPLE
```

输出：

```
{
    "overridden": true,
    "overrider": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的在拉取请求上覆盖审批规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPullRequestOverrideState。

get-pull-request

以下代码示例演示了如何使用 get-pull-request。

Amazon CLI

查看拉取请求的详细信息

此示例演示如何查看 ID 为 27 的拉取请求的相关信息。

```
aws codecommit get-pull-request \
    --pull-request-id 27
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "lastActivityDate": 1562619583.565,
        "pullRequestTargets": [
            {
                "sourceCommit": "ca45e279EXAMPLE",
                "sourceReference": "refs/heads/bugfix-1234",
                "mergeBase": "a99f5ddbEXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false
                },
                "destinationCommit": "2abfc6beEXAMPLE",
```

```
                "repositoryName": "MyDemoRepo"
            }
        ],
        "revisionId": "e47def21EXAMPLE",
        "title": "Quick fix for bug 1234",
        "authorArn": "arn:aws:iam::123456789012:user/Nikhil_Jayashankar",
        "clientRequestToken": "d8d7612e-EXAMPLE",
        "creationDate": 1562619583.565,
        "pullRequestId": "27",
        "pullRequestStatus": "OPEN"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPullRequest。

get-repository-triggers

以下代码示例演示了如何使用 get-repository-triggers。

Amazon CLI

获取有关存储库中触发器的信息

此示例显示有关为名为 MyDemoRepo 的 Amazon CodeCommit 存储库配置的触发器的详细信息。

```
aws codecommit get-repository-triggers \
    --repository-name MyDemoRepo
```

输出：

```
{
    "configurationId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
    "triggers": [
        {
            "destinationArn": "arn:aws:sns:us-
east-1:111111111111:MyCodeCommitTopic",
            "branches": [
                "main",
                "preprod"
            ],
            "name": "MyFirstTrigger",
            "customData": "",
```

```
            "events": [
                "all"
            ]
        },
        {
            "destinationArn": "arn:aws:lambda:us-
east-1:111111111111:function:MyCodeCommitPythonFunction",
            "branches": [],
            "name": "MySecondTrigger",
            "customData": "EXAMPLE",
            "events": [
                "all"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepositoryTriggers。

get-repository

以下代码示例演示了如何使用 get-repository。

Amazon CLI

获取有关存储库的信息

此示例显示了有关 Amazon CodeCommit 存储库的详细信息。

```
aws codecommit get-repository \
    --repository-name MyDemoRepo
```

输出：

```
{
    "repositoryMetadata": {
        "creationDate": 1429203623.625,
        "defaultBranch": "main",
        "repositoryName": "MyDemoRepo",
        "cloneUrlSsh": "ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/v1/
repos/MyDemoRepo",
        "lastModifiedDate": 1430783812.0869999,
        "repositoryDescription": "My demonstration repository",
```

```
        "cloneUrlHttp": "https://codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo",
        "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
        "Arn": "arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo
        "accountId": "111111111111"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepository。

list-approval-rule-templates

以下代码示例演示了如何使用 list-approval-rule-templates。

Amazon CLI

列出 Amazon 区域中的所有审批规则模板

以下 list-approval-rule-templates 示例列出指定区域中的所有审批规则模板。如果未将 Amazon 区域指定为参数，则该命令将返回用于运行该命令的 Amazon CLI 配置文件中指定区域的审批规则模板。

```
aws codecommit list-approval-rule-templates \
    --region us-east-2
```

输出：

```
{
    "approvalRuleTemplateNames": [
        "2-approver-rule-for-main",
        "1-approver-rule-for-all-pull-requests"
    ]
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListApprovalRuleTemplates。

list-associated-approval-rule-templates-for-repository

以下代码示例演示了如何使用 list-associated-approval-rule-templates-for-repository。

Amazon CLI

列出与存储库关联的所有模板

以下 `list-associated-approval-rule-templates-for-repository` 示例列出所有与名为 MyDemoRepo 的存储库关联的审批规则模板。

```
aws codecommit list-associated-approval-rule-templates-for-repository \
    --repository-name MyDemoRepo
```

输出：

```
{
  "approvalRuleTemplateNames": [
    "2-approver-rule-for-main",
    "1-approver-rule-for-all-pull-requests"
  ]
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssociatedApprovalRuleTemplatesForRepository。

list-branches

以下代码示例演示了如何使用 `list-branches`。

Amazon CLI

查看分支名称列表

此示例列出 Amazon CodeCommit 存储库中的所有分支名称。

```
aws codecommit list-branches \
    --repository-name MyDemoRepo
```

输出：

```
{
    "branches": [
        "MyNewBranch",
```

```
        "main"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBranches。

list-pull-requests

以下代码示例演示了如何使用 list-pull-requests。

Amazon CLI

查看存储库中拉取请求的列表

此示例演示了如何列出名为“MyDemoRepo”的 Amazon CodeCommit 存储库中的由 ARN 为“arn:aws:iam::111111111111:user/Li_Juan”且状态为“CLOSED”的 IAM 用户创建的拉取请求：

```
aws codecommit list-pull-requests --author-arn arn:aws:iam::111111111111:user/
Li_Juan --pull-request-status CLOSED --repository-name MyDemoRepo
```

输出：

```
{
 "nextToken": "",
 "pullRequestIds": ["2","12","16","22","23","35","30","39","47"]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPullRequests。

list-repositories-for-approval-rule-template

以下代码示例演示了如何使用 list-repositories-for-approval-rule-template。

Amazon CLI

列出与模板关联的所有存储库

以下 list-repositories-for-approval-rule-template 示例列出与指定审批规则模板关联的所有存储库。

```
aws codecommit list-repositories-for-approval-rule-template \
```

```
    --approval-rule-template-name 2-approver-rule-for-main
```

输出：

```
{
    "repositoryNames": [
        "MyDemoRepo",
        "MyClonedRepo"
    ]
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRepositoriesForApprovalRuleTemplate。

list-repositories

以下代码示例演示了如何使用 `list-repositories`。

Amazon CLI

查看存储库列表

此示例列出了与用户的 Amazon 帐户关联的所有 Amazon CodeCommit 存储库。

命令:

```
aws codecommit list-repositories
```

输出：

```
{
  "repositories": [
      {
         "repositoryName": "MyDemoRepo"
         "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
      },
      {
         "repositoryName": "MyOtherDemoRepo"
         "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE"
      }
  ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRepositories。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

查看存储库的 Amazon 标签

以下 list-tags-for-resource 示例列出指定存储库的标签键和标签值。

```
aws codecommit list-tags-for-resource \
    --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo
```

输出：

```
{
    "tags": {
        "Status": "Secret",
        "Team": "Saanvi"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的查看存储库的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

merge-branches-by-fast-forward

以下代码示例演示了如何使用 merge-branches-by-fast-forward。

Amazon CLI

使用快速转发合并策略合并两个分支

以下 merge-branches-by-fast-forward 示例将指定的源分支与名为 MyDemoRepo 的存储库中的指定目标分支合并。

```
aws codecommit merge-branches-by-fast-forward \
```

```
    --source-commit-specifier bugfix-bug1234 \
    --destination-commit-specifier bugfix-bug1233 \
    --repository-name MyDemoRepo
```

输出：

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的比较和合并分支。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergeBranchesByFastForward。

merge-branches-by-squash

以下代码示例演示了如何使用 merge-branches-by-squash。

Amazon CLI

使用 squash 合并策略合并两个分支

以下 merge-branches-by-squash 示例将指定的源分支与名为 MyDemoRepo 的存储库中的指定目标分支合并。

```
aws codecommit merge-branches-by-squash \
    --source-commit-specifier bugfix-bug1234 \
    --destination-commit-specifier bugfix-bug1233 \
    --author-name "Maria Garcia" \
    --email "maria_garcia@example.com" \
    --commit-message "Merging two fix branches to prepare for a general patch." \
    --repository-name MyDemoRepo
```

输出：

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的比较和合并分支。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergeBranchesBySquash。

merge-branches-by-three-way

以下代码示例演示了如何使用 merge-branches-by-three-way。

Amazon CLI

使用三向合并策略合并两个分支

以下 merge-branches-by-three-way 示例将指定的源分支与名为 MyDemoRepo 的存储库中的指定目标分支合并。

```
aws codecommit merge-branches-by-three-way \
    --source-commit-specifier main \
    --destination-commit-specifier bugfix-bug1234 \
    --author-name "Jorge Souza" --email "jorge_souza@example.com" \
    --commit-message "Merging changes from main to bugfix branch before additional
 testing." \
    --repository-name MyDemoRepo
```

输出：

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的比较和合并分支。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergeBranchesByThreeWay。

merge-pull-request-by-fast-forward

以下代码示例演示了如何使用 merge-pull-request-by-fast-forward。

Amazon CLI

合并并关闭拉取请求

此示例演示了如何在名为 MyDemoRepo 的存储库中合并并关闭 ID 为 47 且源提交 ID 为“99132ab0EXAMPLE”的拉取请求。

```
aws codecommit merge-pull-request-by-fast-forward \
    --pull-request-id 47 \
    --source-commit-id 99132ab0EXAMPLE \
    --repository-name MyDemoRepo
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "I want one approver for this pull request",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.142,
        "description": "Review the latest changes and updates to the global
 variables",
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
```

```
        "title": "Consolidation of global variables"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的合并拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergePullRequestByFastForward。

merge-pull-request-by-squash

以下代码示例演示了如何使用 merge-pull-request-by-squash。

Amazon CLI

使用 squash 合并策略合并拉取请求

以下 merge-pull-request-by-squash 示例在名为 MyDemoRepo 的存储库中使用 ACCEPT_SOURCE 的冲突解决策略合并并关闭指定的拉取请求。

```
aws codecommit merge-pull-request-by-squash \
    --pull-request-id 47 \
    --source-commit-id 99132ab0EXAMPLE \
    --repository-name MyDemoRepo \
    --conflict-detail-level LINE_LEVEL \
    --conflict-resolution-strategy ACCEPT_SOURCE \
    --name "Jorge Souza" --email "jorge_souza@example.com" \
    --commit-message "Merging pull request 47 by squash and accepting source in
 merge conflicts"
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
```

```
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.142,
        "description": "Review the latest changes and updates to the global
 variables",
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的合并拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergePullRequestBySquash。

merge-pull-request-by-three-way

以下代码示例演示了如何使用 merge-pull-request-by-three-way。

Amazon CLI

使用三向合并策略合并拉取请求

以下 merge-pull-request-by-three-way 示例在名为 MyDemoRepo 的存储库中使用冲突详情和冲突解决策略的默认选项合并并关闭指定的拉取请求。

```
aws codecommit merge-pull-request-by-three-way \
    --pull-request-id 47 \
    --source-commit-id 99132ab0EXAMPLE \
    --repository-name MyDemoRepo \
    --name "Maria Garcia" \
    --email "maria_garcia@example.com" \
    --commit-message "Merging pull request 47 by three-way with default options"
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.142,
        "description": "Review the latest changes and updates to the global
 variables",
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
```

```
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的合并拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergePullRequestByThreeWay。

override-pull-request-approval-rules

以下代码示例演示了如何使用 override-pull-request-approval-rules。

Amazon CLI

覆盖拉取请求的审批规则要求

以下 override-pull-request-approval-rules 示例覆盖指定拉取请求的审批规则。要改为撤消覆盖，请将 --override-status 参数值设置为 REVOKE。

```
aws codecommit override-pull-request-approval-rules \
    --pull-request-id 34  \
    --revision-id 927df8d8EXAMPLE \
    --override-status OVERRIDE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的在拉取请求上覆盖审批规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 OverridePullRequestApprovalRules。

post-comment-for-compared-commit

以下代码示例演示了如何使用 post-comment-for-compared-commit。

Amazon CLI

创建提交评论

此示例演示了如何在名为 MyDemoRepo 的存储库中的两个提交之间的比较中，在 cl_sample.js 文件的更改上添加评论 "Can you add a test case for this?"。

```
aws codecommit post-comment-for-compared-commit \
    --repository-name MyDemoRepo \
    --before-commit-id 317f8570EXAMPLE \
    --after-commit-id 5d036259EXAMPLE \
    --client-request-token 123Example \
    --content "Can you add a test case for this?" \
    --location filePath=cl_sample.js,filePosition=1232,relativeFileVersion=AFTER
```

输出：

```
{
    "afterBlobId": "1f330709EXAMPLE",
    "afterCommitId": "317f8570EXAMPLE",
    "beforeBlobId": "80906a4cEXAMPLE",
    "beforeCommitId": "6e147360EXAMPLE",
    "comment": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "",
        "commentId": "553b509bEXAMPLE56198325",
        "content": "Can you add a test case for this?",
        "creationDate": 1508369612.203,
        "deleted": false,
        "commentId": "abc123-EXAMPLE",
        "lastModifiedDate": 1508369612.203,
        "callerReactions": [],
        "reactionCounts": []
    },
    "location": {
        "filePath": "cl_sample.js",
        "filePosition": 1232,
        "relativeFileVersion": "AFTER"
    ,
```

```
    "repositoryName": "MyDemoRepo"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PostCommentForComparedCommit。

post-comment-for-pull-request

以下代码示例演示了如何使用 post-comment-for-pull-request。

Amazon CLI

向拉取请求添加评论

以下 post-comment-for-pull-request 示例添加评论“这些内容似乎没有在任何地方使用。要移除它们吗？”，该评论是在名为 MyDemoRepo 的存储库中 ID 为 47 的拉取请求内，针对 ahs_count.py 文件的更改所添加的评论。

```
aws codecommit post-comment-for-pull-request \
    --pull-request-id "47" \
    --repository-name MyDemoRepo \
    --before-commit-id 317f8570EXAMPLE \
    --after-commit-id 5d036259EXAMPLE \
    --client-request-token 123Example \
    --content "These don't appear to be used anywhere. Can we remove them?" \
    --location filePath=ahs_count.py,filePosition=367,relativeFileVersion=AFTER
```

输出：

```
{
     "afterBlobId": "1f330709EXAMPLE",
     "afterCommitId": "5d036259EXAMPLE",
     "beforeBlobId": "80906a4cEXAMPLE",
     "beforeCommitId": "317f8570EXAMPLE",
     "comment": {
            "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
            "clientRequestToken": "123Example",
            "commentId": "abcd1234EXAMPLEb5678efgh",
            "content": "These don't appear to be used anywhere. Can we remove
 them?",
```

```
            "creationDate": 1508369622.123,
            "deleted": false,
            "CommentId": "",
            "lastModifiedDate": 1508369622.123,
            "callerReactions": [],
            "reactionCounts": []
        },
        "location": {
            "filePath": "ahs_count.py",
            "filePosition": 367,
            "relativeFileVersion": "AFTER"
         },
     "repositoryName": "MyDemoRepo",
     "pullRequestId": "47"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PostCommentForPullRequest。

post-comment-reply

以下代码示例演示了如何使用 post-comment-reply。

Amazon CLI

回复提交或拉取请求中的评论

此示例演示如何在系统生成的 ID 为 abcd1234EXAMPLEb5678efgh 的评论中添加回复 "Good catch. I'll remove them."。

```
aws codecommit post-comment-reply \
    --in-reply-to abcd1234EXAMPLEb5678efgh \
    --content "Good catch. I'll remove them." \
    --client-request-token 123Example
```

输出：

```
{
    "comment": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "123Example",
        "commentId": "442b498bEXAMPLE5756813",
```

```
        "content": "Good catch. I'll remove them.",
        "creationDate": 1508369829.136,
        "deleted": false,
        "CommentId": "abcd1234EXAMPLEb5678efgh",
        "lastModifiedDate": 150836912.221,
        "callerReactions": [],
        "reactionCounts": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PostCommentReply。

put-comment-reaction

以下代码示例演示了如何使用 put-comment-reaction。

Amazon CLI

使用表情符号回复对提交的评论

以下 put-comment-reaction 示例使用反应值为 :thumbsup: 的表情符号回复一条 ID 为 abcd1234EXAMPLEb5678efgh 的评论。

```
aws codecommit put-comment-reaction \
    --comment-id abcd1234EXAMPLEb5678efgh \
    --reaction-value :thumbsup:
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的在 Amazon CodeCommit 中评论提交。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutCommentReaction。

put-file

以下代码示例演示了如何使用 put-file。

Amazon CLI

向存储库添加文件

以下 put-file 示例将“ExampleSolution.py”文件添加到名为“MyDemoRepo”的存储库中的“feature-randomizationfeature”分支，该分支最新提交的 ID 为“4c925148EXAMPLE”。

```
aws codecommit put-file \
    --repository-name MyDemoRepo \
    --branch-name feature-randomizationfeature \
    --file-content file://MyDirectory/ExampleSolution.py \
    --file-path /solutions/ExampleSolution.py \
    --parent-commit-id 4c925148EXAMPLE \
    --name "Maria Garcia" \
    --email "maria_garcia@example.com" \
    --commit-message "I added a third randomization routine."
```

输出：

```
{
    "blobId": "2eb4af3bEXAMPLE",
    "commitId": "317f8570EXAMPLE",
    "treeId": "347a3408EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutFile。

put-repository-triggers

以下代码示例演示了如何使用 put-repository-triggers。

Amazon CLI

在存储库中添加或更新触发器

此示例演示如何使用已创建的 JSON 文件（此处名为 MyTriggers.json）更新名为“MyFirstTrigger”和“MySecondTrigger”的触发器，该文件包含名为 MyDemoRepo 的存储库的所有触发器的结构。要了解如何获取现有触发器的 JSON，请参阅 get-repository-triggers 命令。

```
aws codecommit put-repository-triggers \
    --repository-name MyDemoRepo file://MyTriggers.json
```

的内容MyTriggers.json：

```
{
```

```
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "destinationArn": "arn:aws:sns:us-
east-1:80398EXAMPLE:MyCodeCommitTopic",
            "branches": [
                "main",
                "preprod"
            ],
            "name": "MyFirstTrigger",
            "customData": "",
            "events": [
                "all"
            ]
        },
        {
            "destinationArn": "arn:aws:lambda:us-
east-1:111111111111:function:MyCodeCommitPythonFunction",
            "branches": [],
            "name": "MySecondTrigger",
            "customData": "EXAMPLE",
            "events": [
                "all"
            ]
        }
    ]
}
```

输出：

```
{
    "configurationId": "6fa51cd8-35c1-EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRepositoryTriggers。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向现有存储库添加 Amazon 标签

以下 tag-resource 示例使用两个标签对指定存储库进行标记。

```
aws codecommit tag-resource \
    --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo \
    --tags Status=Secret,Team=Saanvi
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的向存储库添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

test-repository-triggers

以下代码示例演示了如何使用 test-repository-triggers。

Amazon CLI

在存储库中测试触发器

此示例演示了如何在名为 MyDemoRepo 的 Amazon CodeCommit 存储库中测试名为“MyFirstTrigger”的触发器。在此示例中，存储库中的事件会触发 Amazon Simple Notification Service（Amazon SNS）主题通知。

命令:

```
aws codecommit test-repository-triggers --repository-name MyDemoRepo
 --triggers name=MyFirstTrigger,destinationArn=arn:aws:sns:us-
east-1:111111111111:MyCodeCommitTopic,branches=mainline,preprod,events=all
```

输出：

```
{
  "successfulExecutions": [
      "MyFirstTrigger"
  ],
  "failedExecutions": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestRepositoryTriggers。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除存储库的 Amazon 标签

以下 untag-resource 示例移除名为 MyDemoRepo 的存储库中带有指定键的标签。

```
aws codecommit untag-resource \
    --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo \
    --tag-keys Status
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的移除存储库的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-approval-rule-template-content

以下代码示例演示了如何使用 update-approval-rule-template-content。

Amazon CLI

更新审批规则模板的内容

以下 update-approval-rule-template-content 示例更改指定审批规则模板的内容，重新定义 CodeCommitReview 角色用户的审批池。

```
aws codecommit update-approval-rule-template-content \
    --approval-rule-template-name 1-approver-rule  \
    --new-rule-content "{\"Version\": \"2018-11-08\",\"DestinationReferences
\": [\"refs/heads/main\"],\"Statements\": [{\"Type\": \"Approvers
\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

输出：

```
{
    "approvalRuleTemplate": {
        "creationDate": 1571352720.773,
```

```
        "approvalRuleTemplateDescription": "Requires 1 approval for all pull
 requests from the CodeCommitReview pool",
        "lastModifiedDate": 1571358728.41,
        "approvalRuleTemplateId": "41de97b7-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
        "ruleContentSha256": "2f6c21a5EXAMPLE",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApprovalRuleTemplateContent。

update-approval-rule-template-description

以下代码示例演示了如何使用 update-approval-rule-template-description。

Amazon CLI

更新审批规则模板的描述

以下 update-approval-rule-template-description 示例将指定的审批规则模板更改为 Requires 1 approval for all pull requests from the CodeCommitReview pool：

```
aws codecommit update-approval-rule-template-description \
    --approval-rule-template-name 1-approver-rule-for-all-pull-requests  \
    --approval-rule-template-description "Requires 1 approval for all pull requests
 from the CodeCommitReview pool"
```

输出：

```
{
  "approvalRuleTemplate": {
    "creationDate": 1571352720.773,
    "approvalRuleTemplateDescription": "Requires 1 approval for all pull requests
 from the CodeCommitReview pool",
```

```
    "lastModifiedDate": 1571358728.41,
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "ruleContentSha256": "2f6c21a5EXAMPLE",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
  }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApprovalRuleTemplateDescription。

update-approval-rule-template-name

以下代码示例演示了如何使用 update-approval-rule-template-name。

Amazon CLI

更新审批规则模板的名称

以下 update-approval-rule-template-name 示例将审批规则模板的名称从 1-approver-rule 更改为 1-approver-rule-for-all-pull-requests`。

```
aws codecommit update-approval-rule-template-name \
    --old-approval-rule-template-name 1-approver-rule  \
    --new-approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

输出：

```
{
  "approvalRuleTemplate": {
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "lastModifiedDate": 1571358241.619,
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "creationDate": 1571352720.773,
```

```
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
    "approvalRuleTemplateDescription": "All pull requests must be approved by one
 developer on the team.",
    "ruleContentSha256": "2f6c21a5cEXAMPLE"
  }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的管理审批规则模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApprovalRuleTemplateName。

update-comment

以下代码示例演示了如何使用 update-comment。

Amazon CLI

更新提交评论

此示例演示如何将内容 "Fixed as requested. I'll update the pull request." 添加到 ID 为 442b498bEXAMPLE5756813 的评论中。

```
aws codecommit update-comment \
    --comment-id 442b498bEXAMPLE5756813 \
    --content "Fixed as requested. I'll update the pull request."
```

输出：

```
{
    "comment": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "",
        "commentId": "442b498bEXAMPLE5756813",
        "content": "Fixed as requested. I'll update the pull request.",
        "creationDate": 1508369929.783,
        "deleted": false,
        "lastModifiedDate": 1508369929.287,
        "callerReactions": [],
        "reactionCounts":
            {
                "THUMBSUP" : 2
```

```
            }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateComment。

update-default-branch

以下代码示例演示了如何使用 update-default-branch。

Amazon CLI

更改存储库的默认分支

此示例更改 Amazon CodeCommit 存储库的默认分支。该命令只在出现错误时生成输出。

命令:

```
aws codecommit update-default-branch --repository-name MyDemoRepo --default-branch-name MyNewBranch
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDefaultBranch。

update-pull-request-approval-rule-content

以下代码示例演示了如何使用 update-pull-request-approval-rule-content。

Amazon CLI

编辑拉取请求的审批规则

以下 update-pull-request-approval-rule-content 示例更新她指定的审批规则，要求从包括 123456789012 Amazon 账户中任何 IAM 用户的审批池中获得一个用户审批。

```
aws codecommit update-pull-request-approval-rule-content \
    --pull-request-id 27  \
```

```
    --approval-rule-name "Require two approved approvers" \
    --approval-rule-content "{Version: 2018-11-08, Statements: [{Type:
 \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
[\"CodeCommitApprovers:123456789012:user/*\"]}]}"
```

输出：

```
{
    "approvalRule": {
        "approvalRuleContent": "{Version: 2018-11-08, Statements:
 [{Type: \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
[\"CodeCommitApprovers:123456789012:user/*\"]}]}",
        "approvalRuleId": "aac33506-EXAMPLE",
        "originApprovalRuleTemplate": {},
        "creationDate": 1570752871.932,
        "lastModifiedDate": 1570754058.333,
        "approvalRuleName": Require two approved approvers",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "ruleContentSha256": "cd93921cEXAMPLE",
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的编辑或删除审批规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePullRequestApprovalRuleContent。

update-pull-request-approval-state

以下代码示例演示了如何使用 update-pull-request-approval-state。

Amazon CLI

批准或撤消对拉取请求的批准

以下 update-pull-request-approval-state 示例批准 ID 为 27、修订 ID 为 9f29d167EXAMPLE 的拉取请求。如果您想撤消批准，请将 --approval-state 参数值设置为 REVOKE。

```
aws codecommit update-pull-request-approval-state \
    --pull-request-id 27  \
    --revision-id 9f29d167EXAMPLE  \
```

```
    --approval-state "APPROVE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的审查拉取请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePullRequestApprovalState。

update-pull-request-description

以下代码示例演示了如何使用 update-pull-request-description。

Amazon CLI

更改拉取请求的说明

此示例演示如何更改 ID 为 47 的拉取请求的说明。

```
aws codecommit update-pull-request-description \
    --pull-request-id 47 \
    --description "Updated the pull request to remove unused global variable."
```

输出：

```
{
    "pullRequest": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.155,
        "description": "Updated the pull request to remove unused global variable.",
        "lastActivityDate": 1508372423.204,
        "pullRequestId": "47",
        "pullRequestStatus": "OPEN",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false,
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
```

```
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePullRequestDescription。

update-pull-request-status

以下代码示例演示了如何使用 update-pull-request-status。

Amazon CLI

更改拉取请求的状态

此示例演示如何在名为 MyDemoRepo 的 Amazon CodeCommit 存储库中将 ID 为 42 的拉取请求的状态更改为 CLOSED 的状态。

```
aws codecommit update-pull-request-status \
    --pull-request-id 42 \
    --pull-request-status CLOSED
```

输出：

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approvers-needed-for-this-change",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
```

```
        "clientRequestToken": "",
        "creationDate": 1508530823.165,
        "description": "Updated the pull request to remove unused global variable.",
        "lastActivityDate": 1508372423.12,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false,
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePullRequestStatus。

update-pull-request-title

以下代码示例演示了如何使用 update-pull-request-title。

Amazon CLI

更改拉取请求的标题

此示例演示如何更改 ID 为 47 的拉取请求的标题。

```
aws codecommit update-pull-request-title \
    --pull-request-id 47 \
    --title "Consolidation of global variables - updated review"
```

输出：

```
{
    "pullRequest": {
```

```
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b26gr-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.12,
        "description": "Review the latest changes and updates to the global
 variables. I have updated this request with some changes, including removing some
 unused variables.",
        "lastActivityDate": 1508372657.188,
        "pullRequestId": "47",
        "pullRequestStatus": "OPEN",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false,
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables - updated review"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePullRequestTitle。

update-repository-description

以下代码示例演示了如何使用 update-repository-description。

Amazon CLI

更改存储库的说明

此示例更改 Amazon CodeCommit 存储库的说明。该命令只在出现错误时生成输出。

命令:

```
aws codecommit update-repository-description --repository-name MyDemoRepo --
repository-description "This description was changed"
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRepositoryDescription。

update-repository-name

以下代码示例演示了如何使用 update-repository-name。

Amazon CLI

更改存储库的名称

此示例更改 Amazon CodeCommit 存储库的名称。该命令只在出现错误时生成输出。更改 Amazon CodeCommit 存储库的名称将更改用户连接到存储库所需的 SSH 和 HTTPS URL。在更新连接设置之前，用户无法连接到此存储库。此外，由于存储库的 ARN 会发生更改，更改存储库名称会使依赖该存储库 ARN 的所有 IAM 用户策略失效。

命令:

```
aws codecommit update-repository-name --old-name MyDemoRepo --new-
name MyRenamedDemoRepo
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRepositoryName。

使用 Amazon CLI 的 CodeDeploy 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeDeploy 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-on-premises-instances`

以下代码示例演示了如何使用 `add-tags-to-on-premises-instances`。

Amazon CLI

向本地实例添加标签

以下 `add-tags-to-on-premises-instances` 示例在 Amazon CodeDeploy 中将同一个本地实例标签关联到两个本地实例。它不会向 Amazon CodeDeploy 注册本地实例。

```
aws deploy add-tags-to-on-premises-instances \
    --instance-names AssetTag12010298EX AssetTag23121309EX \
    --tags Key=Name,Value=CodeDeployDemo-OnPrem
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToOnPremisesInstances。

`batch-get-application-revisions`

以下代码示例演示了如何使用 `batch-get-application-revisions`。

Amazon CLI

检索有关应用程序修订的信息

以下 batch-get-application-revisions 示例检索有关存储在 GitHub 存储库中的指定修订的信息。

```
aws deploy batch-get-application-revisions \
    --application-name my-codedeploy-application \
    --revisions "[{\"gitHubLocation\": {\"commitId\":
 \"fa85936EXAMPLEa31736c051f10d77297EXAMPLE\",\"repository\": \"my-github-token/my-
repository\"},\"revisionType\": \"GitHub\"}]"
```

输出：

```
{
    "revisions": [
        {
            "genericRevisionInfo": {
                "description": "Application revision registered by Deployment ID: d-
A1B2C3111",
                "lastUsedTime": 1556912355.884,
                "registerTime": 1556912355.884,
                "firstUsedTime": 1556912355.884,
                "deploymentGroups": []
            },
            "revisionLocation": {
                "revisionType": "GitHub",
                "gitHubLocation": {
                    "commitId": "fa85936EXAMPLEa31736c051f10d77297EXAMPLE",
                    "repository": "my-github-token/my-repository"
                }
            }
        }
    ],
    "applicationName": "my-codedeploy-application",
    "errorMessage": ""
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 BatchGetApplicationRevisions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetApplicationRevisions。

batch-get-applications

以下代码示例演示了如何使用 batch-get-applications。

Amazon CLI

获取有关多个应用程序的信息

以下 batch-get-applications 示例显示与用户的 Amazon 账户关联的多个应用程序的相关信息。

```
aws deploy batch-get-applications --application-names WordPress_App MyOther_App
```

输出：

```
{
    "applicationsInfo": [
        {
            "applicationName": "WordPress_App",
            "applicationId": "d9dd6993-f171-44fa-a811-211e4EXAMPLE",
            "createTime": 1407878168.078,
            "linkedToGitHub": false
        },
        {
            "applicationName": "MyOther_App",
            "applicationId": "8ca57519-31da-42b2-9194-8bb16EXAMPLE",
            "createTime": 1407453571.63,
            "linkedToGitHub": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetApplications。

batch-get-deployment-groups

以下代码示例演示了如何使用 batch-get-deployment-groups。

Amazon CLI

获取有关一个或多个部署组的信息

以下 batch-get-deployment-groups 示例检索与指定的 CodeDeploy 应用程序关联的两个部署组的相关信息。

```
aws deploy batch-get-deployment-groups \
    --application-name my-codedeploy-application \
    --deployment-group-names "[\"my-deployment-group-1\",\"my-deployment-group-2\"]"
```

输出：

```
{
    "deploymentGroupsInfo": [
        {
            "deploymentStyle": {
                "deploymentOption": "WITHOUT_TRAFFIC_CONTROL",
                "deploymentType": "IN_PLACE"
            },
            "autoRollbackConfiguration": {
                "enabled": false
            },
            "onPremisesTagSet": {
                "onPremisesTagSetList": []
            },
            "serviceRoleArn": "arn:aws:iam::123456789012:role/
CodeDeployServiceRole",
            "lastAttemptedDeployment": {
                "endTime": 1556912366.415,
                "status": "Failed",
                "createTime": 1556912355.884,
                "deploymentId": "d-A1B2C3111"
            },
            "autoScalingGroups": [],
            "deploymentGroupName": "my-deployment-group-1",
            "ec2TagSet": {
                "ec2TagSetList": [
                    [
                        {
                            "Type": "KEY_AND_VALUE",
                            "Value": "my-EC2-instance",
                            "Key": "Name"
                        }
                    ]
                ]
            },
```

```
            "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111example",
            "triggerConfigurations": [],
            "applicationName": "my-codedeploy-application",
            "computePlatform": "Server",
            "deploymentConfigName": "CodeDeployDefault.AllAtOnce"
        },
        {
            "deploymentStyle": {
                "deploymentOption": "WITHOUT_TRAFFIC_CONTROL",
                "deploymentType": "IN_PLACE"
            },
            "autoRollbackConfiguration": {
                "enabled": false
            },
            "onPremisesTagSet": {
                "onPremisesTagSetList": []
            },
            "serviceRoleArn": "arn:aws:iam::123456789012:role/
CodeDeployServiceRole",
            "autoScalingGroups": [],
            "deploymentGroupName": "my-deployment-group-2",
            "ec2TagSet": {
                "ec2TagSetList": [
                    [
                        {
                            "Type": "KEY_AND_VALUE",
                            "Value": "my-EC2-instance",
                            "Key": "Name"
                            }
                    ]
                ]
            },
            "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-22222example",
            "triggerConfigurations": [],
            "applicationName": "my-codedeploy-application",
            "computePlatform": "Server",
            "deploymentConfigName": "CodeDeployDefault.AllAtOnce"
        }
    ],
    "errorMessage": ""
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 BatchGetDeploymentGroups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetDeploymentGroups。

batch-get-deployment-targets

以下代码示例演示了如何使用 batch-get-deployment-targets。

Amazon CLI

检索与部署关联的目标

以下 batch-get-deployment-targets 示例返回与指定部署关联的其中一个目标的相关信息。

```
aws deploy batch-get-deployment-targets \
    --deployment-id "d-1A2B3C4D5" \
    --target-ids "i-01a2b3c4d5e6f1111"
```

输出：

```
{
    "deploymentTargets": [
        {
            "deploymentTargetType": "InstanceTarget",
            "instanceTarget": {
                "lifecycleEvents": [
                    {
                        "startTime": 1556918592.162,
                        "lifecycleEventName": "ApplicationStop",
                        "status": "Succeeded",
                        "endTime": 1556918592.247,
                        "diagnostics": {
                            "scriptName": "",
                            "errorCode": "Success",
                            "logTail": "",
                            "message": "Succeeded"
                        }
                    },
                    {
                        "startTime": 1556918593.193,
                        "lifecycleEventName": "DownloadBundle",
                        "status": "Succeeded",
                        "endTime": 1556918593.981,
                        "diagnostics": {
```

```
                            "scriptName": "",
                            "errorCode": "Success",
                            "logTail": "",
                            "message": "Succeeded"
                        }
                    },
                    {
                        "startTime": 1556918594.805,
                        "lifecycleEventName": "BeforeInstall",
                        "status": "Succeeded",
                        "endTime": 1556918681.807,
                        "diagnostics": {
                            "scriptName": "",
                            "errorCode": "Success",
                            "logTail": "",
                            "message": "Succeeded"
                        }
                    }
                ],
                "targetArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-01a2b3c4d5e6f1111",
                "deploymentId": "d-1A2B3C4D5",
                "lastUpdatedAt": 1556918687.504,
                "targetId": "i-01a2b3c4d5e6f1111",
                "status": "Succeeded"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 BatchGetDeploymentTargets。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetDeploymentTargets。

batch-get-deployments

以下代码示例演示了如何使用 batch-get-deployments。

Amazon CLI

获取有关多个部署的信息

以下 batch-get-deployments 示例显示与用户的 Amazon 账户关联的多个部署的相关信息。

```
aws deploy batch-get-deployments --deployment-ids d-A1B2C3111 d-A1B2C3222
```

输出：

```
{
    "deploymentsInfo": [
        {
            "applicationName": "WordPress_App",
            "status": "Failed",
            "deploymentOverview": {
                "Failed": 0,
                "InProgress": 0,
                "Skipped": 0,
                "Succeeded": 1,
                "Pending": 0
            },
            "deploymentConfigName": "CodeDeployDefault.OneAtATime",
            "creator": "user",
            "deploymentGroupName": "WordPress_DG",
            "revision": {
                "revisionType": "S3",
                "s3Location": {
                "bundleType": "zip",
                "version": "uTecLusEXAMPLEFXtfUcyfV8bEXAMPLE",
                "bucket": "amzn-s3-demo-bucket",
                "key": "WordPressApp.zip"
                }
            },
            "deploymentId": "d-A1B2C3111",
            "createTime": 1408480721.9,
            "completeTime": 1408480741.822
        },
        {
            "applicationName": "MyOther_App",
            "status": "Failed",
            "deploymentOverview": {
                "Failed": 1,
                "InProgress": 0,
                "Skipped": 0,
                "Succeeded": 0,
                "Pending": 0
            },
            "deploymentConfigName": "CodeDeployDefault.OneAtATime",
```

```
            "creator": "user",
            "errorInformation": {
                "message": "Deployment failed: Constraint default violated: No hosts
 succeeded.",
                "code": "HEALTH_CONSTRAINTS"
            },
            "deploymentGroupName": "MyOther_DG",
            "revision": {
                "revisionType": "S3",
                "s3Location": {
                "bundleType": "zip",
                "eTag": "\"dd56cfdEXAMPLE8e768f9d77fEXAMPLE\"",
                "bucket": "amzn-s3-demo-bucket",
                "key": "MyOtherApp.zip"
                }
            },
            "deploymentId": "d-A1B2C3222",
            "createTime": 1409764576.589,
            "completeTime": 1409764596.101
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetDeployments。

batch-get-on-premises-instances

以下代码示例演示了如何使用 batch-get-on-premises-instances。

Amazon CLI

获取有关一个或多个本地实例的信息

以下 batch-get-on-premises-instances 示例获取有关两个本地实例的信息。

```
aws deploy batch-get-on-premises-instances --instance-
names AssetTag12010298EX AssetTag23121309EX
```

输出：

```
{
    "instanceInfos": [
```

```
        {
            "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag12010298EX",
            "tags": [
                {
                    "Value": "CodeDeployDemo-OnPrem",
                    "Key": "Name"
                }
            ],
            "instanceName": "AssetTag12010298EX",
            "registerTime": 1425579465.228,
            "instanceArn": "arn:aws:codedeploy:us-west-2:123456789012:instance/
AssetTag12010298EX_4IwLNI2Alh"
        },
        {
            "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag23121309EX",
            "tags": [
                {
                    "Value": "CodeDeployDemo-OnPrem",
                    "Key": "Name"
                }
            ],
            "instanceName": "AssetTag23121309EX",
            "registerTime": 1425595585.988,
            "instanceArn": "arn:aws:codedeploy:us-west-2:80398EXAMPLE:instance/
AssetTag23121309EX_PomUy64Was"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetOnPremisesInstances。

continue-deployment

以下代码示例演示了如何使用 continue-deployment。

Amazon CLI

开始重新路由流量时，无需等待指定的等待时间。

以下 continue-deployment 示例开始将原始环境中实例的流量（已准备好开始转移）重新路由到替换环境中的实例。

```
aws deploy continue-deployment \
    --deployment-id "d-A1B2C3111" \
    --deployment-wait-type "READY_WAIT"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 ContinueDeployment。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ContinueDeployment。

create-application

以下代码示例演示了如何使用 create-application。

Amazon CLI

创建应用程序

以下 create-application 示例创建一个应用程序并将其与用户的 Amazon 账户关联。

```
aws deploy create-application --application-name MyOther_App
```

输出：

```
{
    "applicationId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApplication。

create-deployment-config

以下代码示例演示了如何使用 create-deployment-config。

Amazon CLI

创建自定义部署配置

以下 create-deployment-config 示例创建自定义部署配置并将其与用户的 Amazon 账户关联。

```
aws deploy create-deployment-config \
```

```
    --deployment-config-name ThreeQuartersHealthy \
    --minimum-healthy-hosts type=FLEET_PERCENT,value=75
```

输出：

```
{
    "deploymentConfigId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeploymentConfig。

create-deployment-group

以下代码示例演示了如何使用 create-deployment-group。

Amazon CLI

创建部署组

以下 create-deployment-group 示例创建一个部署组并将其与指定应用程序和用户的 Amazon 账户关联。

```
aws deploy create-deployment-group \
    --application-name WordPress_App \
    --auto-scaling-groups CodeDeployDemo-ASG \
    --deployment-config-name CodeDeployDefault.OneAtATime \
    --deployment-group-name WordPress_DG \
    --ec2-tag-filters Key=Name,Value=CodeDeployDemo,Type=KEY_AND_VALUE \
    --service-role-arn arn:aws:iam::123456789012:role/CodeDeployDemoRole
```

输出：

```
{
    "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeploymentGroup。

create-deployment

以下代码示例演示了如何使用 create-deployment。

Amazon CLI

示例 1：使用 EC2/本地计算平台创建 CodeDeploy 部署

以下 create-deployment 示例创建一个部署并将其与用户的 Amazon 账户关联。

```
aws deploy create-deployment \
    --application-name WordPress_App \
    --deployment-config-name CodeDeployDefault.OneAtATime \
    --deployment-group-name WordPress_DG \
    --description "My demo deployment" \
    --s3-location bucket=amzn-s3-demo-
bucket,bundleType=zip,eTag=dd56cfdEXAMPLE8e768f9d77fEXAMPLE,key=WordPressApp.zip
```

输出：

```
{
    "deploymentId": "d-A1B2C3111"
}
```

示例 2：使用 Amazon ECS 计算平台创建 CodeDeploy 部署

以下 create-deployment 示例使用以下两个文件来部署 Amazon ECS 服务。

create-deployment.json 文件的内容：

```
{
    "applicationName": "ecs-deployment",
    "deploymentGroupName": "ecs-deployment-dg",
    "revision": {
        "revisionType": "S3",
        "s3Location": {
            "bucket": "ecs-deployment-bucket",
            "key": "appspec.yaml",
            "bundleType": "YAML"
        }
    }
}
```

反过来，该文件会从名为 ecs-deployment-bucket 的 S3 存储桶中检索以下文件 appspec.yaml。

```
version: 0.0
```

```
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "arn:aws:ecs:region:123456789012:task-definition/ecs-task-
def:2"
        LoadBalancerInfo:
          ContainerName: "sample-app"
          ContainerPort: 80
        PlatformVersion: "LATEST"
```

命令:

```
aws deploy create-deployment \
    --cli-input-json file://create-deployment.json \
    --region us-east-1
```

输出：

```
{
    "deploymentId": "d-1234ABCDE"
}
```

有关更多信息，请参阅《Amazon CodeDeploy API Reference》中的 CreateDeployment。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeployment。

delete-application

以下代码示例演示了如何使用 delete-application。

Amazon CLI

删除 应用程序

以下 delete-application 示例删除与用户的 Amazon 账户关联的指定应用程序。

```
aws deploy delete-application --application-name WordPress_App
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApplication。

delete-deployment-config

以下代码示例演示了如何使用 delete-deployment-config。

Amazon CLI

删除部署配置

以下 delete-deployment-config 示例删除与用户的 Amazon 账户关联的自定义部署配置。

```
aws deploy delete-deployment-config --deployment-config-name ThreeQuartersHealthy
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeploymentConfig。

delete-deployment-group

以下代码示例演示了如何使用 delete-deployment-group。

Amazon CLI

删除部署组

以下 delete-deployment-group 示例删除与指定应用程序关联的部署组。

```
aws deploy delete-deployment-group \
    --application-name WordPress_App \
    --deployment-group-name WordPress_DG
```

输出：

```
{
    "hooksNotCleanedUp": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeploymentGroup。

delete-git-hub-account-token

以下代码示例演示了如何使用 delete-git-hub-account-token。

Amazon CLI

删除 GitHub 账户连接

以下 `delete-git-hub-account-token` 示例删除指定 GitHub 账户的连接。

```
aws deploy delete-git-hub-account-token --token-name my-github-account
```

输出：

```
{
    "tokenName": "my-github-account"
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 DeleteGitHubAccountToken。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGitHubAccountToken。

`deregister-on-premises-instance`

以下代码示例演示了如何使用 `deregister-on-premises-instance`。

Amazon CLI

注销本地实例

以下 `deregister-on-premises-instance` 示例使用 Amazon CodeDeploy 注销本地实例，但不会删除与该实例关联的 IAM 用户，也不会在 Amazon CodeDeploy 中解除本地实例标签与实例的关联。它还不会从实例中卸载 Amazon CodeDeploy 代理程序，也不会从实例中移除本地配置文件。

```
aws deploy deregister-on-premises-instance --instance-name AssetTag12010298EX
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterOnPremisesInstance。

`deregister`

以下代码示例演示了如何使用 `deregister`。

Amazon CLI

注销本地实例

以下 deregister 示例使用 Amazon CodeDeploy 注销本地实例。此操作不会删除与实例关联的 IAM 用户。它会在 Amazon CodeDeploy 中解除本地标签与实例的关联。它不会从实例中卸载 Amazon CodeDeploy 代理程序，也不会从实例中移除本地配置文件。

```
aws deploy deregister \
    --instance-name AssetTag12010298EX \
    --no-delete-iam-user \
    --region us-west-2
```

输出：

```
Retrieving on-premises instance information... DONE
IamUserArn: arn:aws:iam::80398EXAMPLE:user/AWS/CodeDeploy/AssetTag12010298EX
Tags: Key=Name,Value=CodeDeployDemo-OnPrem
Removing tags from the on-premises instance... DONE
Deregistering the on-premises instance... DONE
Run the following command on the on-premises instance to uninstall the codedeploy-
agent:
aws deploy uninstall
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Deregister。

get-application-revision

以下代码示例演示了如何使用 get-application-revision。

Amazon CLI

获取有关应用程序修订的信息

以下 get-application-revision 示例显示与指定应用程序关联的应用程序修订的相关信息。

```
aws deploy get-application-revision \
    --application-name WordPress_App \
    --s3-location bucket=amzn-s3-demo-
bucket,bundleType=zip,eTag=dd56cfdEXAMPLE8e768f9d77fEXAMPLE,key=WordPressApp.zip
```

输出：

```
{
    "applicationName": "WordPress_App",
    "revisionInfo": {
        "description": "Application revision registered by Deployment ID: d-
A1B2C3111",
        "registerTime": 1411076520.009,
        "deploymentGroups": "WordPress_DG",
        "lastUsedTime": 1411076520.009,
        "firstUsedTime": 1411076520.009
    },
    "revision": {
        "revisionType": "S3",
        "s3Location": {
            "bundleType": "zip",
            "eTag": "dd56cfdEXAMPLE8e768f9d77fEXAMPLE",
            "bucket": "amzn-s3-demo-bucket",
            "key": "WordPressApp.zip"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApplicationRevision。

get-application

以下代码示例演示了如何使用 get-application。

Amazon CLI

获取有关应用程序的信息

以下 get-application 示例显示与用户的 Amazon 账户关联的应用程序的相关信息。

```
aws deploy get-application --application-name WordPress_App
```

输出：

```
{
    "application": {
        "applicationName": "WordPress_App",
        "applicationId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "createTime": 1407878168.078,
```

```
        "linkedToGitHub": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApplication。

get-deployment-config

以下代码示例演示了如何使用 get-deployment-config。

Amazon CLI

获取有关部署配置的信息

以下 get-deployment-config 示例显示与用户的 Amazon 账户关联的部署配置的相关信息。

```
aws deploy get-deployment-config --deployment-config-name ThreeQuartersHealthy
```

输出：

```
{
    "deploymentConfigInfo": {
        "deploymentConfigId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "minimumHealthyHosts": {
            "type": "FLEET_PERCENT",
            "value": 75
        },
        "createTime": 1411081164.379,
        "deploymentConfigName": "ThreeQuartersHealthy"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentConfig。

get-deployment-group

以下代码示例演示了如何使用 get-deployment-group。

Amazon CLI

查看有关部署组的信息

以下 get-deployment-group 示例显示与指定应用程序关联的部署组的相关信息。

```
aws deploy get-deployment-group \
    --application-name WordPress_App \
    --deployment-group-name WordPress_DG
```

输出：

```
{
    "deploymentGroupInfo": {
        "applicationName": "WordPress_App",
        "autoScalingGroups": [
            "CodeDeployDemo-ASG"
        ],
        "deploymentConfigName": "CodeDeployDefault.OneAtATime",
        "ec2TagFilters": [
            {
                "Type": "KEY_AND_VALUE",
                "Value": "CodeDeployDemo",
                "Key": "Name"
            }
        ],
        "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "serviceRoleArn": "arn:aws:iam::123456789012:role/CodeDeployDemoRole",
        "deploymentGroupName": "WordPress_DG"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentGroup。

get-deployment-instance

以下代码示例演示了如何使用 get-deployment-instance。

Amazon CLI

获取有关部署实例的信息

以下 get-deployment-instance 示例显示与指定部署关联的部署实例的相关信息。

```
aws deploy get-deployment-instance --deployment-id d-QA4G4F9EX --instance-
id i-902e9fEX
```

输出：


```
{
    "instanceSummary": {
        "instanceId": "arn:aws:ec2:us-east-1:80398EXAMPLE:instance/i-902e9fEX",
        "lifecycleEvents": [
            {
                "status": "Succeeded",
                "endTime": 1408480726.569,
                "startTime": 1408480726.437,
                "lifecycleEventName": "ApplicationStop"
            },
            {
                "status": "Succeeded",
                "endTime": 1408480728.016,
                "startTime": 1408480727.665,
                "lifecycleEventName": "DownloadBundle"
            },
            {
                "status": "Succeeded",
                "endTime": 1408480729.744,
                "startTime": 1408480729.125,
                "lifecycleEventName": "BeforeInstall"
            },
            {
                "status": "Succeeded",
                "endTime": 1408480730.979,
                "startTime": 1408480730.844,
                "lifecycleEventName": "Install"
            },
            {
                "status": "Failed",
                "endTime": 1408480732.603,
                "startTime": 1408480732.1,
                "lifecycleEventName": "AfterInstall"
            },
            {
                "status": "Skipped",
                "endTime": 1408480732.606,
                "lifecycleEventName": "ApplicationStart"
            },
            {
                "status": "Skipped",
                "endTime": 1408480732.606,
```

```
                "lifecycleEventName": "ValidateService"
            }
        ],
        "deploymentId": "d-QA4G4F9EX",
        "lastUpdatedAt": 1408480733.152,
        "status": "Failed"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentInstance。

get-deployment-target

以下代码示例演示了如何使用 get-deployment-target。

Amazon CLI

返回有关部署目标的信息

以下 get-deployment-target 示例返回与指定部署关联的部署目标的相关信息。

```
aws deploy get-deployment-target \
    --deployment-id "d-A1B2C3111" \
    --target-id "i-a1b2c3d4e5f611111"
```

输出：

```
{
    "deploymentTarget": {
        "deploymentTargetType": "InstanceTarget",
        "instanceTarget": {
            "lastUpdatedAt": 1556918687.504,
            "targetId": "i-a1b2c3d4e5f611111",
            "targetArn": "arn:aws:ec2:us-west-2:123456789012:instance/i-
a1b2c3d4e5f611111",
            "status": "Succeeded",
            "lifecycleEvents": [
                {
                    "status": "Succeeded",
                    "diagnostics": {
                        "errorCode": "Success",
                        "message": "Succeeded",
                        "logTail": "",
```

```
                "scriptName": ""
            },
            "lifecycleEventName": "ApplicationStop",
            "startTime": 1556918592.162,
            "endTime": 1556918592.247
        },
        {
            "status": "Succeeded",
            "diagnostics": {
                "errorCode": "Success",
                "message": "Succeeded",
                "logTail": "",
                "scriptName": ""
            },
            "lifecycleEventName": "DownloadBundle",
            "startTime": 1556918593.193,
            "endTime": 1556918593.981
        },
        {
            "status": "Succeeded",
            "diagnostics": {
                "errorCode": "Success",
                "message": "Succeeded",
                "logTail": "",
                "scriptName": ""
            },
            "lifecycleEventName": "BeforeInstall",
            "startTime": 1556918594.805,
            "endTime": 1556918681.807
        },
        {
            "status": "Succeeded",
            "diagnostics": {
                "errorCode": "Success",
                "message": "Succeeded",
                "logTail": "",
                "scriptName": ""
            },
            "lifecycleEventName": "Install",
            "startTime": 1556918682.696,
            "endTime": 1556918683.005
        },
        {
            "status": "Succeeded",
```

```
                    "diagnostics": {
                        "errorCode": "Success",
                        "message": "Succeeded",
                        "logTail": "",
                        "scriptName": ""
                    },
                    "lifecycleEventName": "AfterInstall",
                    "startTime": 1556918684.135,
                    "endTime": 1556918684.216
                },
                {
                    "status": "Succeeded",
                    "diagnostics": {
                        "errorCode": "Success",
                        "message": "Succeeded",
                        "logTail": "",
                        "scriptName": ""
                    },
                    "lifecycleEventName": "ApplicationStart",
                    "startTime": 1556918685.211,
                    "endTime": 1556918685.295
                },
                {
                    "status": "Succeeded",
                    "diagnostics": {
                        "errorCode": "Success",
                        "message": "Succeeded",
                        "logTail": "",
                        "scriptName": ""
                    },
                    "lifecycleEventName": "ValidateService",
                    "startTime": 1556918686.65,
                    "endTime": 1556918686.747
                }
            ],
            "deploymentId": "d-A1B2C3111"
        }
    }
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 GetDeploymentTarget。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentTarget。

get-deployment

以下代码示例演示了如何使用 get-deployment。

Amazon CLI

获取有关部署的信息

以下 get-deployment 示例显示与用户的 Amazon 账户关联的部署的相关信息。

```
aws deploy get-deployment --deployment-id d-A1B2C3123
```

输出：

```
{
    "deploymentInfo": {
        "applicationName": "WordPress_App",
        "status": "Succeeded",
        "deploymentOverview": {
            "Failed": 0,
            "InProgress": 0,
            "Skipped": 0,
            "Succeeded": 1,
            "Pending": 0
        },
        "deploymentConfigName": "CodeDeployDefault.OneAtATime",
        "creator": "user",
        "description": "My WordPress app deployment",
        "revision": {
            "revisionType": "S3",
            "s3Location":  {
            "bundleType": "zip",
            "eTag": "\"dd56cfdEXAMPLE8e768f9d77fEXAMPLE\"",
            "bucket": "amzn-s3-demo-bucket",
            "key": "WordPressApp.zip"
            }
        },
        "deploymentId": "d-A1B2C3123",
        "deploymentGroupName": "WordPress_DG",
        "createTime": 1409764576.589,
        "completeTime": 1409764596.101,
        "ignoreApplicationStopFailures": false
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployment。

get-on-premises-instance

以下代码示例演示了如何使用 get-on-premises-instance。

Amazon CLI

获取有关本地实例的信息

以下 get-on-premises-instance 示例检索有关指定本地实例的信息。

```
aws deploy get-on-premises-instance --instance-name AssetTag12010298EX
```

输出：

```
{
    "instanceInfo": {
    "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag12010298EX",
        "tags": [
        {
            "Value": "CodeDeployDemo-OnPrem",
            "Key": "Name"
        }
        ],
        "instanceName": "AssetTag12010298EX",
        "registerTime": 1425579465.228,
        "instanceArn": "arn:aws:codedeploy:us-east-1:123456789012:instance/
AssetTag12010298EX_4IwLNI2Alh"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOnPremisesInstance。

install

以下代码示例演示了如何使用 install。

Amazon CLI

安装本地实例

以下 `install` 示例将本地配置文件从实例上的指定位置复制到 Amazon CodeDeploy 代理程序预期在实例上找到它的位置。它还会在实例上安装 Amazon CodeDeploy 代理程序。它不会创建任何 IAM 用户，不会向 Amazon CodeDeploy 注册本地实例，也不会在 Amazon CodeDeploy 中为该实例关联任何本地实例标签。

```
aws deploy install \
    --override-config \
    --config-file C:\temp\codedeploy.onpremises.yml \
    --region us-west-2 \
    --agent-installer s3://aws-codedeploy-us-west-2/latest/codedeploy-agent.msi
```

输出：

```
Creating the on-premises instance configuration file... DONE
Installing the AWS CodeDeploy Agent... DONE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Install。

list-application-revisions

以下代码示例演示了如何使用 `list-application-revisions`。

Amazon CLI

获取有关应用程序修订的信息

以下 `list-application-revisions` 示例显示与指定应用程序关联的所有应用程序修订的相关信息。

```
aws deploy list-application-revisions \
    --application-name WordPress_App \
    --s-3-bucket amzn-s3-demo-bucket \
    --deployed exclude \
    --s-3-key-prefix WordPress_ \
    --sort-by lastUsedTime \
    --sort-order descending
```

输出：

```
{
    "revisions": [
        {
            "revisionType": "S3",
            "s3Location": {
                "version": "uTecLusvCB_JqHFXtfUcyfV8bEXAMPLE",
                "bucket": "amzn-s3-demo-bucket",
                "key": "WordPress_App.zip",
                "bundleType": "zip"
            }
        },
        {
            "revisionType": "S3",
            "s3Location": {
                "version": "tMk.UxgDpMEVb7V187ZM6wVAWEXAMPLE",
                "bucket": "amzn-s3-demo-bucket",
                "key": "WordPress_App_2-0.zip",
                "bundleType": "zip"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListApplicationRevisions。

list-applications

以下代码示例演示了如何使用 list-applications。

Amazon CLI

获取有关应用程序的信息

以下 list-applications 示例显示与用户的 Amazon 账户关联的所有应用程序的相关信息。

```
aws deploy list-applications
```

输出：

```
{
```

```
    "applications": [
        "WordPress_App",
        "MyOther_App"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListApplications。

list-deployment-configs

以下代码示例演示了如何使用 list-deployment-configs。

Amazon CLI

获取有关部署配置的信息

以下 list-deployment-configs 示例显示与用户的 Amazon 账户关联的所有部署配置的相关信息。

```
aws deploy list-deployment-configs
```

输出：

```
{
    "deploymentConfigsList": [
        "ThreeQuartersHealthy",
        "CodeDeployDefault.AllAtOnce",
        "CodeDeployDefault.HalfAtATime",
        "CodeDeployDefault.OneAtATime"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeploymentConfigs。

list-deployment-groups

以下代码示例演示了如何使用 list-deployment-groups。

Amazon CLI

获取有关部署组的信息

以下 list-deployment-groups 示例显示与指定应用程序关联的所有部署组的相关信息。

```
aws deploy list-deployment-groups --application-name WordPress_App
```

输出：

```
{
    "applicationName": "WordPress_App",
    "deploymentGroups": [
        "WordPress_DG",
        "WordPress_Beta_DG"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeploymentGroups。

list-deployment-instances

以下代码示例演示了如何使用 list-deployment-instances。

Amazon CLI

获取有关部署实例的信息

以下 list-deployment-instances 示例显示与指定部署关联的所有部署实例的相关信息。

```
aws deploy list-deployment-instances \
    --deployment-id d-A1B2C3111 \
    --instance-status-filter Succeeded
```

输出：

```
{
    "instancesList": [
        "i-EXAMPLE11",
        "i-EXAMPLE22"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeploymentInstances。

list-deployment-targets

以下代码示例演示了如何使用 list-deployment-targets。

Amazon CLI

检索与部署关联的目标 ID 列表

以下 list-deployment-targets 示例检索与状态为“Failed”或“InProgress”的部署关联的目标 ID 列表。

```
aws deploy list-deployment-targets \
    --deployment-id "d-A1B2C3111" \
    --target-filters "{\"TargetStatus\":[\"Failed\",\"InProgress\"]}"
```

输出：

```
{
    "targetIds": [
        "i-0f1558aaf90e5f1f9"
    ]
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 ListDeploymentTargets。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeploymentTargets。

list-deployments

以下代码示例演示了如何使用 list-deployments。

Amazon CLI

获取有关部署的信息

以下 list-deployments 示例显示与指定应用程序和部署组关联的所有部署的相关信息。

```
aws deploy list-deployments \
    --application-name WordPress_App \
    --create-time-range start=2014-08-19T00:00:00,end=2014-08-20T00:00:00 \
    --deployment-group-name WordPress_DG \
```

```
    --include-only-statuses Failed
```

输出：

```
{
    "deployments": [
        "d-EXAMPLE11",
        "d-EXAMPLE22",
        "d-EXAMPLE33"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeployments。

`list-git-hub-account-token-names`

以下代码示例演示了如何使用 `list-git-hub-account-token-names`。

Amazon CLI

列出已存储的 GitHub 账户连接的名称

以下 `list-git-hub-account-token-names` 示例列出当前 Amazon 用户已存储的 GitHub 账户连接的名称。

```
aws deploy list-git-hub-account-token-names
```

输出：

```
{
    "tokenNameList": [
        "my-first-token",
        "my-second-token",
        "my-third-token"
    ]
}
```

有关更多信息，请参阅《Amazon CodeDeploy API 参考》中的 ListGitHubAccountTokenNames。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGitHubAccountTokenNames。

list-on-premises-instances

以下代码示例演示了如何使用 list-on-premises-instances。

Amazon CLI

获取有关一个或多个本地实例的信息

以下 list-on-premises-instances 示例检索已在 Amazon CodeDeploy 中注册且在 Amazon CodeDeploy 中与该实例关联的指定本地实例标签的实例的可用本地实例名称列表。

```
aws deploy list-on-premises-instances \
    --registration-status Registered \
    --tag-filters Key=Name,Value=CodeDeployDemo-OnPrem,Type=KEY_AND_VALUE
```

输出：

```
{
    "instanceNames": [
        "AssetTag12010298EX"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOnPremisesInstances。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源（应用程序）的标签

以下 list-tags-for-resource 示例列出应用于 CodeDeploy 中名为 testApp 的应用程序的标签。

```
aws deploy list-tags-for-resource \
    --resource-arn arn:aws:codedeploy:us-west-2:111122223333:application:testApp
```

输出：

```
{
```

```
    "Tags": [
        {
            "Key": "Type",
            "Value": "testType"
        },
        {
            "Key": "Name",
            "Value": "testName"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeDeploy 用户指南》中的在 CodeDeploy 中为部署组标记实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

push

以下代码示例演示了如何使用 push。

Amazon CLI

捆绑与 Amazon CodeDeploy 兼容的应用程序修订并将其部署到 Amazon S3

以下 push 示例将应用程序修订捆绑并部署到 Amazon S3，然后将该应用程序修订与指定的应用程序关联。

```
aws deploy push \
    --application-name WordPress_App \
    --description "This is my deployment" \
    --ignore-hidden-files \
    --s3-location s3://amzn-s3-demo-bucket/WordPressApp.zip \
    --source /tmp/MyLocalDeploymentFolder/
```

输出描述了如何使用 create-deployment 命令创建使用已上传应用程序修订的部署。

```
To deploy with this revision, run:
aws deploy create-deployment --application-name WordPress_App --
deployment-config-name <deployment-config-name> --deployment-group-
name <deployment-group-name> --s3-location bucket=amzn-s3-demo-
bucket,key=WordPressApp.zip,bundleType=zip,eTag="cecc9b8EXAMPLE50a6e71fdb88EXAMPLE",version=
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Push。

register-application-revision

以下代码示例演示了如何使用 register-application-revision。

Amazon CLI

注册有关已上传的应用程序修订的信息

以下 register-application-revision 示例使用 Amazon CodeDeploy 注册有关存储在 Amazon S3 中的已上传应用程序修订的信息。

```
aws deploy register-application-revision \
    --application-name WordPress_App \
    --description "Revised WordPress application" \
    --s3-location bucket=amzn-s3-demo-
bucket,key=RevisedWordPressApp.zip,bundleType=zip,eTag=cecc9b8a08eac650a6e71fdb88EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterApplicationRevision。

register-on-premises-instance

以下代码示例演示了如何使用 register-on-premises-instance。

Amazon CLI

注册本地实例

以下 register-on-premises-instance 示例使用 Amazon CodeDeploy 注册本地实例。它不会创建指定的 IAM 用户，也不会在 Amazon CodeDeploy 中将任何本地实例标签与已注册实例关联。

```
aws deploy register-on-premises-instance \
    --instance-name AssetTag12010298EX \
    --iam-user-arn arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser-OnPrem
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterOnPremisesInstance。

register

以下代码示例演示了如何使用 register。

Amazon CLI

注册本地实例

以下 register 示例向 Amazon CodeDeploy 注册本地实例，在 Amazon CodeDeploy 中将指定的本地实例标签与已注册实例关联，并创建可复制到该实例的本地配置文件。它不会创建 IAM 用户，也不会在实例上安装 Amazon CodeDeploy 代理程序。

```
aws deploy register \
    --instance-name AssetTag12010298EX \
    --iam-user-arn arn:aws:iam::80398EXAMPLE:user/CodeDeployUser-OnPrem \
    --tags Key=Name,Value=CodeDeployDemo-OnPrem \
    --region us-west-2
```

输出：

```
Registering the on-premises instance... DONE
Adding tags to the on-premises instance... DONE
Copy the on-premises configuration file named codedeploy.onpremises.yml to the on-
premises instance, and run the following command on the on-premises instance to
 install and configure the AWS CodeDeploy Agent:
aws deploy install --config-file codedeploy.onpremises.yml
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Register。

remove-tags-from-on-premises-instances

以下代码示例演示了如何使用 remove-tags-from-on-premises-instances。

Amazon CLI

从一个或多个本地实例移除标签

以下 remove-tags-from-on-premises-instances 示例取消了 Amazon CodeDeploy 中指定本地标签与本地实例的关联。它不会在 Amazon CodeDeploy 中注销本地实例，不会从实例中卸载 Amazon CodeDeploy 代理程序，不会从实例中移除本地配置文件，也不会删除与实例关联的 IAM 用户。

```
aws deploy remove-tags-from-on-premises-instances \
    --instance-names AssetTag12010298EX AssetTag23121309EX \
    --tags Key=Name,Value=CodeDeployDemo-OnPrem
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromOnPremisesInstances。

stop-deployment

以下代码示例演示了如何使用 stop-deployment。

Amazon CLI

尝试停止部署

以下 stop-deployment 示例尝试停止与用户的 Amazon 账户关联的正在进行的部署。

aws deploy stop-deployment --deployment-id d-A1B2C3111

输出：

```
{
    "status": "Succeeded",
    "statusMessage": "No more commands will be scheduled for execution in the
 deployment instances"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDeployment。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源（应用程序）

以下 tag-resource 示例在 CodeDeploy 中向名为 testApp 的应用程序添加了两个标签，其键为 Name 和 Type，其值为 testName 和 testType：

```
aws deploy tag-resource \
    --resource-arn  arn:aws:codedeploy:us-west-2:111122223333:application:testApp \
    --tags Key=Name,Value=testName Key=Type,Value=testType
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《Amazon CodeDeploy 用户指南》中的在 CodeDeploy 中为部署组标记实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`uninstall`

以下代码示例演示了如何使用 `uninstall`。

Amazon CLI

卸载本地实例

以下 `uninstall` 示例从本地实例卸载 Amazon CodeDeploy 代理程序并从该实例中移除本地配置文件。它不会在 Amazon CodeDeploy 中注销实例，不会解除 Amazon CodeDeploy 中任何本地实例标签与实例的关联，也不会删除与实例关联的 IAM 用户。

```
aws deploy uninstall
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Uninstall。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源（应用程序）中移除标签

以下 `untag-resource` 示例从 CodeDeploy 中名为 testApp 的应用程序中移除了两个带有键 Name 和 Type 的标签。

```
aws deploy untag-resource \
```

```
    --resource-arn  arn:aws:codedeploy:us-west-2:111122223333:application:testApp \
    --tag-keys Name Type
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《Amazon CodeDeploy 用户指南》中的在 CodeDeploy 中为部署组标记实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-application

以下代码示例演示了如何使用 update-application。

Amazon CLI

更改应用程序的详细信息

以下 update-application 示例更改与用户的 Amazon 账户关联的应用程序名称。

```
aws deploy update-application \
    --application-name WordPress_App \
    --new-application-name My_WordPress_App
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApplication。

update-deployment-group

以下代码示例演示了如何使用 update-deployment-group。

Amazon CLI

更改有关部署组的信息

以下 update-deployment-group 示例更改与指定应用程序关联的部署组的设置。

```
aws deploy update-deployment-group \
    --application-name WordPress_App \
    --auto-scaling-groups My_CodeDeployDemo_ASG \
    --current-deployment-group-name WordPress_DG \
```

```
    --deployment-config-name CodeDeployDefault.AllAtOnce \
    --ec2-tag-filters Key=Name,Type=KEY_AND_VALUE,Value=My_CodeDeployDemo \
    --new-deployment-group-name My_WordPress_DepGroup \
    --service-role-arn arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo-2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeploymentGroup。

使用 Amazon CLI 的 CodeGuru Reviewer 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeGuru Reviewer 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-repository`

以下代码示例演示了如何使用 `associate-repository`。

Amazon CLI

示例 1：创建 Bitbucket 存储库关联

以下 `associate-repository` 示例使用现有 Bitbucket 存储库创建存储库关联。

```
aws codeguru-reviewer associate-repository \
    --repository 'Bitbucket={Owner=sample-owner, Name=mySampleRepo,
 ConnectionArn=arn:aws:codestar-connections:us-west-2:123456789012:connection/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 }'
```

输出：

```
{
    "RepositoryAssociation": {
        "ProviderType": "Bitbucket",
        "Name": "mySampleRepo",
        "LastUpdatedTimeStamp": 1596216896.979,
        "AssociationId": "association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "CreatedTimeStamp": 1596216896.979,
        "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "State": "Associating",
        "StateReason": "Pending Repository Association",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "Owner": "sample-owner"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 Bitbucket 存储库关联。

示例 2：创建 GitHub Enterprise 存储库关联

以下 `associate-repository` 示例使用现有 GitHub Enterprise 存储库创建存储库关联。

```
aws codeguru-reviewer associate-repository \
    --repository 'GitHubEnterpriseServer={Owner=sample-owner, Name=mySampleRepo,
 ConnectionArn=arn:aws:codestar-connections:us-west-2:123456789012:connection/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 }'
```

输出：

```
{
    "RepositoryAssociation": {
        "ProviderType": "GitHubEnterpriseServer",
        "Name": "mySampleRepo",
        "LastUpdatedTimeStamp": 1596216896.979,
        "AssociationId": "association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "CreatedTimeStamp": 1596216896.979,
        "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "State": "Associating",
        "StateReason": "Pending Repository Association",
```

```
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "Owner": "sample-owner"
    }
}
```

有关更多信息，请参阅《Amazon Codeguru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 GitHub Enterprise Server 存储库关联。

示例 3：创建 Amazon CodeCommit 存储库关联

以下 `associate-repository` 示例使用现有 Amazon CodeCommit 存储库创建存储库关联。

```
aws codeguru-reviewer associate-repository \
    --repository CodeCommit={Name=mySampleRepo}
```

输出：

```
{
    "RepositoryAssociation": {
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Name": "My-ecs-beta-repo",
        "LastUpdatedTimeStamp": 1595634764.029,
        "ProviderType": "CodeCommit",
        "CreatedTimeStamp": 1595634764.029,
        "Owner": "544120495673",
        "State": "Associating",
        "StateReason": "Pending Repository Association",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:544120495673:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 Amazon CodeCommit 存储库关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateRepository。

`create-code-review`

以下代码示例演示了如何使用 `create-code-review`。

Amazon CLI

创建代码审查。

以下 `create-code-review` 创建对名为 `my-repository-name` 的 Amazon CodeCommit 存储库 `mainline` 分支中代码的审查。

```
aws codeguru-reviewer create-code-review \
    --name my-code-review \
    --repository-association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --type '{"RepositoryAnalysis": {"RepositoryHead": {"BranchName": "mainline"}}}'
```

输出：

```
{
    "CodeReview": {
        "Name": "my-code-review",
        "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222:code-
review:RepositoryAnalysis-my-code-review",
        "RepositoryName": "my-repository-name",
        "Owner": "123456789012",
        "ProviderType": "CodeCommit",
        "State": "Pending",
        "StateReason": "CodeGuru Reviewer has received the request, and a code
 review is scheduled.",
        "CreatedTimeStamp": 1618873489.195,
        "LastUpdatedTimeStamp": 1618873489.195,
        "Type": "RepositoryAnalysis",
        "SourceCodeType": {
            "RepositoryHead": {
                "BranchName": "mainline"
            }
        },
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建代码审查。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCodeReview。

describe-code-review

以下代码示例演示了如何使用 describe-code-review。

Amazon CLI

列出有关代码审查的详细信息。

以下 describe-code-review 列出 Amazon CodeCommit 存储库中名为“my-repo-name”的“mainline”分支中代码的审查信息。

```
aws codeguru-reviewer put-recommendation-feedback \
    --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678 \
    --recommendation-
id 3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb \
    --reactions ThumbsUp
```

Output

```
{
        "CodeReview": {
            "Name": "My-ecs-beta-repo-master-xs6di4kfd4j269dz",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222:code-
review:RepositoryAnalysis-my-repo-name",
            "RepositoryName": "My-ecs-beta-repo",
            "Owner": "123456789012",
            "ProviderType": "CodeCommit",
            "State": "Pending",
            "StateReason": "CodeGuru Reviewer is reviewing the source code.",
            "CreatedTimeStamp": 1618874226.226,
            "LastUpdatedTimeStamp": 1618874233.689,
            "Type": "RepositoryAnalysis",
            "SourceCodeType": {
                "RepositoryHead": {
                    "BranchName": "mainline"
                }
            },
```

```
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
    }
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的查看代码审查的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCodeReview。

describe-recommendation-feedback

以下代码示例演示了如何使用 describe-recommendation-feedback。

Amazon CLI

查看有关推荐反馈的信息

以下 describe-recommendation-feedback 显示有关推荐反馈的信息。该推荐有一个 ThumbsUp 回应。

```
aws codeguru-reviewer describe-recommendation-feedback \
    --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678 \
    --recommendation-
id 3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb
```

输出：

```
{
    "RecommendationFeedback": {
        "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678",
        "RecommendationId":
 "3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb",
        "Reactions": [
            "ThumbsUp"
        ],
        "UserId": "aws-user-id",
        "CreatedTimeStamp": 1618877070.313,
        "LastUpdatedTimeStamp": 1618877948.881
    }
```

```
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的查看推荐并提供反馈和步骤 4：提供反馈。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRecommendationFeedback。

`describe-repository-association`

以下代码示例演示了如何使用 `describe-repository-association`。

Amazon CLI

示例 1：返回有关 GitHub 存储库关联的信息

以下 `describe-repository-association` 示例返回有关使用 GitHub Enterprise 存储库且处于 `Associated` 状态的存储库关联的信息。

```
aws codeguru-reviewer describe-repository-association \
    --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RepositoryAssociation": {
        "AssociationId": "b822717e-0711-4e8a-bada-0e738289c75e",
        "Name": "mySampleRepo",
        "LastUpdatedTimeStamp": 1588102637.649,
        "ProviderType": "GitHub",
        "CreatedTimeStamp": 1588102615.636,
        "Owner": "sample-owner",
        "State": "Associated",
        "StateReason": "Pull Request Notification configuration successful",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 GitHub Enterprise Server 存储库关联。

示例 2：返回有关失败的存储库关联的信息

以下 `describe-repository-association` 示例返回有关使用 GitHub Enterprise 存储库且处于 `Failed` 状态的存储库关联的信息。

```
aws codeguru-reviewer describe-repository-association \
    --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RepositoryAssociation": {
        "ProviderType": "GitHubEnterpriseServer",
        "Name": "mySampleRepo",
        "LastUpdatedTimeStamp": 1596217036.892,
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreatedTimeStamp": 1596216896.979,
        "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "State": "Failed",
        "StateReason": "Failed, Please retry.",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "Owner": "sample-owner"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 GitHub Enterprise Server 存储库关联。

示例 3：返回有关正在解除关联的存储库关联的信息

以下 `describe-repository-association` 示例返回有关使用 GitHub Enterprise 存储库且处于 `Disassociating` 状态的存储库关联的信息。

```
aws codeguru-reviewer describe-repository-association \
    --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RepositoryAssociation": {
        "ProviderType": "GitHubEnterpriseServer",
        "Name": "mySampleRepo",
        "LastUpdatedTimeStamp": 1596217036.892,
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreatedTimeStamp": 1596216896.979,
        "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "State": "Disassociating",
        "StateReason": "Source code access removal in progress",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "Owner": "sample-owner"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 Amazon CodeGuru Reviewer 中创建 GitHub Enterprise Server 存储库关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRepositoryAssociation。

`disassociate-repository`

以下代码示例演示了如何使用 `disassociate-repository`。

Amazon CLI

解除与存储库关联的关联

以下 `disassociate-repository` 解除正在使用 Amazon CodeCommit 存储库的存储库关联的关联。

```
aws codeguru-reviewer disassociate-repository \
    --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RepositoryAssociation": {
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Name": "my-repository",
        "Owner": "123456789012",
        "ProviderType": "CodeCommit",
        "State": "Disassociating",
        "LastUpdatedTimeStamp": 1618939174.759,
        "CreatedTimeStamp": 1595636947.096
    },
    "Tags": {
        "Status": "Secret",
        "Team": "Saanvi"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 CodeGuru Reviewer 中解除与存储库的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateRepository。

`list-code-reviews`

以下代码示例演示了如何使用 `list-code-reviews`。

Amazon CLI

列出过去 90 天内在您 Amazon 账户中创建的代码审查。

以下 `list-code-reviews` 示例列出在过去 90 天内使用拉取请求创建的代码审查。

```
aws codeguru-reviewer list-code-reviews \
    --type PullRequest
```

输出：

```
{
    "CodeReviewSummaries": [
        {
            "LastUpdatedTimeStamp": 1588897288.054,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProviderType": "GitHub",
            "PullRequestId": "5",
            "MetricsSummary": {
```

```
                "MeteredLinesOfCodeCount": 24,
                "FindingsCount": 1
            },
            "CreatedTimeStamp": 1588897068.512,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
        },
        {
            "LastUpdatedTimeStamp": 1588869793.263,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "ProviderType": "GitHub",
            "PullRequestId": "4",
            "MetricsSummary": {
                "MeteredLinesOfCodeCount": 29,
                "FindingsCount": 0
            },
            "CreatedTimeStamp": 1588869575.949,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
        },
        {
            "LastUpdatedTimeStamp": 1588870511.211,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "ProviderType": "GitHub",
            "PullRequestId": "4",
            "MetricsSummary": {
                "MeteredLinesOfCodeCount": 2,
                "FindingsCount": 0
            },
            "CreatedTimeStamp": 1588870292.425,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
```

```
        },
        {
            "LastUpdatedTimeStamp": 1588118522.452,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "ProviderType": "GitHub",
            "PullRequestId": "3",
            "MetricsSummary": {
                "MeteredLinesOfCodeCount": 29,
                "FindingsCount": 0
            },
            "CreatedTimeStamp": 1588118301.131,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
        },
        {
            "LastUpdatedTimeStamp": 1588112205.207,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
            "ProviderType": "GitHub",
            "PullRequestId": "2",
            "MetricsSummary": {
                "MeteredLinesOfCodeCount": 25,
                "FindingsCount": 0
            },
            "CreatedTimeStamp": 1588111987.443,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
        },
        {
            "LastUpdatedTimeStamp": 1588104489.981,
            "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
            "ProviderType": "GitHub",
            "PullRequestId": "1",
            "MetricsSummary": {
                "MeteredLinesOfCodeCount": 25,
                "FindingsCount": 0
            },
```

```
            "CreatedTimeStamp": 1588104270.223,
            "State": "Completed",
            "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
            "Owner": "sample-owner",
            "RepositoryName": "sample-repository-name",
            "Type": "PullRequest"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的查看所有代码审查。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCodeReviews。

`list-recommendation-feedback`

以下代码示例演示了如何使用 `list-recommendation-feedback`。

Amazon CLI

在关联存储库中列出客户对推荐的推荐反馈

以下 `list-recommendation-feedback` 列出客户对代码审查中所有推荐的反馈。此代码审查包含来自客户的一条反馈，即“ThumbsUp”。

```
aws codeguru-reviewer list-recommendation-feedback \
    --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678
```

输出：

```
{
    "RecommendationFeedbackSummaries": [
        {
            "RecommendationId":
 "3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb",
            "Reactions": [
                "ThumbsUp"
            ],
            "UserId": "aws-user-id"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的步骤 4：提供反馈。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRecommendationFeedback。

`list-recommendations`

以下代码示例演示了如何使用 `list-recommendations`。

Amazon CLI

列出已完成代码审查的推荐

以下 `list-recommendations` 示例列出已完成代码审查的推荐。此代码审查有一条推荐。

```
aws codeguru-reviewer list-recommendations \
    --code-review-arn arn:aws:codeguru-reviewer:us-west-2:544120495673:code-review:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RecommendationSummaries": [
        {
            "Description": "\n\n**Problem**  \n You are using a `ConcurrentHashMap`, but your usage of `containsKey()` and `get()` may not be thread-safe at lines: **63 and 64**. In between the check and the `get()` another thread can remove the key and the `get()` will return `null`. The remove that can remove the key is at line: **59**.\n\n**Fix**  \n Consider calling `get()`, checking instead of your current check if the returned object is `null`, and then using that object only, without calling `get()` again.\n\n**More info**  \n [View an example on GitHub](https://github.com/apache/hadoop/blob/f16cf877e565084c66bc63605659b157c4394dc8/hadoop-tools/hadoop-aws/src/main/java/org/apache/hadoop/fs/s3a/s3guard/S3Guard.java#L302-L304) (external link).",
            "RecommendationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "StartLine": 63,
            "EndLine": 64,
            "FilePath": "src/main/java/com/company/sample/application/CreateOrderThread.java"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的步骤 4：提供反馈。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRecommendations。

list-repository-associations

以下代码示例演示了如何使用 list-repository-associations。

Amazon CLI

列出您 Amazon 账户中的存储库关联

以下 list-repository-associations 示例返回您账户中存储库关联摘要对象的列表。您可以按 ProviderType、Name、State 和 Owner 筛选返回的列表。

```
aws codeguru-reviewer list-repository-associations
```

输出：

```
{
    "RepositoryAssociationSummaries": [
        {
            "LastUpdatedTimeStamp": 1595886609.616,
            "Name": "test",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Owner": "sample-owner",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProviderType": "Bitbucket"
        },
        {
            "LastUpdatedTimeStamp": 1595636969.035,
            "Name": "CodeDeploy-CodePipeline-ECS-Tutorial",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Owner": "123456789012",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "ProviderType": "CodeCommit"
        },
```

```
        {
            "LastUpdatedTimeStamp": 1595634785.983,
            "Name": "My-ecs-beta-repo",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "Owner": "123456789012",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "ProviderType": "CodeCommit"
        },
        {
            "LastUpdatedTimeStamp": 1590712811.77,
            "Name": "MyTestCodeCommit",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "Owner": "123456789012",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "ProviderType": "CodeCommit"
        },
        {
            "LastUpdatedTimeStamp": 1588102637.649,
            "Name": "aws-codeguru-profiler-sample-application",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
            "Owner": "sample-owner",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
            "ProviderType": "GitHub"
        },
        {
            "LastUpdatedTimeStamp": 1588028233.995,
            "Name": "codeguru-profiler-demo-app",
            "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
            "Owner": "sample-owner",
            "State": "Associated",
            "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
            "ProviderType": "GitHub"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的在 CodeGuru Reviewer 中查看所有存储库关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRepositoryAssociations。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出关联存储库上的标签

以下 list-tags-for-resource 列出关联存储库上的标签。此关联存储库有两个标签。

```
aws codeguru-reviewer list-tags-for-resource \
    --resource-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Tags": {
        "Status": "Secret",
        "Team": "Saanvi"
    }
}
```

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的查看 CodeGuru Reviewer 关联存储库的标签（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-recommendation-feedback

以下代码示例演示了如何使用 put-recommendation-feedback。

Amazon CLI

向代码审查添加推荐

以下 put-recommendation-feedback 为代码审查添加 ThumbsUp 推荐。

```
aws codeguru-reviewer put-recommendation-feedback \
    --code-review-arn \arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678 \
    --recommendation-
id 3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb \
    --reactions ThumbsUp
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的步骤 4：提供反馈。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRecommendationFeedback。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向关联存储库添加标签

以下 `tag-resource` 向关联存储库添加两个标签。

```
aws codeguru-reviewer tag-resource \
    --resource-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --tags Status=Secret,Team=Saanvi
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的向 CodeGuru Reviewer 关联存储库添加标签（Amazon CLI）和在 CodeGuru Reviewer 关联存储库中添加或更新标签（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

取消关联存储库的标记

以下 `untag-resource` 从关联存储库中移除两个密钥为“Secret”和“Team”的标签。

```
aws codeguru-reviewer untag-resource \
    --resource-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --tag-keys Status Team
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeGuru Reviewer 用户指南》中的从 CodeGuru Reviewer 关联存储库中移除标签（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 CodePipeline 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodePipeline 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`acknowledge-job`

以下代码示例演示了如何使用 `acknowledge-job`。

Amazon CLI

检索有关指定任务的信息

此示例返回有关指定任务的信息，包括该任务的状态（如果存在）。这仅用于任务工作人员和自定义操作。要确定 nonce 值和作业 ID，请使用 aws codepipeline poll-for-jobs。

命令:

```
aws codepipeline acknowledge-job --job-id f4f4ff82-2d11-EXAMPLE --nonce 3
```

输出：

```
{
  "status": "InProgress"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcknowledgeJob。

create-custom-action-type

以下代码示例演示了如何使用 create-custom-action-type。

Amazon CLI

创建自定义操作

此示例使用包含自定义操作结构的已创建的 JSON 文件（此处名为 MyCustomAction.json）为 Amazon CodePipeline 创建自定义操作。有关创建自定义操作的要求（包括文件结构）的更多信息，请参阅《Amazon CodePipeline 用户指南》。

```
aws codepipeline create-custom-action-type --cli-input-json file://
MyCustomAction.json
```

JSON 文件 MyCustomAction.json 的内容：

```
{
    "category": "Build",
    "provider": "MyJenkinsProviderName",
    "version": "1",
    "settings": {
        "entityUrlTemplate": "https://192.0.2.4/job/{Config:ProjectName}/",
        "executionUrlTemplate": "https://192.0.2.4/job/{Config:ProjectName}/
lastSuccessfulBuild/{ExternalExecutionId}/"
```

```
    },
    "configurationProperties": [
        {
            "name": "MyJenkinsExampleBuildProject",
            "required": true,
            "key": true,
            "secret": false,
            "queryable": false,
            "description": "The name of the build project must be provided when this
 action is added to the pipeline.",
            "type": "String"
        }
    ],
    "inputArtifactDetails": {
        "maximumCount": 1,
        "minimumCount": 0
    },
    "outputArtifactDetails": {
        "maximumCount": 1,
        "minimumCount": 0
    }
}
```

该命令会返回自定义操作的结构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomActionType。

create-pipeline

以下代码示例演示了如何使用 create-pipeline。

Amazon CLI

创建管道

此示例使用包含管道结构的已创建的 JSON 文件（此处名为 MySecondPipeline.json）在 Amazon CodePipeline 中创建管道。有关创建管道的要求（包括文件结构）的更多信息，请参阅《Amazon CodePipeline 用户指南》。

命令:

```
aws codepipeline create-pipeline --cli-input-json file://MySecondPipeline.json
```

JSON 文件示例内容：

```
{
 "pipeline": {
  "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
  "stages": [
    {
      "name": "Source",
      "actions": [
        {
          "inputArtifacts": [],
          "name": "Source",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "configuration": {
            "S3Bucket": "awscodepipeline-demo-bucket",
            "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
          },
          "runOrder": 1
        }
      ]
    },
    {
      "name": "Beta",
      "actions": [
        {
          "inputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "name": "CodePipelineDemoFleet",
          "actionTypeId": {
            "category": "Deploy",
            "owner": "AWS",
```

```
            "version": "1",
            "provider": "CodeDeploy"
          },
          "outputArtifacts": [],
          "configuration": {
            "ApplicationName": "CodePipelineDemoApplication",
            "DeploymentGroupName": "CodePipelineDemoFleet"
          },
          "runOrder": 1
        }
      ]
    }
  ],
  "artifactStore": {
    "type": "S3",
    "location": "codepipeline-us-east-1-11EXAMPLE11"
  },
  "name": "MySecondPipeline",
  "version": 1
 }
}
```

输出：

```
This command returns the structure of the pipeline.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePipeline。

delete-custom-action-type

以下代码示例演示了如何使用 delete-custom-action-type。

Amazon CLI

删除自定义操作

此示例使用包含要删除的操作的操作类型、提供程序名称和版本号的已创建的 JSON 文件（此处名为 DeleteMyCustomAction.json）删除 Amazon CodePipeline 中的自定义操作。使用 list-action-types 命令查看类别、版本和提供程序的正确值。

命令:

```
aws codepipeline delete-custom-action-type --cli-input-json file://
DeleteMyCustomAction.json
```

JSON 文件示例内容：

```
{
  "category": "Build",
  "version": "1",
  "provider": "MyJenkinsProviderName"
}
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCustomActionType。

delete-pipeline

以下代码示例演示了如何使用 delete-pipeline。

Amazon CLI

删除管道

此示例从 Amazon CodePipeline 中删除了一个名为 MySecondPipeline 的管道。使用 list-pipelines 命令查看与您 Amazon 账户关联的管道列表。

命令:

```
aws codepipeline delete-pipeline --name MySecondPipeline
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePipeline。

delete-webhook

以下代码示例演示了如何使用 delete-webhook。

Amazon CLI

删除 webhook

以下 delete-webhook 示例删除 GitHub 版本 1 源操作的 webhook。必须先使用 deregister-webhook-with-third-party 命令取消注册 webhook，然后才能删除它。

```
aws codepipeline delete-webhook \
    --name my-webhook
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的删除 GitHub 源的 webhook。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWebhook。

deregister-webhook-with-third-party

以下代码示例演示了如何使用 deregister-webhook-with-third-party。

Amazon CLI

取消注册 webhook

以下 deregister-webhook-with-third-party 示例删除 GitHub 版本 1 源操作的 webhook。必须先取消注册 webhook，然后才能删除它。

```
aws codepipeline deregister-webhook-with-third-party \
    --webhook-name my-webhook
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的删除 GitHub 源的 webhook。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterWebhookWithThirdParty。

disable-stage-transition

以下代码示例演示了如何使用 disable-stage-transition。

Amazon CLI

禁用向管道中某个阶段的过渡

此示例禁止在 Amazon CodePipeline 中过渡到 MyFirstPipeline 管道的 Beta 阶段。

命令:

```
aws codepipeline disable-stage-transition --pipeline-name MyFirstPipeline --stage-name Beta  --transition-type Inbound
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableStageTransition。

enable-stage-transition

以下代码示例演示了如何使用 enable-stage-transition。

Amazon CLI

启用向管道中某个阶段的过渡

此示例启用在 Amazon CodePipeline 中过渡到 MyFirstPipeline 管道的 Beta 阶段。

命令:

```
aws codepipeline enable-stage-transition --pipeline-name MyFirstPipeline --stage-name Beta  --transition-type Inbound
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableStageTransition。

get-job-details

以下代码示例演示了如何使用 get-job-details。

Amazon CLI

获取任务的详细信息

此示例返回有关 ID 为 f4f4ff82-2d11-EXAMPLE 的任务的详细信息。此命令仅用于自定义操作。调用此命令时，如果自定义操作需要，Amazon CodePipeline 会返回用于存储管道构件的 Amazon S3 存储桶的临时凭证。此命令还将返回为操作定义的所有密钥值（如果有）。

命令:

```
aws codepipeline get-job-details --job-id f4f4ff82-2d11-EXAMPLE
```

输出：

```
{
 "jobDetails": {
  "accountId": "111111111111",
  "data": {
    "actionConfiguration": {
      "__type": "ActionConfiguration",
      "configuration": {
        "ProjectName": "MyJenkinsExampleTestProject"
      }
    },
    "actionTypeId": {
      "__type": "ActionTypeId",
      "category": "Test",
      "owner": "Custom",
      "provider": "MyJenkinsProviderName",
      "version": "1"
    },
    "artifactCredentials": {
      "__type": "AWSSessionCredentials",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "secretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
      "sessionToken":
 "fICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwd
+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/
```

```
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpEIbb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQ
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs
    },
    "inputArtifacts": [
      {
        "__type": "Artifact",
        "location": {
          "s3Location": {
            "bucketName": "codepipeline-us-east-1-11EXAMPLE11",
            "objectKey": "MySecondPipeline/MyAppBuild/EXAMPLE"
          },
          "type": "S3"
        },
        "name": "MyAppBuild"
      }
    ],
    "outputArtifacts": [],
    "pipelineContext": {
      "__type": "PipelineContext",
      "action": {
        "name": "MyJenkinsTest-Action"
      },
      "pipelineName": "MySecondPipeline",
      "stage": {
        "name": "Testing"
      }
    }
  },
  "id": "f4f4ff82-2d11-EXAMPLE"
 }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobDetails。

get-pipeline-state

以下代码示例演示了如何使用 get-pipeline-state。

Amazon CLI

获取有关管道状态的信息

此示例返回名为 MyFirstPipeline 管道的最新状态。

命令:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

输出：

```
{
 "created": 1446137312.204,
 "pipelineName": "MyFirstPipeline",
 "pipelineVersion": 1,
 "stageStates": [
  {
    "actionStates": [
      {
        "actionName": "Source",
        "entityUrl": "https://console.aws.amazon.com/s3/home?#",
        "latestExecution": {
          "lastStatusChange": 1446137358.328,
          "status": "Succeeded"
        }
      }
    ],
    "stageName": "Source"
  },
  {
    "actionStates": [
      {
        "actionName": "CodePipelineDemoFleet",
        "entityUrl": "https://console.aws.amazon.com/codedeploy/home?#/applications/
CodePipelineDemoApplication/deployment-groups/CodePipelineDemoFleet",
        "latestExecution": {
          "externalExecutionId": "d-EXAMPLE",
          "externalExecutionUrl": "https://console.aws.amazon.com/codedeploy/home?#/
deployments/d-EXAMPLE",
          "lastStatusChange": 1446137493.131,
          "status": "Succeeded",
          "summary": "Deployment Succeeded"
        }
      }
    ],
    "inboundTransitionState": {
      "enabled": true
    },
```

```
    "stageName": "Beta"
  }
 ],
 "updated": 1446137312.204
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPipelineState。

get-pipeline

以下代码示例演示了如何使用 get-pipeline。

Amazon CLI

查看管道的结构

此示例返回名为 MyFirstPipeline 管道的结构。

命令:

```
aws codepipeline get-pipeline --name MyFirstPipeline
```

输出：

```
{
  "pipeline": {
      "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
      "stages": [
          {
              "name": "Source",
              "actions": [
                  {
                      "inputArtifacts": [],
                      "name": "Source",
                      "actionTypeId": {
                          "category": "Source",
                          "owner": "AWS",
                          "version": "1",
                          "provider": "S3"
                      },
                      "outputArtifacts": [
                          {
                              "name": "MyApp"
```

```
                        }
                    ],
                    "configuration": {
                        "S3Bucket": "awscodepipeline-demo-bucket",
                        "S3ObjectKey": "aws-codepipeline-s3-aws-
codedeploy_linux.zip"
                    },
                    "runOrder": 1
                }
            ]
        },
        {
            "name": "Beta",
            "actions": [
                {
                    "inputArtifacts": [
                        {
                            "name": "MyApp"
                        }
                    ],
                    "name": "CodePipelineDemoFleet",
                    "actionTypeId": {
                        "category": "Deploy",
                        "owner": "AWS",
                        "version": "1",
                        "provider": "CodeDeploy"
                    },
                    "outputArtifacts": [],
                    "configuration": {
                        "ApplicationName": "CodePipelineDemoApplication",
                        "DeploymentGroupName": "CodePipelineDemoFleet"
                    },
                    "runOrder": 1
                }
            ]
        }
    ],
    "artifactStore": {
        "type": "S3",
        "location": "codepipeline-us-east-1-11EXAMPLE11"
    },
    "name": "MyFirstPipeline",
    "version": 1
}
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPipeline。

list-action-executions

以下代码示例演示了如何使用 list-action-executions。

Amazon CLI

列出操作执行

以下 list-action-executions 示例查看管道的操作执行详细信息，例如操作执行 ID、输入构件、输出构件、执行结果和状态。

```
aws codepipeline list-action-executions \
    --pipeline-name myPipeline
```

输出：

```
{
    "actionExecutionDetails": [
        {
            "pipelineExecutionId": "EXAMPLE0-adfc-488e-bf4c-1111111720d3",
            "actionExecutionId": "EXAMPLE4-2ee8-4853-bd6a-111111158148",
            "pipelineVersion": 12,
            "stageName": "Deploy",
            "actionName": "Deploy",
            "startTime": 1598572628.6,
            "lastUpdateTime": 1598572661.255,
            "status": "Succeeded",
            "input": {
                "actionTypeId": {
                    "category": "Deploy",
                    "owner": "AWS",
                    "provider": "CodeDeploy",
                    "version": "1"
                },
                "configuration": {
                    "ApplicationName": "my-application",
                    "DeploymentGroupName": "my-deployment-group"
                },
```

```
                "resolvedConfiguration": {
                    "ApplicationName": "my-application",
                    "DeploymentGroupName": "my-deployment-group"
                },
                "region": "us-east-1",
                "inputArtifacts": [
                    {
                        "name": "SourceArtifact",
                        "s3location": {
                            "bucket": "artifact-bucket",
                            "key": "myPipeline/SourceArti/key"
                        }
                    }
                ],
                "namespace": "DeployVariables"
            },
            "output": {
                "outputArtifacts": [],
                "executionResult": {
                    "externalExecutionId": "d-EXAMPLEE5",
                    "externalExecutionSummary": "Deployment Succeeded",
                    "externalExecutionUrl": "https://myaddress.com"
                },
                "outputVariables": {}
            }
        },
        {
            "pipelineExecutionId": "EXAMPLE0-adfc-488e-bf4c-1111111720d3",
            "actionExecutionId": "EXAMPLE5-abb4-4192-9031-11111113a7b0",
            "pipelineVersion": 12,
            "stageName": "Source",
            "actionName": "Source",
            "startTime": 1598572624.387,
            "lastUpdateTime": 1598572628.16,
            "status": "Succeeded",
            "input": {
                "actionTypeId": {
                    "category": "Source",
                    "owner": "AWS",
                    "provider": "CodeCommit",
                    "version": "1"
                },
                "configuration": {
                    "BranchName": "production",
```

```
                    "PollForSourceChanges": "false",
                    "RepositoryName": "my-repo"
                },
                "resolvedConfiguration": {
                    "BranchName": "production",
                    "PollForSourceChanges": "false",
                    "RepositoryName": "my-repo"
                },
                "region": "us-east-1",
                "inputArtifacts": [],
                "namespace": "SourceVariables"
            },
            "output": {
                "outputArtifacts": [
                    {
                        "name": "SourceArtifact",
                        "s3location": {
                            "bucket": "amzn-s3-demo-bucket",
                            "key": "myPipeline/SourceArti/key"
                        }
                    }
                ],
                "executionResult": {
                    "externalExecutionId":
 "1111111ad99dcd35914c00b7fbea13995EXAMPLE",
                    "externalExecutionSummary": "Edited template.yml",
                    "externalExecutionUrl": "https://myaddress.com"
                },
                "outputVariables": {
                    "AuthorDate": "2020-05-08T17:45:43Z",
                    "BranchName": "production",
                    "CommitId": "EXAMPLEad99dcd35914c00b7fbea139951111111",
                    "CommitMessage": "Edited template.yml",
                    "CommitterDate": "2020-05-08T17:45:43Z",
                    "RepositoryName": "my-repo"
                }
            }
        },
. . . .
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的查看操作执行（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListActionExecutions。

list-action-types

以下代码示例演示了如何使用 list-action-types。

Amazon CLI

查看可用的操作类型

单独使用 list-action-types 命令时，会返回您 Amazon 账户可用的所有操作的结构。此示例使用 --action-owner-filter 选项仅返回自定义操作。

命令:

```
aws codepipeline list-action-types --action-owner-filter Custom
```

输出：

```
{
  "actionTypes": [
      {
          "inputArtifactDetails": {
              "maximumCount": 5,
              "minimumCount": 0
          },
          "actionConfigurationProperties": [
              {
                  "secret": false,
                  "required": true,
                  "name": "MyJenkinsExampleBuildProject",
                  "key": true,
                  "queryable": true
              }
          ],
          "outputArtifactDetails": {
              "maximumCount": 5,
              "minimumCount": 0
          },
          "id": {
              "category": "Build",
              "owner": "Custom",
              "version": "1",
              "provider": "MyJenkinsProviderName"
          },
          "settings": {
```

```
            "entityUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}",
            "executionUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}/
{ExternalExecutionId}"
        }
    },
    {
        "inputArtifactDetails": {
            "maximumCount": 5,
            "minimumCount": 0
        },
        "actionConfigurationProperties": [
            {
                "secret": false,
                "required": true,
                "name": "MyJenkinsExampleTestProject",
                "key": true,
                "queryable": true
            }
        ],
        "outputArtifactDetails": {
            "maximumCount": 5,
            "minimumCount": 0
        },
        "id": {
            "category": "Test",
            "owner": "Custom",
            "version": "1",
            "provider": "MyJenkinsProviderName"
        },
        "settings": {
            "entityUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}",
            "executionUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}/
{ExternalExecutionId}"
        }
    }
]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListActionTypes。

list-pipeline-executions

以下代码示例演示了如何使用 list-pipeline-executions。

Amazon CLI

查看管道执行历史记录

以下 `list-pipeline-executions` 示例显示您 Amazon 账户中管道的管道执行历史记录。

```
aws codepipeline list-pipeline-executions \
    --pipeline-name MyPipeline
```

输出：

```
{
    "pipelineExecutionSummaries": [
        {
            "lastUpdateTime": 1496380678.648,
            "pipelineExecutionId": "7cf7f7cb-3137-539g-j458-d7eu3EXAMPLE",
            "startTime": 1496380258.243,
            "status": "Succeeded"
        },
        {
            "lastUpdateTime": 1496591045.634,
            "pipelineExecutionId": "3137f7cb-8d494hj4-039j-d84l-d7eu3EXAMPLE",
            "startTime": 1496590401.222,
            "status": "Succeeded"
        },
        {
            "lastUpdateTime": 1496946071.6456,
            "pipelineExecutionId": "4992f7jf-7cf7-913k-k334-d7eu3EXAMPLE",
            "startTime": 1496945471.5645,
            "status": "Succeeded"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的查看执行历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPipelineExecutions。

list-pipelines

以下代码示例演示了如何使用 `list-pipelines`。

Amazon CLI

查看管道列表

此示例列出与用户的 Amazon 帐户关联的所有 Amazon CodePipeline 管道。

命令:

```
aws codepipeline list-pipelines
```

输出：

```
{
  "pipelines": [
      {
          "updated": 1439504274.641,
          "version": 1,
          "name": "MyFirstPipeline",
          "created": 1439504274.641
      },
      {
          "updated": 1436461837.992,
          "version": 2,
          "name": "MySecondPipeline",
          "created": 1436460801.381
      }
      ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPipelines。

list-tags-for-resource

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出标签

以下 `list-tags-for-resource` 示例检索附加到指定管道资源的所有标签的列表。

```
aws codepipeline list-tags-for-resource \
```

```
    --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline
```

输出：

```
{
    "tags": {
        "Project": "ProjectA",
        "IscontainerBased": "true"
    }
}
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的查看管道的标签（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-webhooks

以下代码示例演示了如何使用 list-webhooks。

Amazon CLI

列出 webhook

以下 list-webhooks 示例检索附加到指定管道资源的所有标签的列表。

```
aws codepipeline list-webhooks \
    --endpoint-url "https://codepipeline.eu-central-1.amazonaws.com" \
    --region "eu-central-1"
```

输出：

```
{
    "webhooks": [
        {
            "url": "https://webhooks.domain.com/
trigger111111111EXAMPLE1111111111111111": {
                "authenticationConfiguration": {
                    "SecretToken": "Secret"
                },
                "name": "my-webhook",
                "authentication": "GITHUB_HMAC",
                "targetPipeline": "my-Pipeline",
```

```
                "targetAction": "Source",
                "filters": [
                    {
                        "jsonPath": "$.ref",
                        "matchEquals": "refs/heads/{Branch}"
                    }
                ]
            },
            "arn": "arn:aws:codepipeline:eu-central-1:123456789012:webhook:my-
webhook"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的列出您账户中的 webhook。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWebhooks。

poll-for-jobs

以下代码示例演示了如何使用 poll-for-jobs。

Amazon CLI

查看任何可用任务

此示例返回有关任何任务的信息，供任务工作人员采取行动。此示例使用预定义的 JSON 文件（MyActionTypeInfo.json）提供有关任务工作人员处理任务的操作类型的信息。此命令仅用于自定义操作。调用此命令时，Amazon CodePipeline 会返回用于存储管道构件的 Amazon S3 存储桶的临时凭证。此命令还将返回为操作定义的所有密钥值（如果有）。

命令:

```
aws codepipeline poll-for-jobs --cli-input-json file://MyActionTypeInfo.json
```

JSON 文件示例内容：

```
{
  "actionTypeId": {
    "category": "Test",
    "owner": "Custom",
    "provider": "MyJenkinsProviderName",
```

```
    "version": "1"
  },
  "maxBatchSize": 5,
  "queryParam": {
      "ProjectName": "MyJenkinsTestProject"
  }
}
```

输出：

```
{
 "jobs": [
  {
    "accountId": "111111111111",
    "data": {
      "actionConfiguration": {
        "__type": "ActionConfiguration",
        "configuration": {
          "ProjectName": "MyJenkinsExampleTestProject"
        }
      },
      "actionTypeId": {
        "__type": "ActionTypeId",
        "category": "Test",
        "owner": "Custom",
        "provider": "MyJenkinsProviderName",
        "version": "1"
      },
      "artifactCredentials": {
        "__type": "AWSSessionCredentials",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "secretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "sessionToken":
 "fICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwd
+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpEIbb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQ
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs
      },
      "inputArtifacts": [
        {
          "__type": "Artifact",
          "location": {
```

```
            "s3Location": {
              "bucketName": "codepipeline-us-east-1-11EXAMPLE11",
              "objectKey": "MySecondPipeline/MyAppBuild/EXAMPLE"
            },
            "type": "S3"
          },
          "name": "MyAppBuild"
        }
      ],
      "outputArtifacts": [],
      "pipelineContext": {
        "__type": "PipelineContext",
        "action": {
          "name": "MyJenkinsTest-Action"
        },
        "pipelineName": "MySecondPipeline",
        "stage": {
          "name": "Testing"
        }
      }
    },
    "id": "ef66c259-64f9-EXAMPLE",
    "nonce": "3"
  }
 ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PollForJobs。

put-webhook

以下代码示例演示了如何使用 put-webhook。

Amazon CLI

创建 webhook

以下 put-webhook 示例创建 GitHub 版本 1 源操作的 webhook。创建 webhook 后，必须使用 register-webhook-with-third-party 命令进行注册。

```
aws codepipeline put-webhook \
    --cli-input-json file://webhook_json.json \
```

```
    --region "eu-central-1"
```

的内容webhook_json.json：

```
{
    "webhook": {
        "name": "my-webhook",
        "targetPipeline": "pipeline_name",
        "targetAction": "source_action_name",
        "filters": [
            {
                "jsonPath": "$.ref",
                "matchEquals": "refs/heads/{Branch}"
            }
        ],
        "authentication": "GITHUB_HMAC",
        "authenticationConfiguration": {
            "SecretToken": "secret"
        }
    }
}
```

输出：

```
{
    "webhook": {
        "url": "https://webhooks.domain.com/
trigger111111111EXAMPLE11111111111111111",
        "definition": {
            "authenticationConfiguration": {
                "SecretToken": "secret"
            },
            "name": "my-webhook",
            "authentication": "GITHUB_HMAC",
            "targetPipeline": "pipeline_name",
            "targetAction": "Source",
            "filters": [
                {
                    "jsonPath": "$.ref",
                    "matchEquals": "refs/heads/{Branch}"
                }
            ]
        },
```

```
        "arn": "arn:aws:codepipeline:eu-central-1:123456789012:webhook:my-webhook"
    },
    "tags": [
        {
            "key": "Project",
            "value": "ProjectA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的创建 GitHub 源 webhook。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutWebhook。

retry-stage-execution

以下代码示例演示了如何使用 retry-stage-execution。

Amazon CLI

重试失败的操作

以下 retry-stage-execution 示例重试操作失败的阶段。

```
aws codepipeline retry-stage-execution \
    --pipeline-name MyPipeline \
    --stage-name Deploy \
    --pipeline-execution-id b59babff-5f34-EXAMPLE \
    --retry-mode FAILED_ACTIONS
```

输出：

```
{
    "pipelineExecutionId": "b59babff-5f34-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的重试失败的操作（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetryStageExecution。

start-pipeline-execution

以下代码示例演示了如何使用 start-pipeline-execution。

Amazon CLI

通过管道运行最新版本

此示例通过名为“MyFirstPipeline”的管道运行管道源阶段中存在的最新版本。

命令:

```
aws codepipeline start-pipeline-execution --name MyFirstPipeline
```

输出：

```
{
  "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPipelineExecution。

stop-pipeline-execution

以下代码示例演示了如何使用 stop-pipeline-execution。

Amazon CLI

停止管道执行

以下 stop-pipeline-execution 示例默认为等待正在进行的操作完成，然后停止管道。如果执行已处于 Stopping 状态，则无法选择停止并等待。您可以选择停止并放弃已处于 Stopping 状态的执行。

```
aws codepipeline stop-pipeline-execution \
    --pipeline-name MyFirstPipeline \
    --pipeline-execution-id d-EXAMPLE \
    --reason "Stopping pipeline after the build action is done"
```

此命令不返回任何输出。

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的停止管道执行（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopPipelineExecution。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例将一组提供的标签与管道关联。使用此命令添加或编辑标签。

```
aws codepipeline tag-resource \
    --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline \
    --tags key=Project,value=ProjectA key=IscontainerBased,value=true
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的向管道添加标签（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除连接资源的 Amazon 标签

以下 untag-resource 示例移除指定资源的标签。

```
aws codepipeline untag-resource \
    --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline \
    --tag-keys Project IscontainerBased
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodePipeline 用户指南》中的移除管道的标签（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-pipeline

以下代码示例演示了如何使用 update-pipeline。

Amazon CLI

更新管道的结构

此示例使用带有 --cli-input-json 参数的 update-pipeline 命令。此示例使用预定义的 JSON 文件（MyFirstPipeline.json）更新管道的结构。AmazonCodePipeline 会识别 JSON 文件中包含的管道名称，然后应用管道结构中已修改字段的任何更改来更新管道。

创建预定义 JSON 文件时，请遵循以下指南：

如果使用 get-pipeline 命令检索管道结构，则必须移除 JSON 文件中管道结构的元数据部分（“metadata”：{ } 行以及其中的“created”、“pipelineARN”和“updated”字段）。管道名称无法更改。

命令:

```
aws codepipeline update-pipeline --cli-input-json file://MyFirstPipeline.json
```

JSON 文件内容示例：

```
{
 "pipeline": {
  "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
  "stages": [
    {
      "name": "Source",
      "actions": [
        {
          "inputArtifacts": [],
          "name": "Source",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
          },
          "outputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "configuration": {
```

```
            "S3Bucket": "awscodepipeline-demo-bucket2",
            "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
          },
          "runOrder": 1
        }
      ]
    },
    {
      "name": "Beta",
      "actions": [
        {
          "inputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "name": "CodePipelineDemoFleet",
          "actionTypeId": {
            "category": "Deploy",
            "owner": "AWS",
            "version": "1",
            "provider": "CodeDeploy"
          },
          "outputArtifacts": [],
          "configuration": {
            "ApplicationName": "CodePipelineDemoApplication",
            "DeploymentGroupName": "CodePipelineDemoFleet"
          },
          "runOrder": 1
        }
      ]
    }
  ],
  "artifactStore": {
    "type": "S3",
    "location": "codepipeline-us-east-1-11EXAMPLE11"
  },
  "name": "MyFirstPipeline",
  "version": 1
 }
}
```

输出：

```
{
 "pipeline": {
  "artifactStore": {
    "location": "codepipeline-us-east-1-11EXAMPLE11",
    "type": "S3"
  },
  "name": "MyFirstPipeline",
  "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
  "stages": [
    {
      "actions": [
        {
          "actionTypeId": {
            "__type": "ActionTypeId",
            "category": "Source",
            "owner": "AWS",
            "provider": "S3",
            "version": "1"
          },
          "configuration": {
            "S3Bucket": "awscodepipeline-demo-bucket2",
            "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
          },
          "inputArtifacts": [],
          "name": "Source",
          "outputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "runOrder": 1
        }
      ],
      "name": "Source"
    },
    {
      "actions": [
        {
          "actionTypeId": {
            "__type": "ActionTypeId",
            "category": "Deploy",
            "owner": "AWS",
            "provider": "CodeDeploy",
```

```
            "version": "1"
          },
          "configuration": {
            "ApplicationName": "CodePipelineDemoApplication",
            "DeploymentGroupName": "CodePipelineDemoFleet"
          },
          "inputArtifacts": [
            {
              "name": "MyApp"
            }
          ],
          "name": "CodePipelineDemoFleet",
          "outputArtifacts": [],
          "runOrder": 1
        }
      ],
      "name": "Beta"
    }
  ],
  "version": 3
 }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePipeline。

使用 Amazon CLI 的 AWS CodeStar Notifications 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 AWS CodeStar Notifications 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-notification-rule`

以下代码示例演示了如何使用 `create-notification-rule`。

Amazon CLI

创建通知规则

以下 `create-notification-rule` 示例使用名为 `rule.json` 的 JSON 文件为指定 Amazon 账户中名为 `MyDemoRepo` 的存储库创建名为 `MyNotificationRule` 的通知规则。在创建分支和标签时，具有 `FULL` 详情类型的通知将发送到指定的目标 Amazon SNS 主题。

```
aws codestar-notifications create-notification-rule \
    --cli-input-json file://rule.json
```

的内容`rule.json`：

```
{
    "Name": "MyNotificationRule",
    "EventTypeIds": [
        "codecommit-repository-branches-and-tags-created"
    ],
    "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",
    "Targets": [
        {
            "TargetType": "SNS",
            "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

输出：

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
```

```
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的创建通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNotificationRule。

delete-notification-rule

以下代码示例演示了如何使用 delete-notification-rule。

Amazon CLI

删除通知规则

以下 delete-notification-rule 示例删除指定的通知规则。

```
aws codestar-notifications delete-notification-rule \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE
```

输出：

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的删除通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNotificationRule。

delete-target

以下代码示例演示了如何使用 delete-target。

Amazon CLI

删除通知规则目标

以下 delete-target 示例将指定目标从所有将其配置为目标的通知规则中移除，然后删除目标。

```
aws codestar-notifications  delete-target \
```

```
    --target-address arn:aws:sns:us-east-1:123456789012:MyNotificationTopic \
    --force-unsubscribe-all
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的删除通知规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTarget。

describe-notification-rule

以下代码示例演示了如何使用 describe-notification-rule。

Amazon CLI

检索通知规则的详细信息

以下 describe-notification-rule 示例检索指定通知规则的详细信息。

```
aws codestar-notifications describe-notification-rule \
    --arn arn:aws:codestar-notifications:us-west-2:123456789012:notificationrule/
dc82df7a-EXAMPLE
```

输出：

```
{
    "LastModifiedTimestamp": 1569199844.857,
    "EventTypes": [
        {
            "ServiceName": "CodeCommit",
            "EventTypeName": "Branches and tags: Created",
            "ResourceType": "Repository",
            "EventTypeId": "codecommit-repository-branches-and-tags-created"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL",
    "Resource": "arn:aws:codecommit:us-west-2:123456789012:MyDemoRepo",
    "Arn": "arn:aws:codestar-notifications:us-west-w:123456789012:notificationrule/
dc82df7a-EXAMPLE",
    "Targets": [
        {
```

```
            "TargetStatus": "ACTIVE",
            "TargetAddress": "arn:aws:sns:us-
west-2:123456789012:MyNotificationTopic",
            "TargetType": "SNS"
        }
    ],
    "Name": "MyNotificationRule",
    "CreatedTimestamp": 1569199844.857,
    "CreatedBy": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的查看通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNotificationRule。

`list-event-types`

以下代码示例演示了如何使用 `list-event-types`。

Amazon CLI

获取通知规则的事件类型列表

以下 `list-event-types` 示例检索 CodeDeploy 应用程序所有可用通知事件类型的筛选列表。相反，如果您不使用任何筛选条件，则该命令将返回所有资源类型的所有通知事件类型。

```
aws codestar-notifications list-event-types \
    --filters Name=SERVICE_NAME,Value=CodeDeploy
```

输出：

```
{
    "EventTypes": [
        {
            "EventTypeId": "codedeploy-application-deployment-succeeded",
            "ServiceName": "CodeDeploy",
            "EventTypeName": "Deployment: Succeeded",
            "ResourceType": "Application"
        },
        {
            "EventTypeId": "codedeploy-application-deployment-failed",
            "ServiceName": "CodeDeploy",
```

```
            "EventTypeName": "Deployment: Failed",
            "ResourceType": "Application"
        },
        {
            "EventTypeId": "codedeploy-application-deployment-started",
            "ServiceName": "CodeDeploy",
            "EventTypeName": "Deployment: Started",
            "ResourceType": "Application"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的创建通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEventTypes。

`list-notification-rules`

以下代码示例演示了如何使用 `list-notification-rules`。

Amazon CLI

检索通知规则列表

以下 `list-notification-rules` 示例检索指定 Amazon 区域内的所有通知规则的列表。

```
aws codestar-notifications list-notification-rules --region us-east-1
```

输出：

```
{
    "NotificationRules": [
        {
            "Id": "dc82df7a-EXAMPLE",
            "Arn": "arn:aws:codestar-notifications:us-
east-1:123456789012:notificationrule/dc82df7a-EXAMPLE"
        },
        {
            "Id": "8d1f0983-EXAMPLE",
            "Arn": "arn:aws:codestar-notifications:us-
east-1:123456789012:notificationrule/8d1f0983-EXAMPLE"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的查看通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListNotificationRules。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

获取附加到通知规则的标签列表

以下 `list-tags-for-resource` 示例检索附加到指定通知规则的所有标签的列表。在此示例中，通知规则当前没有与之关联的标签。

```
aws codestar-notifications list-tags-for-resource \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
fe1efd35-EXAMPLE
```

输出：

```
{
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的创建通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-targets`

以下代码示例演示了如何使用 `list-targets`。

Amazon CLI

检索通知规则目标的列表

以下 `list-targets` 示例检索指定 Amazon 区域内的所有通知规则目标的列表。

```
aws codestar-notifications list-targets \
    --region us-east-1
```

输出：

```
{
    "Targets": [
        {
            "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MySNSTopicForNotificationRules",
            "TargetType": "SNS",
            "TargetStatus": "ACTIVE"
        },
        {
            "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MySNSTopicForNotificationsAboutMyDemoRepo",
            "TargetType": "SNS",
            "TargetStatus": "ACTIVE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的查看通知规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargets。

subscribe

以下代码示例演示了如何使用 subscribe。

Amazon CLI

向通知规则添加目标

以下 subscribe 示例将 Amazon SNS 主题添加为指定通知规则的目标。

```
aws codestar-notifications subscribe \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE \
    --target TargetType=SNS,TargetAddress=arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic
```

输出：

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的添加或删除 Amazon SNS 主题作为通知规则的目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Subscribe。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向通知规则添加标签

以下 tag-resource 示例向指定的通知规则添加了键名称为 Team、值为 Li_Juan 的标签。

```
aws codestar-notifications tag-resource \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
fe1efd35-EXAMPLE \
    --tags Team=Li_Juan
```

输出：

```
{
    "Tags": {
        "Team": "Li_Juan"
    }
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的创建通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

unsubscribe

以下代码示例演示了如何使用 unsubscribe。

Amazon CLI

从通知规则中移除目标

以下 unsubscribe 示例从指定的通知规则中移除了作为目标的 Amazon SNS 主题。

```
aws codestar-notifications unsubscribe \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE \
    --target TargetType=SNS,TargetAddress=arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic
```

输出：

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
    "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic"
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的添加或删除 Amazon SNS 主题作为通知规则的目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Unsubscribe。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从通知规则中移除标签

以下 untag-resource 示例从指定通知规则移除键名为 Team 的标签。

```
aws codestar-notifications untag-resource \
    --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
fe1efd35-EXAMPLE \
    --tag-keys Team
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的编辑通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-notification-rule

以下代码示例演示了如何使用 update-notification-rule。

Amazon CLI

更新通知规则

以下 update-notification-rule 示例使用名为 update.json 的 JSON 文件更新 Amazon 账户 123456789012 中名为 MyNotificationRule 的通知规则。

```
aws codestar-notifications update-notification-rule \
    --cli-input-json file://update.json
```

的内容update.json：

```
{
    "Name": "MyUpdatedNotificationRule",
    "EventTypeIds": [
        "codecommit-repository-branches-and-tags-created"
    ],
    "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",
    "Targets": [
        {
            "TargetType": "SNS",
            "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

输出：

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
```

```
}
```

有关更多信息，请参阅《Amazon 开发人员工具控制台用户指南》中的编辑通知规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateNotificationRule。

使用 Amazon CLI 的 CodeConnections 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 CodeConnections 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-connection`

以下代码示例演示了如何使用 `create-connection`。

Amazon CLI

创建连接

以下 `create-connection` 示例演示如何创建与第三方存储库的连接。此示例创建第三方提供程序为 Bitbucket 的连接。

默认情况下，通过 Amazon CLI 或 Amazon CloudFormation 创建的连接处于 Pending 状态。使用 CLI 或 Amazon CloudFormation 创建一个连接后，可使用控制台编辑该连接以使其状态为 Available。

```
aws codestar-connections create-connection \
    --provider-type Bitbucket \
```

```
    --connection-name MyConnection
```

输出：

```
{
    "ConnectionArn": "arn:aws:codestar-connections:us-
east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的创建连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnection。

create-host

以下代码示例演示了如何使用 create-host。

Amazon CLI

创建主机

以下 create-host 示例演示如何创建主机，来表示安装第三方提供程序的基础设施的端点。此示例创建了一台主机，其中安装的第三方提供程序是 GitHub Enterprise Server。

默认情况下，通过 Amazon CLI 创建的主机处于 Pending 状态。使用 CLI 创建主机后，可使用控制台或 CLI 设置主机以使其状态为 Available。

```
aws codestar-connections create-host \
    --name MyHost \
    --provider-type GitHubEnterpriseServer \
    --provider-endpoint "https://my-instance.dev"
```

输出：

```
{
    "HostArn": "arn:aws:codestar-connections:us-east-1:123456789012:host/My-
Host-28aef605"
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的创建主机（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHost。

delete-connection

以下代码示例演示了如何使用 delete-connection。

Amazon CLI

删除连接

以下 delete-connection 示例演示如何删除连接。

```
aws codestar-connections delete-connection \
    --connection-arn arn:aws:codestar-connections:us-west-2:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

此命令不生成任何输出。

有关更多信息，请参阅《开发人员工具控制台用户指南》中的删除连接（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnection。

delete-host

以下代码示例演示了如何使用 delete-host。

Amazon CLI

删除主机

下面的 delete-host 示例演示如何删除主机。必须先删除与主机关联的所有连接，然后才能删除主机。

```
aws codestar-connections delete-host \
    --host-arn "arn:aws:codestar-connections:us-east-1 :123456789012:host/My-Host-28aef605"
```

此命令不生成任何输出。

有关更多信息，请参阅《开发人员工具控制台用户指南》中的删除主机（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHost。

get-connection

以下代码示例演示了如何使用 get-connection。

Amazon CLI

获取有关连接的信息

以下 get-connection 示例演示有关连接的信息。

```
aws codestar-connections get-connection \
    --connection-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

输出：

```
{
    "Connection": {
        "ConnectionName": "MyConnection",
        "ConnectionArn": "arn:aws:codestar-connections:us-east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",
        "ProviderType": "Bitbucket",
        "OwnerAccountId": "123456789012",
        "ConnectionStatus": "AVAILABLE"
    }
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的查看连接的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnection。

get-host

以下代码示例演示了如何使用 get-host。

Amazon CLI

获取有关主机的信息

以下 get-host 示例演示有关主机的信息。

```
aws codestar-connections get-host \
```

```
    --host-arn arn:aws:codestar-connections:us-east-1:123456789012:host/
MyHost-28aef605
```

输出：

```
{
    "Name": "MyHost",
    "Status": "AVAILABLE",
    "ProviderType": "GitHubEnterpriseServer",
    "ProviderEndpoint": "https://test-instance-1.dev/"
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的查看主机的详细信息（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetHost。

list-connections

以下代码示例演示了如何使用 list-connections。

Amazon CLI

列出连接

以下 list-connections 示例检索您账户中 Bitbucket 提供程序类型的所有连接的列表：

```
aws codestar-connections list-connections \
--provider-type Bitbucket \
--max-results 5 \
--next-token: next-token
```

输出：

```
{
    "Connections": [
        {
            "ConnectionName": "my-connection",
            "ProviderType": "Bitbucket",
            "Status": "PENDING",
            "ARN": "arn:aws:codestar-connections:us-east-1:123456789012:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",
            "OwnerAccountId": "123456789012"
```

```
        },
        {
            "ConnectionName": "my-other-connection",
            "ProviderType": "Bitbucket",
            "Status": "AVAILABLE",
            "ARN": "arn:aws:codestar-connections:us-east-1:123456789012:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",
            "OwnerAccountId": "123456789012"
        },
    ],
    "NextToken": "next-token"
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的列出连接（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConnections。

list-hosts

以下代码示例演示了如何使用 `list-hosts`。

Amazon CLI

列出主机

以下 `list-hosts` 示例检索您账户中所有主机的列表。

```
aws codestar-connections list-hosts
```

输出：

```
{
    "Hosts": [
        {
            "Name": "My-Host",
            "HostArn": "arn:aws:codestar-connections:us-east-1:123456789012:host/My-
Host-28aef605",
            "ProviderType": "GitHubEnterpriseServer",
            "ProviderEndpoint": "https://my-instance.test.dev",
            "Status": "AVAILABLE"
        }
    ]
```

```
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的列出主机（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHosts。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出标签

以下 list-tags-for-resource 示例检索附加到指定连接资源的所有标签的列表。

```
aws codestar-connections list-tags-for-resource \
    --resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

输出：

```
{
    "Tags": [
        {
            "Key": "Project",
            "Value": "ProjectA"
        },
        {
            "Key": "ReadOnly",
            "Value": "true"
        }
    ]
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的查看连接资源的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例将一组提供的标签与连接关联。使用此命令添加或编辑标签。

```
aws codestar-connections tag-resource \
    --resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f \
    --tags Key=Project,Value=ProjectA Key=IscontainerBased,Value=true
```

此命令不生成任何输出。

有关更多信息，请参阅《开发人员工具控制台用户指南》中的向连接资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除连接资源的 Amazon 标签

以下 untag-resource 移除指定资源的标签。

```
aws codestar-connections untag-resource \
    --resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f \
    --tag-keys Project ReadOnly
```

输出：

```
{
    "Tags": []
}
```

有关更多信息，请参阅《开发人员工具控制台用户指南》中的移除连接资源的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Amazon Cognito Identity 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Cognito Identity 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-identity-pool`

以下代码示例演示了如何使用 `create-identity-pool`。

Amazon CLI

创建带有 Cognito 身份池提供者的身份池

此示例创建名为 MyIdentityPool 的身份池。它有一个 Cognito 身份池提供者。不允许使用未经身份验证的身份。

命令:

```
aws cognito-identity create-identity-pool --identity-pool-
name MyIdentityPool --no-allow-unauthenticated-identities --cognito-
identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
west-2_aaaaaaaaa",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

输出：

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
```

```
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIdentityPool。

delete-identities

以下代码示例演示了如何使用 delete-identities。

Amazon CLI

删除身份池

此示例删除身份池。

命令:

```
aws cognito-identity delete-identity-pool --identity-ids-to-delete "us-
west-2:11111111-1111-1111-1111-111111111111"
```

输出：

```
{
  "UnprocessedIdentityIds": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIdentities。

delete-identity-pool

以下代码示例演示了如何使用 delete-identity-pool。

Amazon CLI

删除身份池

以下 delete-identity-pool 示例删除指定身份池。

命令:

```
aws cognito-identity delete-identity-pool \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIdentityPool。

describe-identity-pool

以下代码示例演示了如何使用 describe-identity-pool。

Amazon CLI

描述身份池

此示例描述身份池。

命令:

```
aws cognito-identity describe-identity-pool --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

输出：

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIdentityPool。

get-identity-pool-roles

以下代码示例演示了如何使用 get-identity-pool-roles。

Amazon CLI

获取身份池角色

此示例获取身份池角色。

命令:

```
aws cognito-identity get-identity-pool-roles --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

输出：

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "Roles": {
      "authenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role",
      "unauthenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolUnauth_Role"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIdentityPoolRoles。

list-identity-pools

以下代码示例演示了如何使用 list-identity-pools。

Amazon CLI

列出身份池

此示例列出身份池。最多可列出 20 个身份。

命令:

```
aws cognito-identity list-identity-pools --max-results 20
```

输出：

```
{
  "IdentityPools": [
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "MyIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "AnotherIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "IdentityPoolRegionA"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIdentityPools。

set-identity-pool-roles

以下代码示例演示了如何使用 set-identity-pool-roles。

Amazon CLI

设置身份池角色

以下 set-identity-pool-roles 示例设置身份池角色。

```
aws cognito-identity set-identity-pool-roles \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" \
    --roles authenticated="arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetIdentityPoolRoles。

update-identity-pool

以下代码示例演示了如何使用 update-identity-pool。

Amazon CLI

更新身份池

此示例更新身份池。它将名称设置为 MyIdentityPool。它添加 Cognito 作为身份池提供者。它不允许使用未经身份验证的身份。

命令:

```
aws cognito-identity update-identity-pool --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" --identity-pool-name "MyIdentityPool" --no-allow-unauthenticated-identities --cognito-identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-west-2_111111111",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

输出：

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIdentityPool。

使用 Amazon CLI 的 Amazon Cognito 身份提供者示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Cognito 身份提供者结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-custom-attributes

以下代码示例演示了如何使用 add-custom-attributes。

Amazon CLI

添加自定义属性

此示例将自定义属性 CustomAttr1 添加到用户池。它是 String 类型，最少需要 1 个字符，最多 15 个。但其并非必要项目。

命令:

```
aws cognito-idp add-custom-attributes --user-pool-id us-west-2_aaaaaaaaa --custom-
attributes
 Name="CustomAttr1",AttributeDataType="String",DeveloperOnlyAttribute=false,Required=false,S
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddCustomAttributes。

admin-add-user-to-group

以下代码示例演示了如何使用 admin-add-user-to-group。

Amazon CLI

将用户添加到组

此示例将用户 Jane 添加到 MyGroup 组。

命令:

```
aws cognito-idp admin-add-user-to-group --user-pool-id us-west-2_aaaaaaaaa --
username Jane --group-name MyGroup
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminAddUserToGroup。

admin-confirm-sign-up

以下代码示例演示了如何使用 admin-confirm-sign-up。

Amazon CLI

确认用户注册

此示例确认用户 jane@example.com。

命令:

```
aws cognito-idp admin-confirm-sign-up --user-pool-id us-west-2_aaaaaaaaa --
username jane@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminConfirmSignUp。

admin-create-user

以下代码示例演示了如何使用 admin-create-user。

Amazon CLI

创建用户

以下的 admin-create-user 示例创建具有指定设置电子邮件地址和电话号码的用户。

```
aws cognito-idp admin-create-user \
    --user-pool-id us-west-2_aaaaaaaaa \
    --username diego \
    --user-attributes Name=email,Value=diego@example.com
 Name=phone_number,Value="+15555551212" \
    --message-action SUPPRESS
```

输出：

```
{
    "User": {
        "Username": "diego",
        "Attributes": [
            {
```

```
                "Name": "sub",
                "Value": "7325c1de-b05b-4f84-b321-9adc6e61f4a2"
            },
            {
                "Name": "phone_number",
                "Value": "+15555551212"
            },
            {
                "Name": "email",
                "Value": "diego@example.com"
            }
        ],
        "UserCreateDate": 1548099495.428,
        "UserLastModifiedDate": 1548099495.428,
        "Enabled": true,
        "UserStatus": "FORCE_CHANGE_PASSWORD"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminCreateUser。

admin-delete-user-attributes

以下代码示例演示了如何使用 admin-delete-user-attributes。

Amazon CLI

删除用户属性

此示例删除用户 diego@example.com 的自定义属性 CustomAttr1。

命令:

```
aws cognito-idp admin-delete-user-attributes --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com --user-attribute-names "custom:CustomAttr1"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminDeleteUserAttributes。

admin-delete-user

以下代码示例演示了如何使用 admin-delete-user。

Amazon CLI

删除用户

此示例删除一个用户。

命令:

```
aws cognito-idp admin-delete-user --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminDeleteUser。

admin-disable-provider-for-user

以下代码示例演示了如何使用 admin-disable-provider-for-user。

Amazon CLI

取消联合用户与本地用户配置文件的关联

以下 admin-disable-provider-for-user 示例断开 Google 用户与其关联的本地配置文件的连接。

```
aws cognito-idp admin-disable-provider-for-user \
    --user-pool-id us-west-2_EXAMPLE \
    --
user ProviderAttributeName=Cognito_Subject,ProviderAttributeValue=0000000000000000,ProviderN
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Linking federated users to an existing user profile。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminDisableProviderForUser。

admin-disable-user

以下代码示例演示了如何使用 admin-disable-user。

Amazon CLI

阻止用户登录

以下 admin-disable-user 示例阻止用户 diego@example.com 登录。

```
aws cognito-idp admin-disable-user \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing users。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminDisableUser。

admin-enable-user

以下代码示例演示了如何使用 admin-enable-user。

Amazon CLI

允许用户登录

以下 admin-enable-user 示例允许用户 diego@example.com 登录。

```
aws cognito-idp admin-enable-user \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing users。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminEnableUser。

admin-forget-device

以下代码示例演示了如何使用 admin-forget-device。

Amazon CLI

忘记设备

此示例忘记用户名为 jane@example.com 的设备

命令:

```
aws cognito-idp admin-forget-device --user-pool-id us-west-2_aaaaaaaaa --
username jane@example.com --device-key us-west-2_abcd_1234-5678
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminForgetDevice。

admin-get-device

以下代码示例演示了如何使用 admin-get-device。

Amazon CLI

获取设备

以下 admin-get-device 示例为用户 diego 显示一台设备。

```
aws cognito-idp admin-get-device \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego \
    --device-key us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Device": {
        "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "DeviceAttributes": [
            {
                "Name": "device_status",
                "Value": "valid"
            },
            {
                "Name": "device_name",
                "Value": "MyDevice"
            },
            {
                "Name": "dev:device_arn",
                "Value": "arn:aws:cognito-idp:us-west-2:123456789012:owner/diego.us-
west-2_EXAMPLE/device/us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            },
            {
                "Name": "dev:device_owner",
                "Value": "diego.us-west-2_EXAMPLE"
            },
            {
                "Name": "last_ip_used",
```

```
                "Value": "192.0.2.1"
            },
            {
                "Name": "dev:device_remembered_status",
                "Value": "remembered"
            },
            {
                "Name": "dev:device_sdk",
                "Value": "aws-sdk"
            }
        ],
        "DeviceCreateDate": 1715100742.022,
        "DeviceLastModifiedDate": 1723233651.167,
        "DeviceLastAuthenticatedDate": 1715100742.0
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with user devices in your user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminGetDevice。

admin-get-user

以下代码示例演示了如何使用 admin-get-user。

Amazon CLI

获取用户

此示例获取有关用户名 jane@example.com 的信息。

命令:

```
aws cognito-idp admin-get-user --user-pool-id us-west-2_aaaaaaaaa --
username jane@example.com
```

输出：

```
{
  "Username": "4320de44-2322-4620-999b-5e2e1c8df013",
  "Enabled": true,
```

```
  "UserStatus": "FORCE_CHANGE_PASSWORD",
  "UserCreateDate": 1548108509.537,
  "UserAttributes": [
      {
          "Name": "sub",
          "Value": "4320de44-2322-4620-999b-5e2e1c8df013"
      },
      {
          "Name": "email_verified",
          "Value": "true"
      },
      {
          "Name": "phone_number_verified",
          "Value": "true"
      },
      {
          "Name": "phone_number",
          "Value": "+01115551212"
      },
      {
          "Name": "email",
          "Value": "jane@example.com"
      }
  ],
  "UserLastModifiedDate": 1548108509.537
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminGetUser。

admin-initiate-auth

以下代码示例演示了如何使用 admin-initiate-auth。

Amazon CLI

以管理员身份注册用户

以下 admin-initiate-auth 示例注册用户 diego@example.com。此示例还包括用于威胁防护的元数据和用于 Lambda 触发器的 ClientMetadata。已为用户配置 TOTP MFA，用户会收到质询，需要先提供来自其身份验证器应用程序的代码，之后才能完成身份验证。

```
aws cognito-idp admin-initiate-auth \
```

```
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --auth-flow ADMIN_USER_PASSWORD_AUTH \
    --auth-parameters USERNAME=diego@example.com,PASSWORD="My@Example
$Password3!",SECRET_HASH=ExampleEncodedClientIdSecretAndUsername= \
    --context-data="{\"EncodedData\":\"abc123example\",\"HttpHeaders\":
[{\"headerName\":\"UserAgent\",\"headerValue\":\"Mozilla/5.0 (Windows NT 6.1;
 Win64; x64; rv:47.0) Gecko/20100101 Firefox/47.0\"}],\"IpAddress\":\"192.0.2.1\",
\"ServerName\":\"example.com\",\"ServerPath\":\"/login\"}" \
    --client-metadata="{\"MyExampleKey\": \"MyExampleValue\"}"
```

输出：

```
{
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeExample...",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "MyAuthenticatorApp",
        "USER_ID_FOR_SRP": "diego@example.com"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Admin authentication flow。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 AdminInitiateAuth。

admin-link-provider-for-user

以下代码示例演示了如何使用 admin-link-provider-for-user。

Amazon CLI

将本地用户与联合用户相关联

以下 admin-link-provider-for-user 示例将本地用户 diego 与使用 Google 进行联合登录的用户相关联。

```
aws cognito-idp admin-link-provider-for-user \
    --user-pool-id us-west-2_EXAMPLE \
    --destination-user ProviderName=Cognito,ProviderAttributeValue=diego \
    --source-
user ProviderAttributeName=Cognito_Subject,ProviderAttributeValue=0000000000000000,ProviderN
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Linking federated users to an existing user profile。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminLinkProviderForUser。

admin-list-devices

以下代码示例演示了如何使用 admin-list-devices。

Amazon CLI

列出用户的设备

以下 admin-list-devices 示例为用户 diego 列出设备。

```
aws cognito-idp admin-list-devices \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego \
    --limit 1
```

输出：

```
{
    "Devices": [
        {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceAttributes": [
                {
                    "Name": "device_status",
                    "Value": "valid"
                },
                {
                    "Name": "device_name",
                    "Value": "MyDevice"
                },
                {
                    "Name": "dev:device_arn",
                    "Value": "arn:aws:cognito-idp:us-west-2:123456789012:owner/
diego.us-west-2_EXAMPLE/device/us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                },
                {
```

```
                    "Name": "dev:device_owner",
                    "Value": "diego.us-west-2_EXAMPLE"
                },
                {
                    "Name": "last_ip_used",
                    "Value": "192.0.2.1"
                },
                {
                    "Name": "dev:device_remembered_status",
                    "Value": "remembered"
                },
                {
                    "Name": "dev:device_sdk",
                    "Value": "aws-sdk"
                }
            ],
            "DeviceCreateDate": 1715100742.022,
            "DeviceLastModifiedDate": 1723233651.167,
            "DeviceLastAuthenticatedDate": 1715100742.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with user devices in your user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminListDevices。

admin-list-groups-for-user

以下代码示例演示了如何使用 admin-list-groups-for-user。

Amazon CLI

列出用户的组

此示例列出用户名为 jane@example.com 的组。

命令:

```
aws cognito-idp admin-list-groups-for-user --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com
```

输出：

```
{
  "Groups": [
      {
          "Description": "Sample group",
          "Precedence": 1,
          "LastModifiedDate": 1548097827.125,
          "RoleArn": "arn:aws:iam::111111111111:role/SampleRole",
          "GroupName": "SampleGroup",
          "UserPoolId": "us-west-2_aaaaaaaaa",
          "CreationDate": 1548097827.125
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminListGroupsForUser。

admin-list-user-auth-events

以下代码示例演示了如何使用 admin-list-user-auth-events。

Amazon CLI

列出用户的授权事件

以下 admin-list-user-auth-events 示例列出用户 diego 的最新用户活动日志事件。

```
aws cognito-idp admin-list-user-auth-events \
    --user-pool-id us-west-2_ywDJHlIfU \
    --username brcotter+050123 \
    --max-results 1
```

输出：

```
{
    "AuthEvents": [
        {
            "EventId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "EventType": "SignIn",
            "CreationDate": 1726694203.495,
            "EventResponse": "InProgress",
```

```
            "EventRisk": {
                "RiskDecision": "AccountTakeover",
                "RiskLevel": "Medium",
                "CompromisedCredentialsDetected": false
            },
            "ChallengeResponses": [
                {
                    "ChallengeName": "Password",
                    "ChallengeResponse": "Success"
                }
            ],
            "EventContextData": {
                "IpAddress": "192.0.2.1",
                "City": "Seattle",
                "Country": "United States"
            }
        }
    ],
    "NextToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222#2024-09-18T21:16:43.495Z"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Viewing and exporting user event history。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminListUserAuthEvents。

admin-remove-user-from-group

以下代码示例演示了如何使用 admin-remove-user-from-group。

Amazon CLI

从组中移除用户

此示例从 SampleGroup 中移除 jane@example.com。

命令:

```
aws cognito-idp admin-remove-user-from-group --user-pool-id us-west-2_aaaaaaaaa --
username jane@example.com --group-name SampleGroup
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminRemoveUserFromGroup。

admin-reset-user-password

以下代码示例演示了如何使用 admin-reset-user-password。

Amazon CLI

重置用户密码

此示例重置 diego@example.com 的密码。

命令:

```
aws cognito-idp admin-reset-user-password --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminResetUserPassword。

admin-respond-to-auth-challenge

以下代码示例演示了如何使用 admin-respond-to-auth-challenge。

Amazon CLI

响应身份验证质询

可以通过多种方式响应不同的身份验证质询，具体取决于身份验证流程、用户池配置和用户设置。以下 admin-respond-to-auth-challenge 示例提供 diego@example.com 的 TOTP MFA 代码并完成登录。此用户池已启用设备记忆功能，因此身份验证结果还将返回新的设备密钥。

```
aws cognito-idp admin-respond-to-auth-challenge \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-responses USERNAME=diego@example.com,SOFTWARE_TOKEN_MFA_CODE=000000
 \
    --session AYABeExample...
```

输出：

```
{
    "ChallengeParameters": {},
```

```
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-ExAmPlE1"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Admin authentication flow。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminRespondToAuthChallenge。

admin-set-user-mfa-preference

以下代码示例演示了如何使用 admin-set-user-mfa-preference。

Amazon CLI

设置用户 MFA 首选项

此示例设置用户名 diego@example.com 的 SMS MFA 首选项。

命令:

```
aws cognito-idp admin-set-user-mfa-preference --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com --sms-mfa-settings Enabled=false,PreferredMfa=false
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminSetUserMfaPreference。

admin-set-user-password

以下代码示例演示了如何使用 admin-set-user-password。

Amazon CLI

以管理员身份设置用户密码

以下 admin-set-user-password 示例永久设置 diego@example.com 的密码。

```
aws cognito-idp admin-set-user-password \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com \
    --password MyExamplePassword1! \
    --permanent
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Passwords, password recovery, and password policies。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminSetUserPassword。

admin-set-user-settings

以下代码示例演示了如何使用 admin-set-user-settings。

Amazon CLI

设置用户设置

此示例将用户名 diego@example.com 的 MFA 发送首选项设置为 EMAIL。

命令:

```
aws cognito-idp admin-set-user-settings --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com --mfa-options DeliveryMedium=EMAIL
```

- 有关 API 详细信息，请参阅《Amazon CLI API 参考》中的 AdminSetUserSettings。

admin-update-auth-event-feedback

以下代码示例演示了如何使用 admin-update-auth-event-feedback。

Amazon CLI

提供授权事件的反馈

此示例将由 event-id 标识的授权事件的反馈值设置为“有效”。

命令:

```
aws cognito-idp admin-update-auth-event-feedback --user-pool-id us-west-2_aaaaaaaaa
 --username diego@example.com --event-id c2c2cf89-c0d3-482d-aba6-99d78a5b0bfe --
feedback-value Valid
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminUpdateAuthEventFeedback。

admin-update-device-status

以下代码示例演示了如何使用 admin-update-device-status。

Amazon CLI

更新设备状态

此示例将由 device-key 标识的设备的设备记忆状态设置为 not_remembered。

命令:

```
aws cognito-idp admin-update-device-status --user-pool-id us-west-2_aaaaaaaaa
 --username diego@example.com --device-key xxxx  --device-remembered-
status not_remembered
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminUpdateDeviceStatus。

admin-update-user-attributes

以下代码示例演示了如何使用 admin-update-user-attributes。

Amazon CLI

更新用户属性

此示例更新用户 diego@example.com 的自定义用户属性 CustomAttr1。

命令:

```
aws cognito-idp admin-update-user-attributes --user-pool-id us-
west-2_aaaaaaaaa --username diego@example.com --user-attributes
 Name="custom:CustomAttr1",Value="Purple"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdminUpdateUserAttributes。

admin-user-global-sign-out

以下代码示例演示了如何使用 admin-user-global-sign-out。

Amazon CLI

以管理员身份注销用户

以下 admin-user-global-sign-out 示例注销用户 diego@example.com。

```
aws cognito-idp admin-user-global-sign-out \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Authentication with a user pool。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AdminUserGlobalSignOut。

associate-software-token

以下代码示例演示了如何使用 associate-software-token。

Amazon CLI

为 MFA 身份验证器应用程序生成私有密钥

以下 associate-software-token 示例为已注册并收到访问令牌的用户生成 TOTP 私有密钥。可手动将生成的私有密钥输入到身份验证器应用程序中，或者应用程序可以将私有密钥呈现为用户可扫描的二维码。

```
aws cognito-idp associate-software-token \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "SecretCode": "QWERTYUIOP123456EXAMPLE"
```

```
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 TOTP software token MFA。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateSoftwareToken。

change-password

以下代码示例演示了如何使用 change-password。

Amazon CLI

更改密码

此示例更改密码。

命令:

```
aws cognito-idp change-password --previous-password OldPassword --proposed-
password NewPassword --access-token ACCESS_TOKEN
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangePassword。

confirm-device

以下代码示例演示了如何使用 confirm-device。

Amazon CLI

确认用户设备

以下 confirm-device 示例为当前用户添加新的记忆设备。

```
aws cognito-idp confirm-device \
   --access-token eyJra456defEXAMPLE \
   --device-key us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
   --device-secret-verifier-
config PasswordVerifier=TXlWZXJpZmllclN0cmluZw,Salt=TXlTUlBTYWx0
```

输出：

```
{
    "UserConfirmationNecessary": false
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with user devices in your user pool。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ConfirmDevice。

`confirm-forgot-password`

以下代码示例演示了如何使用 `confirm-forgot-password`。

Amazon CLI

确认忘记的密码

此示例确认用户名 diego@example.com 的已忘记密码。

命令:

```
aws cognito-idp confirm-forgot-password --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --password PASSWORD --confirmation-code CONF_CODE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmForgotPassword。

`confirm-sign-up`

以下代码示例演示了如何使用 `confirm-sign-up`。

Amazon CLI

确认注册

此示例确认用户名 diego@example.com 的注册。

命令:

```
aws cognito-idp confirm-sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --confirmation-code CONF_CODE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmSignUp。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

创建组

此示例创建一个带有说明的组。

命令:

```
aws cognito-idp create-group --user-pool-id us-west-2_aaaaaaaaa --group-
name MyNewGroup --description "New group."
```

输出：

```
{
  "Group": {
      "GroupName": "MyNewGroup",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "Description": "New group.",
      "LastModifiedDate": 1548270073.795,
      "CreationDate": 1548270073.795
  }
}
```

创建具有角色和优先级的组

此示例创建一个带有说明的组。它还包括“角色”和“优先级”。

命令:

```
aws cognito-idp create-group --user-pool-id us-west-2_aaaaaaaaa --group-
name MyNewGroupWithRole --description "New group with a role." --role-
arn arn:aws:iam::111111111111:role/MyNewGroupRole --precedence 2
```

输出：

```
{
```

```
  "Group": {
      "GroupName": "MyNewGroupWithRole",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "Description": "New group with a role.",
      "RoleArn": "arn:aws:iam::111111111111:role/MyNewGroupRole",
      "Precedence": 2,
      "LastModifiedDate": 1548270211.761,
      "CreationDate": 1548270211.761
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-identity-provider

以下代码示例演示了如何使用 create-identity-provider。

Amazon CLI

示例 1：使用元数据 URL 创建用户池 SAML 身份提供者（IdP）

以下 create-identity-provider 示例使用来自公共 URL 的元数据、属性映射和两个标识符创建新的 SAML IdP。

```
aws cognito-idp create-identity-provider \
    --user-pool-id us-west-2_EXAMPLE \
    --provider-name MySAML \
    --provider-type SAML \
    --provider-
details IDPInit=true,IDPSignout=true,EncryptedResponses=true,MetadataURL=https://
auth.example.com/sso/saml/metadata,RequestSigningAlgorithm=rsa-sha256 \
    --attribute-mapping email=emailaddress,phone_number=phone,custom:111=department
 \
    --idp-identifiers CorpSAML WestSAML
```

输出：

```
{
    "IdentityProvider": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ProviderName": "MySAML",
        "ProviderType": "SAML",
```

```
        "ProviderDetails": {
            "ActiveEncryptionCertificate": "MIICvTCCAaEXAMPLE",
            "EncryptedResponses": "true",
            "IDPInit": "true",
            "IDPSignout": "true",
            "MetadataURL": "https://auth.example.com/sso/saml/metadata",
            "RequestSigningAlgorithm": "rsa-sha256",
            "SLORedirectBindingURI": "https://auth.example.com/slo/saml",
            "SSORedirectBindingURI": "https://auth.example.com/sso/saml"
        },
        "AttributeMapping": {
            "custom:111": "department",
            "emailaddress": "email",
            "phone": "phone_number"
        },
        "IdpIdentifiers": [
            "CorpSAML",
            "WestSAML"
        ],
        "LastModifiedDate": 1726853833.977,
        "CreationDate": 1726853833.977
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding user pool sign-in through a third party。

示例 2：使用元数据文件创建用户池 SAML 身份提供者（IdP）

以下 `create-identity-provider` 示例使用来自文件的元数据、属性映射和两个标识符创建新的 SAML IdP。对于不同的操作系统，`--provider-details` 参数中的文件语法会有所不同。可以非常轻松地为此操作创建 JSON 输入文件。

```
aws cognito-idp create-identity-provider \
    --cli-input-json file://.\SAML-identity-provider.json
```

的内容`SAML-identity-provider.json`：

```
{
    "AttributeMapping": {
        "email" : "idp_email",
        "email_verified" : "idp_email_verified"
```

```
    },
    "IdpIdentifiers": [ "platform" ],
    "ProviderDetails": {
        "MetadataFile": "<md:EntityDescriptor xmlns:md=
\"urn:oasis:names:tc:SAML:2.0:metadata\" entityID=\"http://www.example.com/
sso\"><md:IDPSSODescriptor WantAuthnRequestsSigned=\"false\"
 protocolSupportEnumeration=\"urn:oasis:names:tc:SAML:2.0:protocol
\"><md:KeyDescriptor use=\"signing\"><ds:KeyInfo xmlns:ds=\"http://
www.w3.org/2000/09/xmldsig#
\"><ds:X509Data><ds:X509Certificate>[IDP_CERTIFICATE_DATA]</ds:X509Certificate></
ds:X509Data></ds:KeyInfo></md:KeyDescriptor><md:SingleLogoutService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST\" Location=
\"https://www.example.com/slo/saml\"/><md:SingleLogoutService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
Redirect\" Location=\"https://www.example.com/slo/saml\"/
><md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</
md:NameIDFormat><md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress</md:NameIDFormat><md:SingleSignOnService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST\" Location=
\"https://www.example.com/sso/saml\"/><md:SingleSignOnService Binding=
\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect\" Location=\"https://
www.example.com/sso/saml\"/></md:IDPSSODescriptor></md:EntityDescriptor>",
        "IDPSignout" : "true",
        "RequestSigningAlgorithm" : "rsa-sha256",
        "EncryptedResponses" : "true",
        "IDPInit" : "true"
    },
    "ProviderName": "MySAML2",
    "ProviderType": "SAML",
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

输出：

```
{
    "IdentityProvider": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ProviderName": "MySAML2",
        "ProviderType": "SAML",
        "ProviderDetails": {
            "ActiveEncryptionCertificate":
 "[USER_POOL_ENCRYPTION_CERTIFICATE_DATA]",
            "EncryptedResponses": "true",
```

```
            "IDPInit": "true",
            "IDPSignout": "true",
            "MetadataFile": "<md:EntityDescriptor xmlns:md=
\"urn:oasis:names:tc:SAML:2.0:metadata\" entityID=\"http://www.example.com/
sso\"><md:IDPSSODescriptor WantAuthnRequestsSigned=\"false\"
 protocolSupportEnumeration=\"urn:oasis:names:tc:SAML:2.0:protocol
\"><md:KeyDescriptor use=\"signing\"><ds:KeyInfo xmlns:ds=\"http://
www.w3.org/2000/09/xmldsig#
\"><ds:X509Data><ds:X509Certificate>[IDP_CERTIFICATE_DATA]</ds:X509Certificate></
ds:X509Data></ds:KeyInfo></md:KeyDescriptor><md:SingleLogoutService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST\" Location=
\"https://www.example.com/slo/saml\"/><md:SingleLogoutService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
Redirect\" Location=\"https://www.example.com/slo/saml\"/
><md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</
md:NameIDFormat><md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress</md:NameIDFormat><md:SingleSignOnService
 Binding=\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST\" Location=
\"https://www.example.com/sso/saml\"/><md:SingleSignOnService Binding=
\"urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect\" Location=\"https://
www.example.com/sso/saml\"/></md:IDPSSODescriptor></md:EntityDescriptor>",
            "RequestSigningAlgorithm": "rsa-sha256",
            "SLORedirectBindingURI": "https://www.example.com/slo/saml",
            "SSORedirectBindingURI": "https://www.example.com/sso/saml"
        },
        "AttributeMapping": {
            "email": "idp_email",
            "email_verified": "idp_email_verified"
        },
        "IdpIdentifiers": [
            "platform"
        ],
        "LastModifiedDate": 1726855290.731,
        "CreationDate": 1726855290.731
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding user pool sign-in through a third party。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateIdentityProvider。

create-resource-server

以下代码示例演示了如何使用 create-resource-server。

Amazon CLI

创建用户池客户端

以下 create-resource-server 示例创建具有自定义范围的新资源服务器。

```
aws cognito-idp create-resource-server \
    --user-pool-id us-west-2_EXAMPLE \
    --identifier solar-system-data \
    --name "Solar system object tracker" \
    --scopes ScopeName=sunproximity.read,ScopeDescription="Distance in AU from Sol"
 ScopeName=asteroids.add,ScopeDescription="Enter a new asteroid"
```

输出：

```
{
    "ResourceServer": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "Identifier": "solar-system-data",
        "Name": "Solar system object tracker",
        "Scopes": [
            {
                "ScopeName": "sunproximity.read",
                "ScopeDescription": "Distance in AU from Sol"
            },
            {
                "ScopeName": "asteroids.add",
                "ScopeDescription": "Enter a new asteroid"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Scopes, M2M, and APIs with resource servers。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateResourceServer。

create-user-import-job

以下代码示例演示了如何使用 create-user-import-job。

Amazon CLI

创建用户导入任务

此示例创建一个名为 MyImportJob 的用户导入任务。

有关导入用户的更多信息，请参阅“从 CSV 文件将用户导入用户池”。

命令:

```
aws cognito-idp create-user-import-job --user-pool-id us-west-2_aaaaaaaaa --
job-name MyImportJob --cloud-watch-logs-role-arn arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole
```

输出：

```
{
  "UserImportJob": {
      "JobName": "MyImportJob",
      "JobId": "import-qQ0DCt2fRh",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "PreSignedUrl": "PRE_SIGNED_URL",
      "CreationDate": 1548271795.471,
      "Status": "Created",
      "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
      "ImportedUsers": 0,
      "SkippedUsers": 0,
      "FailedUsers": 0
  }
}
```

使用预签名 URL 上传带 curl 的 .csv 文件：

命令:

```
curl -v -T "PATH_TO_CSV_FILE" -H "x-amz-server-side-encryption:aws:kms"
 "PRE_SIGNED_URL"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUserImportJob。

create-user-pool-client

以下代码示例演示了如何使用 create-user-pool-client。

Amazon CLI

创建用户池客户端

以下 create-user-pool-client 示例创建新的用户池客户端，该客户端具有客户端密钥、显式读取和写入属性、使用用户名密码和 SRP 流程进行登录、使用三个 IdP 进行登录、访问一部分 OAuth 范围、PinPoint 分析和延长的身份验证会话有效期。

```
aws cognito-idp create-user-pool-client \
    --user-pool-id us-west-2_EXAMPLE \
    --client-name MyTestClient \
    --generate-secret \
    --refresh-token-validity 10 \
    --access-token-validity 60 \
    --id-token-validity 60 \
    --token-validity-units AccessToken=minutes,IdToken=minutes,RefreshToken=days \
    --read-attributes email phone_number email_verified phone_number_verified \
    --write-attributes email phone_number \
    --explicit-auth-
flows ALLOW_USER_PASSWORD_AUTH ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH \
    --supported-identity-providers Google Facebook MyOIDC \
    --callback-urls https://www.amazon.com https://example.com http://
localhost:8001 myapp://example \
    --allowed-o-auth-flows code implicit \
    --allowed-o-auth-scopes openid profile aws.cognito.signin.user.admin solar-
system-data/asteroids.add \
    --allowed-o-auth-flows-user-pool-client \
    --analytics-configuration ApplicationArn=arn:aws:mobiletargeting:us-
west-2:767671399759:apps/thisisanexamplepinpointapplicationid,UserDataShared=TRUE \
    --prevent-user-existence-errors ENABLED \
    --enable-token-revocation \
    --enable-propagate-additional-user-context-data \
    --auth-session-validity 4
```

输出：

```
{
    "UserPoolClient": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ClientName": "MyTestClient",
        "ClientId": "123abc456defEXAMPLE",
        "ClientSecret": "this1234is5678my91011example1213client1415secret",
        "LastModifiedDate": 1726788459.464,
        "CreationDate": 1726788459.464,
        "RefreshTokenValidity": 10,
        "AccessTokenValidity": 60,
        "IdTokenValidity": 60,
        "TokenValidityUnits": {
            "AccessToken": "minutes",
            "IdToken": "minutes",
            "RefreshToken": "days"
        },
        "ReadAttributes": [
            "email_verified",
            "phone_number_verified",
            "phone_number",
            "email"
        ],
        "WriteAttributes": [
            "phone_number",
            "email"
        ],
        "ExplicitAuthFlows": [
            "ALLOW_USER_PASSWORD_AUTH",
            "ALLOW_USER_SRP_AUTH",
            "ALLOW_REFRESH_TOKEN_AUTH"
        ],
        "SupportedIdentityProviders": [
            "Google",
            "MyOIDC",
            "Facebook"
        ],
        "CallbackURLs": [
            "https://example.com",
            "https://www.amazon.com",
            "myapp://example",
            "http://localhost:8001"
        ],
        "AllowedOAuthFlows": [
```

```
            "implicit",
            "code"
        ],
        "AllowedOAuthScopes": [
            "aws.cognito.signin.user.admin",
            "openid",
            "profile",
            "solar-system-data/asteroids.add"
        ],
        "AllowedOAuthFlowsUserPoolClient": true,
        "AnalyticsConfiguration": {
            "ApplicationArn": "arn:aws:mobiletargeting:us-west-2:123456789012:apps/
thisisanexamplepinpointapplicationid",
            "RoleArn": "arn:aws:iam::123456789012:role/aws-service-role/cognito-
idp.amazonaws.com/AWSServiceRoleForAmazonCognitoIdp",
            "UserDataShared": true
        },
        "PreventUserExistenceErrors": "ENABLED",
        "EnableTokenRevocation": true,
        "EnablePropagateAdditionalUserContextData": true,
        "AuthSessionValidity": 4
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Application-specific settings with app clients。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUserPoolClient。

create-user-pool-domain

以下代码示例演示了如何使用 create-user-pool-domain。

Amazon CLI

示例 1：创建用户池域

以下 create-user-pool-domain 示例创建新的自定义域。

```
aws cognito-idp create-user-pool-domain \
    --user-pool-id us-west-2_EXAMPLE \
    --domain auth.example.com \
```

```
    --custom-domain-config CertificateArn=arn:aws:acm:us-
east-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

输出：

```
{
    "CloudFrontDomain": "example1domain.cloudfront.net"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Configuring a user pool domain。

示例 2：创建用户池域

以下 `create-user-pool-domain` 示例使用服务拥有的前缀创建新域。

```
aws cognito-idp create-user-pool-domain \
    --user-pool-id us-west-2_EXAMPLE2 \
    --domain mydomainprefix
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Configuring a user pool domain。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUserPoolDomain。

create-user-pool

以下代码示例演示了如何使用 `create-user-pool`。

Amazon CLI

创建最低配置的用户池

此示例将使用默认值创建一个名为 MyUserPool 的用户池。没有必要的属性，也没有应用程序客户端。MFA 和高级安全功能已禁用。

命令:

```
aws cognito-idp create-user-pool --pool-name MyUserPool
```

输出：

```
{
  "UserPool": {
      "SchemaAttributes": [
          {
              "Name": "sub",
              "StringAttributeConstraints": {
                  "MinLength": "1",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": true,
              "AttributeDataType": "String",
              "Mutable": false
          },
          {
              "Name": "name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
              "AttributeDataType": "String",
              "Mutable": true
          },
          {
              "Name": "given_name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
              "AttributeDataType": "String",
              "Mutable": true
          },
          {
              "Name": "family_name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
```

```
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "middle_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "nickname",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "preferred_username",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "profile",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
```

```
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "picture",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "website",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "email",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "email_verified",
            "Mutable": true
```

```
        },
        {
            "Name": "gender",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "birthdate",
            "StringAttributeConstraints": {
                "MinLength": "10",
                "MaxLength": "10"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "zoneinfo",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "locale",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
```

```
        },
        {
            "Name": "phone_number",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "phone_number_verified",
            "Mutable": true
        },
        {
            "Name": "address",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "updated_at",
            "NumberAttributeConstraints": {
                "MinValue": "0"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "Number",
            "Mutable": true
        }
    ],
    "MfaConfiguration": "OFF",
    "Name": "MyUserPool",
    "LastModifiedDate": 1547833345.777,
```

```
      "AdminCreateUserConfig": {
          "UnusedAccountValidityDays": 7,
          "AllowAdminCreateUserOnly": false
      },
      "EmailConfiguration": {},
      "Policies": {
          "PasswordPolicy": {
              "RequireLowercase": true,
              "RequireSymbols": true,
              "RequireNumbers": true,
              "MinimumLength": 8,
              "RequireUppercase": true
          }
      },
      "CreationDate": 1547833345.777,
      "EstimatedNumberOfUsers": 0,
      "Id": "us-west-2_aaaaaaaaa",
      "LambdaConfig": {}
  }
}
```

创建具有两个必要属性的用户池

此示例创建一个用户池 MyUserPool。该池配置为接受电子邮件作为用户名属性。它还使用 Amazon Simple Email Service 将电子邮件源地址设置为经过验证的地址。

命令:

```
aws cognito-idp create-user-pool --pool-name MyUserPool --username-
attributes "email" --email-configuration=SourceArn="arn:aws:ses:us-
east-1:111111111111:identity/
jane@example.com",ReplyToEmailAddress="jane@example.com"
```

输出：

```
{
  "UserPool": {
      "SchemaAttributes": [
          {
              "Name": "sub",
              "StringAttributeConstraints": {
                  "MinLength": "1",
                  "MaxLength": "2048"
```

```
            },
            "DeveloperOnlyAttribute": false,
            "Required": true,
            "AttributeDataType": "String",
            "Mutable": false
        },
        {
            "Name": "name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "given_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "family_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "middle_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
```

```
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "nickname",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "preferred_username",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "profile",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "picture",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
```

```
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "website",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "email",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "email_verified",
            "Mutable": true
        },
        {
            "Name": "gender",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
```

```
            "Mutable": true
        },
        {
            "Name": "birthdate",
            "StringAttributeConstraints": {
                "MinLength": "10",
                "MaxLength": "10"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "zoneinfo",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "locale",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "phone_number",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
```

```
                "Mutable": true
            },
            {
                "AttributeDataType": "Boolean",
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "Name": "phone_number_verified",
                "Mutable": true
            },
            {
                "Name": "address",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "updated_at",
                "NumberAttributeConstraints": {
                    "MinValue": "0"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "Number",
                "Mutable": true
            }
        ],
        "MfaConfiguration": "OFF",
        "Name": "MyUserPool",
        "LastModifiedDate": 1547837788.189,
        "AdminCreateUserConfig": {
            "UnusedAccountValidityDays": 7,
            "AllowAdminCreateUserOnly": false
        },
        "EmailConfiguration": {
            "ReplyToEmailAddress": "jane@example.com",
            "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/
jane@example.com"
        },
        "Policies": {
```

```
            "PasswordPolicy": {
                "RequireLowercase": true,
                "RequireSymbols": true,
                "RequireNumbers": true,
                "MinimumLength": 8,
                "RequireUppercase": true
            }
        },
        "UsernameAttributes": [
            "email"
        ],
        "CreationDate": 1547837788.189,
        "EstimatedNumberOfUsers": 0,
        "Id": "us-west-2_aaaaaaaaa",
        "LambdaConfig": {}
    }
}
```

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUserPool。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除组

此示例删除组。

命令:

```
aws cognito-idp delete-group --user-pool-id us-west-2_aaaaaaaaa  --group-
name MyGroupName
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-identity-provider

以下代码示例演示了如何使用 delete-identity-provider。

Amazon CLI

删除身份提供者

此示例删除一个身份提供者。

命令:

```
aws cognito-idp delete-identity-provider --user-pool-id us-west-2_aaaaaaaaa --
provider-name Facebook
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIdentityProvider。

delete-resource-server

以下代码示例演示了如何使用 delete-resource-server。

Amazon CLI

删除资源服务器

此示例删除一个名为 weather.example.com 的资源服务器。

命令:

```
aws cognito-idp delete-resource-server --user-pool-id us-west-2_aaaaaaaaa --
identifier weather.example.com
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourceServer。

delete-user-attributes

以下代码示例演示了如何使用 delete-user-attributes。

Amazon CLI

删除用户属性

以下 delete-user-attributes 示例从当前已登录用户中删除自定义属性“custom:attribute”。

```
aws cognito-idp delete-user-attributes \
```

```
    --access-token ACCESS_TOKEN \
    --user-attribute-names "custom:department"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with user attributes。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserAttributes。

delete-user-pool-client

以下代码示例演示了如何使用 delete-user-pool-client。

Amazon CLI

删除用户池客户端

此示例删除一个用户池客户端。

命令:

```
aws cognito-idp delete-user-pool-client --user-pool-id us-west-2_aaaaaaaaa --client-id 38fjsnc484p94kpqsnet7mpld0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserPoolClient。

delete-user-pool-domain

以下代码示例演示了如何使用 delete-user-pool-domain。

Amazon CLI

删除用户池域

以下 delete-user-pool-domain 示例删除名为 my-domain 的用户池域。

```
aws cognito-idp delete-user-pool-domain \
    --user-pool-id us-west-2_aaaaaaaaa \
    --domain my-domain
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserPoolDomain。

delete-user-pool

以下代码示例演示了如何使用 delete-user-pool。

Amazon CLI

删除用户池

此示例使用用户池 ID us-west-2_aaaaaaaaa 删除用户池。

命令:

```
aws cognito-idp delete-user-pool --user-pool-id us-west-2_aaaaaaaaa
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserPool。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

此示例删除一个用户。

命令:

```
aws cognito-idp delete-user --access-token ACCESS_TOKEN
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-identity-provider

以下代码示例演示了如何使用 describe-identity-provider。

Amazon CLI

描述身份提供者

此示例描述一个名为 Facebook 的身份提供者。

命令:

```
aws cognito-idp describe-identity-provider --user-pool-id us-west-2_aaaaaaaaa --
provider-name Facebook
```

输出：

```
{
  "IdentityProvider": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "ProviderName": "Facebook",
      "ProviderType": "Facebook",
      "ProviderDetails": {
          "attributes_url": "https://graph.facebook.com/me?fields=",
          "attributes_url_add_attributes": "true",
          "authorize_scopes": myscope",
          "authorize_url": "https://www.facebook.com/v2.9/dialog/oauth",
          "client_id": "11111",
          "client_secret": "11111",
          "token_request_method": "GET",
          "token_url": "https://graph.facebook.com/v2.9/oauth/access_token"
      },
      "AttributeMapping": {
          "username": "id"
      },
      "IdpIdentifiers": [],
      "LastModifiedDate": 1548105901.736,
      "CreationDate": 1548105901.736
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIdentityProvider。

describe-resource-server

以下代码示例演示了如何使用 describe-resource-server。

Amazon CLI

描述资源服务器

此示例描述资源服务器 weather.example.com。

命令:

```
aws cognito-idp describe-resource-server --user-pool-id us-west-2_aaaaaaaaa --
identifier weather.example.com
```

输出：

```
{
  "ResourceServer": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "Identifier": "weather.example.com",
      "Name": "Weather",
      "Scopes": [
          {
              "ScopeName": "weather.update",
              "ScopeDescription": "Update weather forecast"
          },
          {
              "ScopeName": "weather.read",
              "ScopeDescription": "Read weather forecasts"
          },
          {
              "ScopeName": "weather.delete",
              "ScopeDescription": "Delete a weather forecast"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResourceServer。

describe-risk-configuration

以下代码示例演示了如何使用 describe-risk-configuration。

Amazon CLI

描述风险配置

此示例描述与池 us-west-2_aaaaaaaaa 相关的风险配置。

命令:

```
aws cognito-idp describe-risk-configuration --user-pool-id us-west-2_aaaaaaaaa
```

输出：


```
{
  "RiskConfiguration": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "CompromisedCredentialsRiskConfiguration": {
          "EventFilter": [
              "SIGN_IN",
              "SIGN_UP",
              "PASSWORD_CHANGE"
          ],
          "Actions": {
              "EventAction": "BLOCK"
          }
      },
      "AccountTakeoverRiskConfiguration": {
          "NotifyConfiguration": {
              "From": "diego@example.com",
              "ReplyTo": "diego@example.com",
              "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/
diego@example.com",
              "BlockEmail": {
                  "Subject": "Blocked sign-in attempt",
                  "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 blocked an unrecognized sign-in to your account with this information:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                  "TextBody": "We blocked an unrecognized sign-in to your account
 with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation:
 {city}, {country}\nIf this sign-in was not by you, you should change your password
 and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you,
 you can follow {one-click-link-valid} to let us know"
              },
              "NoActionEmail": {
                  "Subject": "New sign-in attempt",
                  "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 observed an unrecognized sign-in to your account with this information:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
```

```
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                    "TextBody": "We observed an unrecognized sign-in to your account
 with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation:
 {city}, {country}\nIf this sign-in was not by you, you should change your password
 and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you,
 you can follow {one-click-link-valid} to let us know"
                },
                "MfaEmail": {
                    "Subject": "New sign-in attempt",
                    "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML email
 context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We required
 you to use multi-factor authentication for the following sign-in attempt:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                    "TextBody": "We required you to use multi-factor authentication
 for the following sign-in attempt:\nTime: {login-time}\nDevice: {device-
name}\nLocation: {city}, {country}\nIf this sign-in was not by you, you should
 change your password and notify us by clicking on {one-click-link-invalid}\nIf this
 sign-in was by you, you can follow {one-click-link-valid} to let us know"
                }
            },
            "Actions": {
                "LowAction": {
                    "Notify": true,
                    "EventAction": "NO_ACTION"
                },
                "MediumAction": {
                    "Notify": true,
                    "EventAction": "MFA_IF_CONFIGURED"
                },
                "HighAction": {
                    "Notify": true,
                    "EventAction": "MFA_IF_CONFIGURED"
                }
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRiskConfiguration。

`describe-user-import-job`

以下代码示例演示了如何使用 `describe-user-import-job`。

Amazon CLI

描述用户导入任务

此示例描述用户输入任务。

有关导入用户的更多信息，请参阅“从 CSV 文件将用户导入用户池”。

命令:

```
aws cognito-idp describe-user-import-job --user-pool-id us-west-2_aaaaaaaaa --job-id import-TZqNQvDRnW
```

输出：

```
{
  "UserImportJob": {
      "JobName": "import-Test1",
      "JobId": "import-TZqNQvDRnW",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "PreSignedUrl": "PRE_SIGNED URL",
      "CreationDate": 1548271708.512,
      "Status": "Created",
      "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
      "ImportedUsers": 0,
      "SkippedUsers": 0,
      "FailedUsers": 0
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserImportJob。

`describe-user-pool-client`

以下代码示例演示了如何使用 `describe-user-pool-client`。

Amazon CLI

描述用户池客户端

此示例描述一个用户池客户端。

命令:

```
aws cognito-idp describe-user-pool-client --user-pool-id us-west-2_aaaaaaaaa --
client-id 38fjsnc484p94kpqsnet7mpld0
```

输出：

```
{
  "UserPoolClient": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "ClientName": "MyApp",
      "ClientId": "38fjsnc484p94kpqsnet7mpld0",
      "ClientSecret": "CLIENT_SECRET",
      "LastModifiedDate": 1548108676.163,
      "CreationDate": 1548108676.163,
      "RefreshTokenValidity": 30,
      "ReadAttributes": [
          "address",
          "birthdate",
          "custom:CustomAttr1",
          "custom:CustomAttr2",
          "email",
          "email_verified",
          "family_name",
          "gender",
          "given_name",
          "locale",
          "middle_name",
          "name",
          "nickname",
          "phone_number",
          "phone_number_verified",
          "picture",
          "preferred_username",
          "profile",
          "updated_at",
          "website",
          "zoneinfo"
```

```
        ],
        "WriteAttributes": [
            "address",
            "birthdate",
            "custom:CustomAttr1",
            "custom:CustomAttr2",
            "email",
            "family_name",
            "gender",
            "given_name",
            "locale",
            "middle_name",
            "name",
            "nickname",
            "phone_number",
            "picture",
            "preferred_username",
            "profile",
            "updated_at",
            "website",
            "zoneinfo"
        ],
        "ExplicitAuthFlows": [
            "ADMIN_NO_SRP_AUTH",
            "USER_PASSWORD_AUTH"
        ],
        "AllowedOAuthFlowsUserPoolClient": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserPoolClient。

`describe-user-pool-domain`

以下代码示例演示了如何使用 `describe-user-pool-domain`。

Amazon CLI

描述用户池客户端

此示例描述一个名为 my-domain 的用户池域。

命令:

```
aws cognito-idp describe-user-pool-domain --domain my-domain
```

输出：

```
{
  "DomainDescription": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "AWSAccountId": "111111111111",
      "Domain": "my-domain",
      "S3Bucket": "aws-cognito-prod-pdx-assets",
      "CloudFrontDistribution": "aaaaaaaaaaaaa.cloudfront.net",
      "Version": "20190128175402",
      "Status": "ACTIVE",
      "CustomDomainConfig": {}
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserPoolDomain。

describe-user-pool

以下代码示例演示了如何使用 describe-user-pool。

Amazon CLI

描述用户池

以下示例描述用户池 ID 为 us-west-2_EXAMPLE 的用户池。

```
aws cognito-idp describe-user-pool \
    --user-pool-id us-west-2_EXAMPLE
```

输出：

```
{
    "UserPool": {
        "Id": "us-west-2_EXAMPLE",
        "Name": "MyUserPool",
        "Policies": {
            "PasswordPolicy": {
                "MinimumLength": 8,
                "RequireUppercase": true,
```

```
                "RequireLowercase": true,
                "RequireNumbers": true,
                "RequireSymbols": true,
                "TemporaryPasswordValidityDays": 1
            }
        },
        "DeletionProtection": "ACTIVE",
        "LambdaConfig": {
            "PreSignUp": "arn:aws:lambda:us-
west-2:123456789012:function:MyPreSignUpFunction",
            "CustomMessage": "arn:aws:lambda:us-
west-2:123456789012:function:MyCustomMessageFunction",
            "PostConfirmation": "arn:aws:lambda:us-
west-2:123456789012:function:MyPostConfirmationFunction",
            "PreAuthentication": "arn:aws:lambda:us-
west-2:123456789012:function:MyPreAuthenticationFunction",
            "PostAuthentication": "arn:aws:lambda:us-
west-2:123456789012:function:MyPostAuthenticationFunction",
            "DefineAuthChallenge": "arn:aws:lambda:us-
west-2:123456789012:function:MyDefineAuthChallengeFunction",
            "CreateAuthChallenge": "arn:aws:lambda:us-
west-2:123456789012:function:MyCreateAuthChallengeFunction",
            "VerifyAuthChallengeResponse": "arn:aws:lambda:us-
west-2:123456789012:function:MyVerifyAuthChallengeFunction",
            "PreTokenGeneration": "arn:aws:lambda:us-
west-2:123456789012:function:MyPreTokenGenerationFunction",
            "UserMigration": "arn:aws:lambda:us-
west-2:123456789012:function:MyMigrateUserFunction",
            "PreTokenGenerationConfig": {
                "LambdaVersion": "V2_0",
                "LambdaArn": "arn:aws:lambda:us-
west-2:123456789012:function:MyPreTokenGenerationFunction"
            },
            "CustomSMSSender": {
                "LambdaVersion": "V1_0",
                "LambdaArn": "arn:aws:lambda:us-
west-2:123456789012:function:MyCustomSMSSenderFunction"
            },
            "CustomEmailSender": {
                "LambdaVersion": "V1_0",
                "LambdaArn": "arn:aws:lambda:us-
west-2:123456789012:function:MyCustomEmailSenderFunction"
            },
```

```
            "KMSKeyID": "arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
        },
        "LastModifiedDate": 1726784814.598,
        "CreationDate": 1602103465.273,
        "SchemaAttributes": [
            {
                "Name": "sub",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": false,
                "Required": true,
                "StringAttributeConstraints": {
                    "MinLength": "1",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "name",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "given_name",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "family_name",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
```

```
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "middle_name",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "nickname",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "preferred_username",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "profile",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
```

```
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "picture",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "website",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "email",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": true,
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "email_verified",
                "AttributeDataType": "Boolean",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
```

```
            "Required": false
        },
        {
            "Name": "gender",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            }
        },
        {
            "Name": "birthdate",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "10",
                "MaxLength": "10"
            }
        },
        {
            "Name": "zoneinfo",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            }
        },
        {
            "Name": "locale",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
```

```
            }
        },
        {
            "Name": "phone_number",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            }
        },
        {
            "Name": "phone_number_verified",
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false
        },
        {
            "Name": "address",
            "AttributeDataType": "String",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            }
        },
        {
            "Name": "updated_at",
            "AttributeDataType": "Number",
            "DeveloperOnlyAttribute": false,
            "Mutable": true,
            "Required": false,
            "NumberAttributeConstraints": {
                "MinValue": "0"
            }
        },
        {
            "Name": "identities",
            "AttributeDataType": "String",
```

```
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {}
            },
            {
                "Name": "custom:111",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "1",
                    "MaxLength": "256"
                }
            },
            {
                "Name": "dev:custom:222",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": true,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MinLength": "1",
                    "MaxLength": "421"
                }
            },
            {
                "Name": "custom:accesstoken",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
                    "MaxLength": "2048"
                }
            },
            {
                "Name": "custom:idtoken",
                "AttributeDataType": "String",
                "DeveloperOnlyAttribute": false,
                "Mutable": true,
                "Required": false,
                "StringAttributeConstraints": {
```

```
                    "MaxLength": "2048"
                }
            }
        ],
        "AutoVerifiedAttributes": [
            "email"
        ],
        "SmsVerificationMessage": "Your verification code is {####}. ",
        "EmailVerificationMessage": "Your verification code is {####}. ",
        "EmailVerificationSubject": "Your verification code",
        "VerificationMessageTemplate": {
            "SmsMessage": "Your verification code is {####}. ",
            "EmailMessage": "Your verification code is {####}. ",
            "EmailSubject": "Your verification code",
            "EmailMessageByLink": "Please click the link below to verify your email
 address. <b>{##Verify Your Email##}</b>\n this is from us-west-2_ywDJHlIfU",
            "EmailSubjectByLink": "Your verification link",
            "DefaultEmailOption": "CONFIRM_WITH_LINK"
        },
        "SmsAuthenticationMessage": "Your verification code is {####}. ",
        "UserAttributeUpdateSettings": {
            "AttributesRequireVerificationBeforeUpdate": []
        },
        "MfaConfiguration": "OPTIONAL",
        "DeviceConfiguration": {
            "ChallengeRequiredOnNewDevice": true,
            "DeviceOnlyRememberedOnUserPrompt": false
        },
        "EstimatedNumberOfUsers": 166,
        "EmailConfiguration": {
            "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com",
            "EmailSendingAccount": "DEVELOPER"
        },
        "SmsConfiguration": {
            "SnsCallerArn": "arn:aws:iam::123456789012:role/service-role/userpool-
SMS-Role",
            "ExternalId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "SnsRegion": "us-west-2"
        },
        "UserPoolTags": {},
        "Domain": "myCustomDomain",
        "CustomDomain": "auth.example.com",
        "AdminCreateUserConfig": {
```

```
            "AllowAdminCreateUserOnly": false,
            "UnusedAccountValidityDays": 1,
            "InviteMessageTemplate": {
                "SMSMessage": "Your username is {username} and temporary password is
 {####}. ",
                "EmailMessage": "Your username is {username} and temporary password
 is {####}. ",
                "EmailSubject": "Your temporary password"
            }
        },
        "UserPoolAddOns": {
            "AdvancedSecurityMode": "ENFORCED",
            "AdvancedSecurityAdditionalFlows": {}
        },
        "Arn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE",
        "AccountRecoverySetting": {
            "RecoveryMechanisms": [
                {
                    "Priority": 1,
                    "Name": "verified_email"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅 Amazon Cognito 开发人员指南中的 Amazon Cognito 用户池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserPool。

forget-device

以下代码示例演示了如何使用 forget-device。

Amazon CLI

忘记设备

此示例忘记一个设备。

命令:

```
aws cognito-idp forget-device --device-key us-west-2_abcd_1234-5678
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ForgetDevice。

forgot-password

以下代码示例演示了如何使用 forgot-password。

Amazon CLI

强制更改密码

以下的 forgot-password 示例向 jane@example.com 发送一条消息，要求他们更改密码。

```
aws cognito-idp forgot-password --client-id 38fjsnc484p94kpqsnet7mpld0 --
username jane@example.com
```

输出：

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ForgotPassword。

get-csv-header

以下代码示例演示了如何使用 get-csv-header。

Amazon CLI

创建 csv 标题

此示例创建一个 csv 标题。

有关导入用户的更多信息，请参阅“从 CSV 文件将用户导入用户池”。

命令:

```
aws cognito-idp get-csv-header --user-pool-id us-west-2_aaaaaaaaa
```

输出：

```
{
  "UserPoolId": "us-west-2_aaaaaaaaa",
  "CSVHeader": [
      "name",
      "given_name",
      "family_name",
      "middle_name",
      "nickname",
      "preferred_username",
      "profile",
      "picture",
      "website",
      "email",
      "email_verified",
      "gender",
      "birthdate",
      "zoneinfo",
      "locale",
      "phone_number",
      "phone_number_verified",
      "address",
      "updated_at",
      "cognito:mfa_enabled",
      "cognito:username"
  ]
}
```

...从 CSV 文件将用户导入用户池：https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-pools-using-import-tool.html

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCsvHeader。

get-device

以下代码示例演示了如何使用 get-device。

Amazon CLI

获取设备

以下 get-device 示例为当前已登录用户显示一台设备。

```
aws cognito-idp get-device \
    --access-token eyJra456defEXAMPLE \
    --device-key us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Device": {
        "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "DeviceAttributes": [
            {
                "Name": "device_status",
                "Value": "valid"
            },
            {
                "Name": "device_name",
                "Value": "MyDevice"
            },
            {
                "Name": "dev:device_arn",
                "Value": "arn:aws:cognito-idp:us-west-2:123456789012:owner/diego.us-
west-2_EXAMPLE/device/us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            },
            {
                "Name": "dev:device_owner",
                "Value": "diego.us-west-2_EXAMPLE"
            },
            {
                "Name": "last_ip_used",
                "Value": "192.0.2.1"
            },
            {
                "Name": "dev:device_remembered_status",
                "Value": "remembered"
            },
            {
                "Name": "dev:device_sdk",
```

```
                "Value": "aws-sdk"
            }
        ],
        "DeviceCreateDate": 1715100742.022,
        "DeviceLastModifiedDate": 1723233651.167,
        "DeviceLastAuthenticatedDate": 1715100742.0
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with user devices in your user pool。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDevice。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

获取有关组的信息

以下 get-group 示例列出名为 MyGroup 的用户组的属性。此组具有优先级及其关联的 IAM 角色。

```
aws cognito-idp get-group \
    --user-pool-id us-west-2_EXAMPLE \
    --group-name MyGroup
```

输出：

```
{
    "Group": {
        "GroupName": "MyGroup",
        "UserPoolId": "us-west-2_EXAMPLE",
        "RoleArn": "arn:aws:iam::123456789012:role/example-cognito-role",
        "Precedence": 7,
        "LastModifiedDate": 1697211218.305,
        "CreationDate": 1611685503.954
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding groups to a user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

get-identity-provider-by-identifier

以下代码示例演示了如何使用 get-identity-provider-by-identifier。

Amazon CLI

从 IdP 标识符获取身份提供者的配置

以下 get-identity-provider-by-identifier 示例返回带有标识符 mysso 的身份提供者的配置。

```
aws cognito-idp get-identity-provider-by-identifier \
    --user-pool-id us-west-2_EXAMPLE \
    --idp-identifier mysso
```

输出：

```
{
    "IdentityProvider": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ProviderName": "MYSAML",
        "ProviderType": "SAML",
        "ProviderDetails": {
            "ActiveEncryptionCertificate": "[Certificate contents]",
            "IDPSignout": "false",
            "MetadataURL": "https://auth.example.com/saml/metadata/",
            "SLORedirectBindingURI": "https://auth.example.com/saml/logout/",
            "SSORedirectBindingURI": "https://auth.example.com/saml/assertion/"
        },
        "AttributeMapping": {
            "email": "email"
        },
        "IdpIdentifiers": [
            "mysso",
            "mysamlsso"
        ],
        "LastModifiedDate": 1705616729.188,
        "CreationDate": 1643734622.919
    }
```

```
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Third-party IdP sign-in。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetIdentityProviderByIdentifier。

get-log-delivery-configuration

以下代码示例演示了如何使用 get-log-delivery-configuration。

Amazon CLI

显示日志传输配置

以下 get-log-delivery-configuration 示例显示所请求的用户池的日志导出设置。

```
aws cognito-idp get-log-delivery-configuration \
    --user-pool-id us-west-2_EXAMPLE
```

输出：

```
{
    "LogDeliveryConfiguration": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "LogConfigurations": [
            {
                "LogLevel": "INFO",
                "EventSource": "userAuthEvents",
                "FirehoseConfiguration": {
                    "StreamArn": "arn:aws:firehose:us-
west-2:123456789012:deliverystream/my-test-deliverystream"
                }
            },
            {
                "LogLevel": "ERROR",
                "EventSource": "userNotification",
                "CloudWatchLogsConfiguration": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-
group:my-message-delivery-logs"
                }
            }
```

```
        ]
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Exporting user pool logs。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetLogDeliveryConfiguration。

get-signing-certificate

以下代码示例演示了如何使用 get-signing-certificate。

Amazon CLI

显示 SAML 签名证书

以下 get-signing-certificate 示例显示了请求用户池的 SAML 2.0 签名证书。

```
aws cognito-idp get-signing-certificate \
    --user-pool-id us-west-2_EXAMPLE
```

输出：

```
{
    "Certificate": "[Certificate content]"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 SAML signing and encryption。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSigningCertificate。

get-ui-customization

以下代码示例演示了如何使用 get-ui-customization。

Amazon CLI

显示应用程序客户端的经典托管 UI 自定义设置

以下 get-ui-customization 示例显示了未从用户池继承设置的应用程序客户端的经典托管 UI 自定义设置。

```
aws cognito-idp get-ui-customization \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789
```

输出：

```
{
    "UICustomization": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ClientId": "1example23456789",
        "ImageUrl": "https://example.cloudfront.net/us-
west-2_EXAMPLE/1example23456789/20250115191928/assets/images/image.jpg",
        "CSS": "\n.logo-customizable {\n  max-width: 80%;\n  max-height: 30%;
\n}\n\n.banner-customizable {\n  padding: 25px 0px 25px 0px;\n  background-color:
 lightgray;\n}\n\n.label-customizable {\n  font-weight: 400;\n}\n\n.textDescription-
customizable {\n  padding-top: 100px;\n  padding-bottom: 10px;\n  display: block;
\n  font-size: 12px;\n}\n\n.idpDescription-customizable {\n  padding-top: 10px;\n
  padding-bottom: 10px;\n  display: block;\n  font-size: 16px;\n}\n\n.legalText-
customizable {\n  color: #747474;\n  font-size: 11px;\n}\n\n.submitButton-
customizable {\n  font-size: 14px;\n  font-weight: bold;\n  margin: 20px 0px
 10px 0px;\n  height: 50px;\n  width: 100%;\n  color: #fff;\n  background-color:
 #337ab7;\n}\n\n.submitButton-customizable:hover {\n  color: #fff;\n  background-
color: #286090;\n}\n\n.errorMessage-customizable {\n  padding: 5px;\n  font-size:
 12px;\n  width: 100%;\n  background: #F5F5F5;\n  border: 2px solid #D64958;\n
  color: #D64958;\n}\n\n.inputField-customizable {\n  width: 100%;\n  height:
 34px;\n  color: #555;\n  background-color: #fff;\n  border: 1px solid #ccc;\n}\n
\n.inputField-customizable:focus {\n  border-color: #66afe9;\n  outline: 0;\n}\n
\n.idpButton-customizable {\n  height: 40px;\n  width: 100%;\n  width: 100%;\n
 text-align: center;\n  margin-bottom: 15px;\n  color: #fff;\n  background-color:
 #5bc0de;\n  border-color: #46b8da;\n}\n\n.idpButton-customizable:hover {\n
 color: #fff;\n  background-color: #31b0d5;\n}\n\n.socialButton-customizable {\n
  border-radius: 2px;\n  height: 60px;\n  margin-bottom: 15px;\n  padding: 1px;
\n  text-align: left;\n  width: 100%;\n}\n\n.redirect-customizable {\n  text-
align: center;\n}\n\n.passwordCheck-notValid-customizable {\n  color: #DF3312;
\n}\n\n.passwordCheck-valid-customizable {\n  color: #19BF00;\n}\n\n.background-
customizable {\n  background-color: #fff;\n}\n",
        "CSSVersion": "20250115191928"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Hosted UI (classic) branding。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUiCustomization。

get-user-attribute-verification-code

以下代码示例演示了如何使用 get-user-attribute-verification-code。

Amazon CLI

向当前用户发送属性验证码

以下 get-user-attribute-verification-code 示例向当前登录用户的电子邮件地址发送属性验证码。

```
aws cognito-idp get-user-attribute-verification-code \
    --access-token eyJra456defEXAMPLE \
    --attribute-name email
```

输出：

```
{
    "CodeDeliveryDetails": {
        "Destination": "a***@e***",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

有关更多信息，请参阅《Amazon Cognito 开发人员指南》中的注册并确认用户账户。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetUserAttributeVerificationCode。

get-user-auth-factors

以下代码示例演示了如何使用 get-user-auth-factors。

Amazon CLI

列出当前用户可用的身份验证因素

以下 get-user-auth-factors 示例列出了当前登录用户可用的身份验证因素。

```
aws cognito-idp get-user-auth-factors \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "Username": "testuser",
    "ConfiguredUserAuthFactors": [
        "PASSWORD",
        "EMAIL_OTP",
        "SMS_OTP",
        "WEB_AUTHN"
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Authentication。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetUserAuthFactors。

get-user-pool-mfa-config

以下代码示例演示了如何使用 get-user-pool-mfa-config。

Amazon CLI

显示用户池的多重身份验证和 WebAuthn 设置

以下 get-user-pool-mfa-config 示例显示了所请求的用户池的 MFA 和 WebAuthn 配置。

```
aws cognito-idp get-user-pool-mfa-config \
    --user-pool-id us-west-2_EXAMPLE
```

输出：

```
{
    "SmsMfaConfiguration": {
        "SmsAuthenticationMessage": "Your OTP for MFA or sign-in: use {####}.",
        "SmsConfiguration": {
            "SnsCallerArn": "arn:aws:iam::123456789012:role/service-role/my-SMS-
Role",
            "ExternalId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "SnsRegion": "us-west-2"
        }
    },
    "SoftwareTokenMfaConfiguration": {
```

```
        "Enabled": true
    },
    "EmailMfaConfiguration": {
        "Message": "Your OTP for MFA or sign-in: use {####}",
        "Subject": "OTP test"
    },
    "MfaConfiguration": "OPTIONAL",
    "WebAuthnConfiguration": {
        "RelyingPartyId": "auth.example.com",
        "UserVerification": "preferred"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding MFA。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetUserPoolMfaConfig。

get-user

以下代码示例演示了如何使用 get-user。

Amazon CLI

获取当前用户的详细信息

以下 get-user 示例显示了当前登录用户的配置文件。

```
aws cognito-idp get-user \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "Username": "johndoe",
    "UserAttributes": [
        {
            "Name": "sub",
            "Value": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        },
        {
            "Name": "identities",
```

```
            "Value": "[{\"userId\":\"a1b2c3d4-5678-90ab-cdef-EXAMPLE22222\",
\"providerName\":\"SignInWithApple\",\"providerType\":\"SignInWithApple\",\"issuer
\":null,\"primary\":false,\"dateCreated\":1701125599632}]"
        },
        {
            "Name": "email_verified",
            "Value": "true"
        },
        {
            "Name": "custom:state",
            "Value": "Maine"
        },
        {
            "Name": "name",
            "Value": "John Doe"
        },
        {
            "Name": "phone_number_verified",
            "Value": "true"
        },
        {
            "Name": "phone_number",
            "Value": "+12065551212"
        },
        {
            "Name": "preferred_username",
            "Value": "jamesdoe"
        },
        {
            "Name": "locale",
            "Value": "EMEA"
        },
        {
            "Name": "email",
            "Value": "jamesdoe@example.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing users。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUser。

global-sign-out

以下代码示例演示了如何使用 global-sign-out。

Amazon CLI

注销当前用户

以下 global-sign-out 示例注销当前用户。

```
aws cognito-idp global-sign-out \
    --access-token eyJra456defEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing users。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GlobalSignOut。

initiate-auth

以下代码示例演示了如何使用 initiate-auth。

Amazon CLI

让用户登录

以下 initiate-auth 示例让用户使用基本的用户名/密码流程登录，无需进行其它质询。

```
aws cognito-idp initiate-auth \
    --auth-flow USER_PASSWORD_AUTH \
    --client-id 1example23456789 \
    --analytics-metadata AnalyticsEndpointId=d70b2ba36a8c4dc5a04a0451aEXAMPLE \
    --auth-parameters USERNAME=testuser,PASSWORD=[Password] --user-context-
data EncodedData=mycontextdata --client-metadata MyTestKey=MyTestValue
```

输出：

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
```

```
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Authentication。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 InitiateAuth。

list-devices

以下代码示例演示了如何使用 list-devices。

Amazon CLI

列出用户的设备

以下 list-devices 示例列出了当前用户已注册的设备。

```
aws cognito-idp list-devices \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "Devices": [
        {
            "DeviceAttributes": [
                {
                    "Name": "device_status",
                    "Value": "valid"
                },
                {
                    "Name": "device_name",
                    "Value": "Dart-device"
                },
                {
                    "Name": "last_ip_used",
                    "Value": "192.0.2.1"
```

```
                }
            ],
            "DeviceCreateDate": 1715100742.022,
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceLastAuthenticatedDate": 1715100742.0,
            "DeviceLastModifiedDate": 1723233651.167
        },
        {
            "DeviceAttributes": [
                {
                    "Name": "device_status",
                    "Value": "valid"
                },
                {
                    "Name": "last_ip_used",
                    "Value": "192.0.2.2"
                }
            ],
            "DeviceCreateDate": 1726856147.993,
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "DeviceLastAuthenticatedDate": 1726856147.0,
            "DeviceLastModifiedDate": 1726856147.993
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Working with devices。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDevices。

`list-groups`

以下代码示例演示了如何使用 `list-groups`。

Amazon CLI

列出用户池中的组

以下 `list-groups` 示例列出了所请求的用户池中的前两个组。

```
aws cognito-idp list-groups \
    --user-pool-id us-west-2_EXAMPLE \
    --max-items 2
```

输出：

```
{
    "Groups": [
        {
            "CreationDate": 1681760899.633,
            "Description": "My test group",
            "GroupName": "testgroup",
            "LastModifiedDate": 1681760899.633,
            "Precedence": 1,
            "UserPoolId": "us-west-2_EXAMPLE"
        },
        {
            "CreationDate": 1642632749.051,
            "Description": "Autogenerated group for users who sign in using
 Facebook",
            "GroupName": "us-west-2_EXAMPLE_Facebook",
            "LastModifiedDate": 1642632749.051,
            "UserPoolId": "us-west-2_EXAMPLE"
        }
    ],
    "NextToken": "[Pagination token]"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding groups to a user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

`list-identity-providers`

以下代码示例演示了如何使用 `list-identity-providers`。

Amazon CLI

列出身份提供者

以下 `list-identity-providers` 示例列出了所请求的用户池中的前两个身份提供者。

```
aws cognito-idp list-identity-providers \
    --user-pool-id us-west-2_EXAMPLE \
    --max-items 2
```

输出：

```
{
    "Providers": [
        {
            "CreationDate": 1619477386.504,
            "LastModifiedDate": 1703798328.142,
            "ProviderName": "Azure",
            "ProviderType": "SAML"
        },
        {
            "CreationDate": 1642698776.175,
            "LastModifiedDate": 1642699086.453,
            "ProviderName": "LoginWithAmazon",
            "ProviderType": "LoginWithAmazon"
        }
    ],
    "NextToken": "[Pagination token]"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Third-party IdP sign-in。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListIdentityProviders。

`list-resource-servers`

以下代码示例演示了如何使用 `list-resource-servers`。

Amazon CLI

列出资源服务器

以下 `list-resource-servers` 示例列出了所请求的用户池中的前两个资源服务器。

```
aws cognito-idp list-resource-servers \
    --user-pool-id us-west-2_EXAMPLE \
    --max-results 2
```

输出：

```
{
    "ResourceServers": [
        {
            "Identifier": "myapi.example.com",
            "Name": "Example API with custom access control scopes",
```

```
            "Scopes": [
                {
                    "ScopeDescription": "International customers",
                    "ScopeName": "international.read"
                },
                {
                    "ScopeDescription": "Domestic customers",
                    "ScopeName": "domestic.read"
                }
            ],
            "UserPoolId": "us-west-2_EXAMPLE"
        },
        {
            "Identifier": "myapi2.example.com",
            "Name": "Another example API for access control",
            "Scopes": [
                {
                    "ScopeDescription": "B2B customers",
                    "ScopeName": "b2b.read"
                }
            ],
            "UserPoolId": "us-west-2_EXAMPLE"
        }
    ],
    "NextToken": "[Pagination token]"
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Access control with resource servers。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceServers。

list-tags-for-resource

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出用户池标签

以下 `list-tags-for-resource` 示例列出了分配给带有所请求的 ARN 的用户池的标签。

```
aws cognito-idp list-tags-for-resource \
```

```
    --resource-arn arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE
```

输出：

```
{
    "Tags": {
        "administrator": "Jie",
        "tenant": "ExampleCorp"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Tagging Amazon Cognito resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-user-import-jobs

以下代码示例演示了如何使用 list-user-import-jobs。

Amazon CLI

列出用户导入任务和状态

以下 list-user-import-jobs 示例列出了所请求的用户池中的前三个用户导入任务及其详细信息。

```
aws cognito-idp list-user-import-jobs \
    --user-pool-id us-west-2_EXAMPLE \
    --max-results 3
```

输出：

```
{
    "PaginationToken": "us-west-2_EXAMPLE#import-example3#1667948397084",
    "UserImportJobs": [
        {
            "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/service-role/
Cognito-UserImport-Role",
            "CompletionDate": 1735329786.142,
            "CompletionMessage": "The user import job has expired.",
```

```
            "CreationDate": 1735241621.022,
            "FailedUsers": 0,
            "ImportedUsers": 0,
            "JobId": "import-example1",
            "JobName": "Test-import-job-1",
            "PreSignedUrl": "https://aws-cognito-idp-user-import-pdx.s3.us-
west-2.amazonaws.com/123456789012/us-west-2_EXAMPLE/import-mAgUtd8PMm?
X-Amz-Security-Token=[token]&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20241226T193341Z&X-Amz-SignedHeaders=host%3Bx-amz-server-side-encryption&X-Amz-
Expires=899&X-Amz-Credential=[credential]&X-Amz-Signature=[signature]",
            "SkippedUsers": 0,
            "Status": "Expired",
            "UserPoolId": "us-west-2_EXAMPLE"
        },
        {
            "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/service-role/
Cognito-UserImport-Role",
            "CompletionDate": 1681509058.408,
            "CompletionMessage": "Too many users have failed or been skipped during
 the import.",
            "CreationDate": 1681509001.477,
            "FailedUsers": 1,
            "ImportedUsers": 0,
            "JobId": "import-example2",
            "JobName": "Test-import-job-2",
            "PreSignedUrl": "https://aws-cognito-idp-user-import-pdx.s3.us-
west-2.amazonaws.com/123456789012/us-west-2_EXAMPLE/import-mAgUtd8PMm?
X-Amz-Security-Token=[token]&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20241226T193341Z&X-Amz-SignedHeaders=host%3Bx-amz-server-side-encryption&X-Amz-
Expires=899&X-Amz-Credential=[credential]&X-Amz-Signature=[signature]",
            "SkippedUsers": 0,
            "StartDate": 1681509057.965,
            "Status": "Failed",
            "UserPoolId": "us-west-2_EXAMPLE"
        },
        {
            "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/service-role/
Cognito-UserImport-Role",
            "CompletionDate": 1.667864578676E9,
            "CompletionMessage": "Import Job Completed Successfully.",
            "CreationDate": 1.667864480281E9,
            "FailedUsers": 0,
            "ImportedUsers": 6,
            "JobId": "import-example3",
```

```
            "JobName": "Test-import-job-3",
            "PreSignedUrl": "https://aws-cognito-idp-user-import-pdx.s3.us-
west-2.amazonaws.com/123456789012/us-west-2_EXAMPLE/import-mAgUtd8PMm?
X-Amz-Security-Token=[token]&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20241226T193341Z&X-Amz-SignedHeaders=host%3Bx-amz-server-side-encryption&X-Amz-
Expires=899&X-Amz-Credential=[credential]&X-Amz-Signature=[signature]",
            "SkippedUsers": 0,
            "StartDate": 1.667864578167E9,
            "Status": "Succeeded",
            "UserPoolId": "us-west-2_EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Importing users from a CSV file。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUserImportJobs。

list-user-pool-clients

以下代码示例演示了如何使用 list-user-pool-clients。

Amazon CLI

列出应用程序客户端

以下 list-user-pool-clients 示例列出了所请求的用户池中的前三个应用程序客户端。

```
aws cognito-idp list-user-pool-clients \
    --user-pool-id us-west-2_EXAMPLE \
    --max-results 3
```

输出：

```
{
    "NextToken": "[Pagination token]",
    "UserPoolClients": [
        {
            "ClientId": "1example23456789",
            "ClientName": "app-client-1",
            "UserPoolId": "us-west-2_EXAMPLE"
        },
```

```
        {
            "ClientId": "2example34567890",
            "ClientName": "app-client-2",
            "UserPoolId": "us-west-2_EXAMPLE"
        },
        {
            "ClientId": "3example45678901",
            "ClientName": "app-client-3",
            "UserPoolId": "us-west-2_EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 App clients。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListUserPoolClients。

`list-user-pools`

以下代码示例演示了如何使用 `list-user-pools`。

Amazon CLI

列出用户池

以下 `list-user-pools` 示例列出了当前 CLI 凭证的 Amazon 账户中的 3 个可用用户池。

```
aws cognito-idp list-user-pools \
    --max-results 3
```

输出：

```
{
    "NextToken": "[Pagination token]",
    "UserPools": [
        {
            "CreationDate": 1681502497.741,
            "Id": "us-west-2_EXAMPLE1",
            "LambdaConfig": {
                "CustomMessage": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreSignUp": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
```

```
                "PreTokenGeneration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreTokenGenerationConfig": {
                    "LambdaArn": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                    "LambdaVersion": "V1_0"
                }
            },
            "LastModifiedDate": 1681502497.741,
            "Name": "user pool 1"
        },
        {
            "CreationDate": 1686064178.717,
            "Id": "us-west-2_EXAMPLE2",
            "LambdaConfig": {
            },
            "LastModifiedDate": 1686064178.873,
            "Name": "user pool 2"
        },
        {
            "CreationDate": 1627681712.237,
            "Id": "us-west-2_EXAMPLE3",
            "LambdaConfig": {
                "UserMigration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction"
            },
            "LastModifiedDate": 1678486942.479,
            "Name": "user pool 3"
        }
    ]
}
```

有关更多信息，请参阅 Amazon Cognito 开发人员指南中的 Amazon Cognito 用户池。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListUserPools。

`list-users-in-group`

以下代码示例演示了如何使用 `list-users-in-group`。

Amazon CLI

列出组中的用户

此示例列出组 MyGroup 中的用户。

命令:

```
aws cognito-idp list-users-in-group --user-pool-id us-west-2_aaaaaaaaa --group-
name MyGroup
```

输出：

```
{
  "Users": [
      {
          "Username": "acf10624-80bb-401a-ac61-607bee2110ec",
          "Attributes": [
              {
                  "Name": "sub",
                  "Value": "acf10624-80bb-401a-ac61-607bee2110ec"
              },
              {
                  "Name": "custom:CustomAttr1",
                  "Value": "New Value!"
              },
              {
                  "Name": "email",
                  "Value": "jane@example.com"
              }
          ],
          "UserCreateDate": 1548102770.284,
          "UserLastModifiedDate": 1548103204.893,
          "Enabled": true,
          "UserStatus": "CONFIRMED"
      },
      {
          "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
          "Attributes": [
              {
                  "Name": "sub",
                  "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
              },
              {
                  "Name": "email_verified",
                  "Value": "true"
              },
```

```
            {
                "Name": "email",
                "Value": "diego@example.com"
            }
        ],
        "UserCreateDate": 1548089817.683,
        "UserLastModifiedDate": 1548089817.683,
        "Enabled": true,
        "UserStatus": "FORCE_CHANGE_PASSWORD"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsersInGroup。

`list-users`

以下代码示例演示了如何使用 `list-users`。

Amazon CLI

示例 1：使用服务器端筛选条件列出用户

以下 `list-users` 示例列出了所请求的用户池中其电子邮件地址以 `testuser` 开头的 3 个用户。

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --filter email^=\"testuser\" \
    --max-items 3
```

输出：

```
{
    "PaginationToken": "efgh5678EXAMPLE",
    "Users": [
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "eaad0219-2117-439f-8d46-4db20e59268f"
                },
                {
                    "Name": "email",
```

```
                    "Value": "testuser@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1682955829.578,
            "UserLastModifiedDate": 1689030181.63,
            "UserStatus": "CONFIRMED",
            "Username": "testuser"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "3b994cfd-0b07-4581-be46-3c82f9a70c90"
                },
                {
                    "Name": "email",
                    "Value": "testuser2@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427979.201,
            "UserLastModifiedDate": 1684427979.201,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser2"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "5929e0d1-4c34-42d1-9b79-a5ecacfe66f7"
                },
                {
                    "Name": "email",
                    "Value": "testuser3@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427823.641,
            "UserLastModifiedDate": 1684427823.641,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser3@example.com"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing and searching for users。

示例 2：使用客户端筛选条件列出用户

以下 `list-users` 示例列出了三个用户的属性，这些用户所具有的一个属性（在本例中为他们的电子邮件地址）包含电子邮件域“@example.com”。如果其它属性包含了此字符串，则也会显示这些属性。第二个用户没有与查询匹配的属性，因此会从显示的输出中排除，但不会从服务器响应中排除。

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --max-items 3
    --query Users\[\*\].Attributes\[\?Value\.contains\(\@\,\'@example.com\'\)\]
```

输出：

```
[
    [
        {
            "Name": "email",
            "Value": "admin@example.com"
        }
    ],
    [],
    [
        {
            "Name": "email",
            "Value": "operator@example.com"
        }
    ]
]
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managing and searching for users。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

list-web-authn-credentials

以下代码示例演示了如何使用 list-web-authn-credentials。

Amazon CLI

列出通行密钥凭证

以下 list-web-authn-credentials 示例列出了当前用户的通行密钥或 WebAuthn 凭证。他们有一台注册的设备。

```
aws cognito-idp list-web-authn-credentials \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "Credentials": [
        {
            "AuthenticatorAttachment": "cross-platform",
            "CreatedAt": 1736293876.115,
            "CredentialId": "8LApgk4-lNUFHbhm2w6Und7-
uxcc8coJGsPxiogvHoItc64xWQc3r4CEXAMPLE",
            "FriendlyCredentialName": "Roaming passkey",
            "RelyingPartyId": "auth.example.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Passkey sign-in。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListWebAuthnCredentials。

resend-confirmation-code

以下代码示例演示了如何使用 resend-confirmation-code。

Amazon CLI

重新发送确认码

以下 resend-confirmation-code 示例向用户 jane 发送确认码。

```
aws cognito-idp resend-confirmation-code \
    --client-id 12a3b456c7de890f11g123hijk \
    --username jane
```

输出：

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

有关更多信息，请参阅《Amazon Cognito 开发人员指南》中的注册并确认用户账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResendConfirmationCode。

respond-to-auth-challenge

以下代码示例演示了如何使用 respond-to-auth-challenge。

Amazon CLI

示例 1：响应 NEW_PASSWORD_REQUIRED 质询

以下 respond-to-auth-challenge 示例响应了 initiate-auth 返回的 NEW_PASSWORD_REQUIRED 质询。它为用户 jane@example.com 设置密码。

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --challenge-name NEW_PASSWORD_REQUIRED \
    --challenge-responses USERNAME=jane@example.com,NEW_PASSWORD=[Password] \
    --session AYABeEv5HklEXAMPLE
```

输出：

```
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "ACCESS_TOKEN",
        "ExpiresIn": 3600,
```

```
        "TokenType": "Bearer",
        "RefreshToken": "REFRESH_TOKEN",
        "IdToken": "ID_TOKEN",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-wt2ha1Zd"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Authentication。

示例 2：响应 SELECT_MFA_TYPE 质询

以下 `respond-to-auth-challenge` 示例选择 TOTP MFA 作为当前用户的 MFA 选项。系统会提示用户选择 MFA 类型，接下来将提示用户输入其 MFA 代码。

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789
    --session AYABeEv5HklEXAMPLE
    --challenge-name SELECT_MFA_TYPE
    --challenge-responses USERNAME=testuser,ANSWER=SOFTWARE_TOKEN_MFA
```

输出：

```
{
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeEv5HklEXAMPLE",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "transparent"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding MFA。

示例 3：响应 SOFTWARE_TOKEN_MFA 质询

以下 `respond-to-auth-challenge` 示例提供 TOTP MFA 代码并完成登录。

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --session AYABeEv5HklEXAMPLE \
```

```
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-responses USERNAME=testuser,SOFTWARE_TOKEN_MFA_CODE=123456
```

输出：

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding MFA。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 RespondToAuthChallenge。

`revoke-token`

以下代码示例演示了如何使用 `revoke-token`。

Amazon CLI

撤消刷新令牌

以下 `revoke-token` 撤消所请求的刷新令牌和关联的访问令牌。

```
aws cognito-idp revoke-token \
    --token eyJjd123abcEXAMPLE \
    --client-id 1example23456789
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Revoking tokens。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RevokeToken。

set-log-delivery-configuration

以下代码示例演示了如何使用 set-log-delivery-configuration。

Amazon CLI

设置从用户池导出日志

以下 set-log-delivery-configuration 示例将所请求的用户池配置为将用户通知错误记录到日志组，并将用户身份验证信息记录到 S3 存储桶。

```
aws cognito-idp set-log-delivery-configuration \
    --user-pool-id us-west-2_EXAMPLE \
    --log-
configurations LogLevel=ERROR,EventSource=userNotification,CloudWatchLogsConfiguration={LogG
west-2:123456789012:log-group:cognito-
exported} LogLevel=INFO,EventSource=userAuthEvents,S3Configuration={BucketArn=arn:aws:s3:::a
s3-demo-bucket1}
```

输出：

```
{
   "LogDeliveryConfiguration": {
        "LogConfigurations": [
            {
                "CloudWatchLogsConfiguration": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-
group:cognito-exported"
                },
                "EventSource": "userNotification",
                "LogLevel": "ERROR"
            },
            {
                "EventSource": "userAuthEvents",
                "LogLevel": "INFO",
                "S3Configuration": {
                    "BucketArn": "arn:aws:s3:::amzn-s3-demo-bucket1"
                }
            }
        ],
        "UserPoolId": "us-west-2_EXAMPLE"
   }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Exporting user pool logs。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SetLogDeliveryConfiguration。

`set-risk-configuration`

以下代码示例演示了如何使用 `set-risk-configuration`。

Amazon CLI

设置威胁防护风险配置

以下 `set-risk-configuration` 示例在所请求的应用程序客户端中配置威胁防护消息和操作、遭盗用的凭证以及 IP 地址异常。由于 NotifyConfiguration 对象的复杂性，JSON 输入是此命令的最佳实践。

```
aws cognito-idp set-risk-configuration \
    --cli-input-json file://set-risk-configuration.json
```

的内容`set-risk-configuration.json`：

```
{
    "AccountTakeoverRiskConfiguration": {
        "Actions": {
            "HighAction": {
                "EventAction": "MFA_REQUIRED",
                "Notify": true
            },
            "LowAction": {
                "EventAction": "NO_ACTION",
                "Notify": true
            },
            "MediumAction": {
                "EventAction": "MFA_IF_CONFIGURED",
                "Notify": true
            }
        },
        "NotifyConfiguration": {
            "BlockEmail": {
                "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 blocked an unrecognized sign-in to your account with this information:\n<ul>
```

```
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                "Subject": "Blocked sign-in attempt",
                "TextBody": "We blocked an unrecognized sign-in to your account with
 this information:\nTime: {login-time}\nDevice: {device-name}\nLocation: {city},
 {country}\nIf this sign-in was not by you, you should change your password and
 notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you, you
 can follow {one-click-link-valid} to let us know"
            },
            "From": "admin@example.com",
            "MfaEmail": {
                "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML email
 context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We required
 you to use multi-factor authentication for the following sign-in attempt:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                "Subject": "New sign-in attempt",
                "TextBody": "We required you to use multi-factor authentication
 for the following sign-in attempt:\nTime: {login-time}\nDevice: {device-
name}\nLocation: {city}, {country}\nIf this sign-in was not by you, you should
 change your password and notify us by clicking on {one-click-link-invalid}\nIf this
 sign-in was by you, you can follow {one-click-link-valid} to let us know"
            },
            "NoActionEmail": {
                "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 observed an unrecognized sign-in to your account with this information:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                "Subject": "New sign-in attempt",
                "TextBody": "We observed an unrecognized sign-in to your account
 with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation:
 {city}, {country}\nIf this sign-in was not by you, you should change your password
 and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you,
 you can follow {one-click-link-valid} to let us know"
```

```
            },
            "ReplyTo": "admin@example.com",
            "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
        }
    },
    "ClientId": "1example23456789",
    "CompromisedCredentialsRiskConfiguration": {
        "Actions": {
            "EventAction": "BLOCK"
        },
        "EventFilter": [
            "PASSWORD_CHANGE",
            "SIGN_UP",
            "SIGN_IN"
        ]
    },
    "RiskExceptionConfiguration": {
        "BlockedIPRangeList": [
            "192.0.2.1/32",
            "192.0.2.2/32"
        ],
        "SkippedIPRangeList": [
            "203.0.113.1/32",
            "203.0.113.2/32"
        ]
    },
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

输出：

```
{
    "RiskConfiguration": {
        "AccountTakeoverRiskConfiguration": {
            "Actions": {
                "HighAction": {
                    "EventAction": "MFA_REQUIRED",
                    "Notify": true
                },
                "LowAction": {
                    "EventAction": "NO_ACTION",
                    "Notify": true
```

```
                },
                "MediumAction": {
                    "EventAction": "MFA_IF_CONFIGURED",
                    "Notify": true
                }
            },
            "NotifyConfiguration": {
                "BlockEmail": {
                    "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 blocked an unrecognized sign-in to your account with this information:\n<ul>
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                    "Subject": "Blocked sign-in attempt",
                    "TextBody": "We blocked an unrecognized sign-in to your account
 with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation:
 {city}, {country}\nIf this sign-in was not by you, you should change your password
 and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you,
 you can follow {one-click-link-valid} to let us know"
                },
                "From": "admin@example.com",
                "MfaEmail": {
                    "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 required you to use multi-factor authentication for the following sign-in attempt:
\n<ul>\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location:
 {city}, {country}</li>\n</ul>\nIf this sign-in was not by you, you should change
 your password and notify us by clicking on <a href={one-click-link-invalid}>this
 link</a>\nIf this sign-in was by you, you can follow <a href={one-click-link-
valid}>this link</a> to let us know</pre>\n</body>\n</html>",
                    "Subject": "New sign-in attempt",
                    "TextBody": "We required you to use multi-factor authentication
 for the following sign-in attempt:\nTime: {login-time}\nDevice: {device-
name}\nLocation: {city}, {country}\nIf this sign-in was not by you, you should
 change your password and notify us by clicking on {one-click-link-invalid}\nIf this
 sign-in was by you, you can follow {one-click-link-valid} to let us know"
                },
                "NoActionEmail": {
                    "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML
 email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We
 observed an unrecognized sign-in to your account with this information:\n<ul>
```

```
\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city},
 {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your
 password and notify us by clicking on <a href={one-click-link-invalid}>this link</
a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this
 link</a> to let us know</pre>\n</body>\n</html>",
                    "Subject": "New sign-in attempt",
                    "TextBody": "We observed an unrecognized sign-in to your account
 with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation:
 {city}, {country}\nIf this sign-in was not by you, you should change your password
 and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you,
 you can follow {one-click-link-valid} to let us know"
                },
                "ReplyTo": "admin@example.com",
                "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
            }
        },
        "ClientId": "1example23456789",
        "CompromisedCredentialsRiskConfiguration": {
            "Actions": {
                "EventAction": "BLOCK"
            },
            "EventFilter": [
                "PASSWORD_CHANGE",
                "SIGN_UP",
                "SIGN_IN"
            ]
        },
        "RiskExceptionConfiguration": {
            "BlockedIPRangeList": [
                "192.0.2.1/32",
                "192.0.2.2/32"
            ],
            "SkippedIPRangeList": [
                "203.0.113.1/32",
                "203.0.113.2/32"
            ]
        },
        "UserPoolId": "us-west-2_EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Threat protection。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetRiskConfiguration。

set-ui-customization

以下代码示例演示了如何使用 set-ui-customization。

Amazon CLI

示例 1：为应用程序客户端自定义经典托管 UI

以下 set-ui-customization 示例使用一些自定义 CSS 和将 Amazon Cognito 徽标作为应用程序徽标，来配置所请求的应用程序客户端。

```
aws cognito-idp set-ui-customization \
    --user-pool-id us-west-2_ywDJHlIfU \
    --client-id 14pq32c5q2uq2q7keorloqvb23 \
    --css ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;\n}\n.banner-
customizable {\n\tpadding: 25px 0px 25px 0px;\n\tbackground-color: lightgray;
\n}\n.label-customizable {\n\tfont-weight: 400;\n}\n.textDescription-customizable
 {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-
size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-
bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-customizable
 {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable {\n\tfont-
size: 11px;\n\tfont-weight: normal;\n\tmargin: 20px -15px 10px -13px;\n\theight:
 40px;\n\twidth: 108%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;\n\ttext-align:
 center;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color:
 #286090;\n}\n.errorMessage-customizable {\n\tpadding: 5px;\n\tfont-size: 14px;
\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder: 2px solid #D64958;\n\tcolor:
 #D64958;\n}\n.inputField-customizable {\n\twidth: 100%;\n\theight: 34px;\n\tcolor:
 #555;\n\tbackground-color: #fff;\n\tborder: 1px solid #ccc;\n\tborder-radius:
 0px;\n}\n.inputField-customizable:focus {\n\tborder-color: #66afe9;\n\toutline:
 0;\n}\n.idpButton-customizable {\n\theight: 40px;\n\twidth: 100%;\n\twidth: 100%;
\n\ttext-align: center;\n\tmargin-bottom: 15px;\n\tcolor: #fff;\n\tbackground-
color: #5bc0de;\n\tborder-color: #46b8da;\n}\n.idpButton-customizable:hover {\n
\tcolor: #fff;\n\tbackground-color: #31b0d5;\n}\n.socialButton-customizable {\n
\tborder-radius: 2px;\n\theight: 40px;\n\tmargin-bottom: 15px;\n\tpadding: 1px;
\n\ttext-align: left;\n\twidth: 100%;\n}\n.redirect-customizable {\n\ttext-
align: center;\n}\n.passwordCheck-notValid-customizable {\n\tcolor: #DF3312;
\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;\n}\n.background-
customizable {\n\tbackground-color: #fff;\n}\n" \
    --image-
file iVBORw0KGgoAAAANSUhEUgAAAFAAAABQCAMAAAC5zwKfAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAA
Cmsfvm6f3y9P////fM0uqAj
```

```
+yNmu6ZpvnZ3eNabuFNYuZneehzhPKzvPTAxwAAAOiMMlkAAAASdFJOU//////////////////////
AOK/vxIAAAAJcEhZcwAADsMAAA7DAcdvqGQAAAKDSURBVFhH7ZfpkoMgEISDHKuEw/
d/2u2BQWMiBrG29o+fVsKatdPMAeZxc3Nz8w+ISekzmB++sYIw/I/
tjHzrPpO2Tx62EbR2PNxFac+jVuKxRaV50IzXkUe76NOCoUuwlvnQKei02gNF0ykotOLRBq/
nboeWRxAISx2EbsHFoRhK6Igk2JJlwScfQjgt06dOaWWiTbEDAe/
iq8N9kqCw2uCbHkHlYkaXEF8EYeL9RDqT4FhC6XMIIEifdcUwCc4leNyhabadWU6OlKYJE1Oac3NSPhB5rlaXlSgmr/1
F0L6Q5pZiSG0SfZTSTCOUhxOCH1AdIoCpTTIjtd
+VpEjUDDytQH/0Fpc661Aisas/4qmyUItD557pSCOSQQzlx27J
+meyDGc5zZgfhWuXE1lGgmVOMwmWdeGdzhjqZV14x5vSj7vsC5JDz/Cl0Vhp56n2NQt1wQIpury1EPbwyaYm
+IhmAQKoajkH51wg4cMZ1wQ3QG9efKWWOaDhYWnU6jXjCMdRmm21PArI
+Pb5DYoH93hq0ZCPlxeGJho/DI15C6sQc/L2sTC47UFBKZGHT6k+zlXg7WebA0Nr0HTcLMfk/
Y4Rc65D3iG6WDd7YLSlVqk87bVhUwhnClrx11RsVQwlAA818Mn
+QEs71BhSFU6orsUfKhHp72XMGYXi4q9c64RXRvzkWurRfG2vI2be/VaNcNgpX0Evb/
vio7nPMmj5qujkpQgSaPd1UcVqciHFDNZpOcGlcOPyi+AamCbIL9fitxAGeFN2Dl
+3vZubm5u/4fH4Bd14HhIPdwZPAAAAAElFTkSuQmCC
```

输出：

```
{
    "UICustomization": {
        "UserPoolId": "us-west-2_ywDJHlIfU",
        "ClientId": "14pq32c5q2uq2q7keorloqvb23",
        "ImageUrl": "https://
cf.thewrong.club/14pq32c5q2uq2q7keorloqvb23/20250117005911/assets/images/image.jpg",
        "CSS": ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;
\n}\n.banner-customizable {\n\tpadding: 25px 0px 25px 0px;\n\tbackground-color:
 lightgray;\n}\n.label-customizable {\n\tfont-weight: 400;\n}\n.textDescription-
customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;
\n\tfont-size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n
\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-
customizable {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable
 {\n\tfont-size: 11px;\n\tfont-weight: normal;\n\tmargin: 20px -15px 10px -13px;
\n\theight: 40px;\n\twidth: 108%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;
\n\ttext-align: center;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;
\n\tbackground-color: #286090;\n}\n.errorMessage-customizable {\n\tpadding:
 5px;\n\tfont-size: 14px;\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder:
 2px solid #D64958;\n\tcolor: #D64958;\n}\n.inputField-customizable {\n\twidth:
 100%;\n\theight: 34px;\n\tcolor: #555;\n\tbackground-color: #fff;\n\tborder:
 1px solid #ccc;\n\tborder-radius: 0px;\n}\n.inputField-customizable:focus {\n
\tborder-color: #66afe9;\n\toutline: 0;\n}\n.idpButton-customizable {\n\theight:
 40px;\n\twidth: 100%;\n\twidth: 100%;\n\ttext-align: center;\n\tmargin-bottom:
 15px;\n\tcolor: #fff;\n\tbackground-color: #5bc0de;\n\tborder-color: #46b8da;
\n}\n.idpButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #31b0d5;
```

```
\n}\n.socialButton-customizable {\n\tborder-radius: 2px;\n\theight: 40px;\n\tmargin-
bottom: 15px;\n\tpadding: 1px;\n\ttext-align: left;\n\twidth: 100%;\n}\n.redirect-
customizable {\n\ttext-align: center;\n}\n.passwordCheck-notValid-customizable
 {\n\tcolor: #DF3312;\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;
\n}\n.background-customizable {\n\tbackground-color: #fff;\n}\n",
        "CSSVersion": "20250117005911"
    }
}
```

示例 2：为所有应用程序客户端设置默认 UI 自定义

以下 `set-ui-customization` 示例为所有没有客户端特定配置的应用程序客户端配置所请求的用户池。该命令应用一些自定义 CSS，并使用 Amazon Cognito 徽标作为应用程序徽标。

```
aws cognito-idp set-ui-customization \
--user-pool-id us-west-2_ywDJHlIfU \
--client-id ALL \
--css ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;\n}\n.banner-
customizable {\n\tpadding: 25px 0px 25px 0px;\n\tbackground-color: lightgray;
\n}\n.label-customizable {\n\tfont-weight: 400;\n}\n.textDescription-customizable
 {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-
size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-
bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-customizable
 {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable {\n\tfont-
size: 11px;\n\tfont-weight: normal;\n\tmargin: 20px -15px 10px -13px;\n\theight:
 40px;\n\twidth: 108%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;\n\ttext-align:
 center;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color:
 #286090;\n}\n.errorMessage-customizable {\n\tpadding: 5px;\n\tfont-size: 14px;
\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder: 2px solid #D64958;\n\tcolor:
 #D64958;\n}\n.inputField-customizable {\n\twidth: 100%;\n\theight: 34px;\n\tcolor:
 #555;\n\tbackground-color: #fff;\n\tborder: 1px solid #ccc;\n\tborder-radius:
 0px;\n}\n.inputField-customizable:focus {\n\tborder-color: #66afe9;\n\toutline:
 0;\n}\n.idpButton-customizable {\n\theight: 40px;\n\twidth: 100%;\n\twidth: 100%;
\n\ttext-align: center;\n\tmargin-bottom: 15px;\n\tcolor: #fff;\n\tbackground-
color: #5bc0de;\n\tborder-color: #46b8da;\n}\n.idpButton-customizable:hover {\n
\tcolor: #fff;\n\tbackground-color: #31b0d5;\n}\n.socialButton-customizable {\n
\tborder-radius: 2px;\n\theight: 40px;\n\tmargin-bottom: 15px;\n\tpadding: 1px;
\n\ttext-align: left;\n\twidth: 100%;\n}\n.redirect-customizable {\n\ttext-
align: center;\n}\n.passwordCheck-notValid-customizable {\n\tcolor: #DF3312;
\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;\n}\n.background-
customizable {\n\tbackground-color: #fff;\n}\n" \
--image-
file iVBORw0KGgoAAAANSUhEUgAAAFAAAABQCAMAAAC5zwKfAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAA
```

```
Cmsfvm6f3y9P////fM0uqAj
+yNmu6ZpvnZ3eNabuFNYuZneehzhPKzvPTAxwAAAOiMMlkAAAASdFJOU//////////////////////
AOK/vxIAAAAJcEhZcwAADsMAAA7DAcdvqGQAAAKDSURBVFhH7ZfpkoMgEISDHKuEw/
d/2u2BQWMiBrG29o+fVsKatdPMAeZxc3Nz8w+ISekzmB++sYIw/I/
tjHzrPpO2Tx62EbR2PNxFac+jVuKxRaV50IzXkUe76NOCoUuwlvnQKei02gNF0ykotOLRBq/
nboeWRxAISx2EbsHFoRhK6Igk2JJlwScfQjgt06dOaWWiTbEDAe/
iq8N9kqCw2uCbHkHlYkaXEF8EYeL9RDqT4FhC6XMIIEifdcUwCc4leNyhabadWU6OlKYJE1Oac3NSPhB5rlaXlSgmr/1
F0L6Q5pZiSG0SfZTSTCOUhxOCH1AdIoCpTTIjtd
+VpEjUDDytQH/0Fpc661Aisas/4qmyUItD557pSCOSQQzlx27J
+meyDGc5zZgfhWuXE1lGgmVOMwmWdeGdzhjqZV14x5vSj7vsC5JDz/Cl0Vhp56n2NQt1wQIpury1EPbwyaYm
+IhmAQKoajkH51wg4cMZ1wQ3QG9efKWWOaDhYWnU6jXjCMdRmm21PArI
+Pb5DYoH93hq0ZCPlxeGJho/DI15C6sQc/L2sTC47UFBKZGHT6k+zlXg7WebA0Nr0HTcLMfk/
Y4Rc65D3iG6WDd7YLSlVqk87bVhUwhnClrx11RsVQwlAA818Mn
+QEs71BhSFU6orsUfKhHp72XMGYXi4q9c64RXRvzkWurRfG2vI2be/VaNcNgpX0Evb/
vio7nPMmj5qujkpQgSaPd1UcVqciHFDNZpOcGlcOPyi+AamCbIL9fitxAGeFN2Dl
+3vZubm5u/4fH4Bd14HhIPdwZPAAAAAElFTkSuQmCC
```

输出：

```
{
    "UICustomization": {
        "UserPoolId": "us-west-2_ywDJHlIfU",
        "ClientId": "14pq32c5q2uq2q7keorloqvb23",
        "ImageUrl": "https://
cf.thewrong.club/14pq32c5q2uq2q7keorloqvb23/20250117005911/assets/images/image.jpg",
        "CSS": ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;
\n}\n.banner-customizable {\n\tpadding: 25px 0px 25px 0px;\n\tbackground-color:
 lightgray;\n}\n.label-customizable {\n\tfont-weight: 400;\n}\n.textDescription-
customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;
\n\tfont-size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n
\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-
customizable {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable
 {\n\tfont-size: 11px;\n\tfont-weight: normal;\n\tmargin: 20px -15px 10px -13px;
\n\theight: 40px;\n\twidth: 108%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;
\n\ttext-align: center;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;
\n\tbackground-color: #286090;\n}\n.errorMessage-customizable {\n\tpadding:
 5px;\n\tfont-size: 14px;\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder:
 2px solid #D64958;\n\tcolor: #D64958;\n}\n.inputField-customizable {\n\twidth:
 100%;\n\theight: 34px;\n\tcolor: #555;\n\tbackground-color: #fff;\n\tborder:
 1px solid #ccc;\n\tborder-radius: 0px;\n}\n.inputField-customizable:focus {\n
\tborder-color: #66afe9;\n\toutline: 0;\n}\n.idpButton-customizable {\n\theight:
 40px;\n\twidth: 100%;\n\twidth: 100%;\n\ttext-align: center;\n\tmargin-bottom:
 15px;\n\tcolor: #fff;\n\tbackground-color: #5bc0de;\n\tborder-color: #46b8da;
```

```
\n}\n.idpButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #31b0d5;
\n}\n.socialButton-customizable {\n\tborder-radius: 2px;\n\theight: 40px;\n\tmargin-
bottom: 15px;\n\tpadding: 1px;\n\ttext-align: left;\n\twidth: 100%;\n}\n.redirect-
customizable {\n\ttext-align: center;\n}\n.passwordCheck-notValid-customizable
 {\n\tcolor: #DF3312;\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;
\n}\n.background-customizable {\n\tbackground-color: #fff;\n}\n",
        "CSSVersion": "20250117005911"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Hosted UI (classic) branding。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetUiCustomization。

set-user-mfa-preference

以下代码示例演示了如何使用 set-user-mfa-preference。

Amazon CLI

设置用户的 MFA 首选项

以下 set-user-mfa-preference 示例将当前用户配置为使用 TOTP MFA 并禁用所有其它 MFA 因素。

```
aws cognito-idp set-user-mfa-preference \
    --access-token eyJra456defEXAMPLE \
    --software-token-mfa-settings Enabled=true,PreferredMfa=true \
    --sms-mfa-settings Enabled=false,PreferredMfa=false \
    --email-mfa-settings Enabled=false,PreferredMfa=false
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding MFA。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetUserMfaPreference。

set-user-pool-mfa-config

以下代码示例演示了如何使用 set-user-pool-mfa-config。

Amazon CLI

配置用户池 MFA 和 WebAuthn

以下 `set-user-pool-mfa-config` 示例使用可选 MFA 及所有可用的 MFA 方法配置所请求的用户池，并设置 WebAuthn 配置。

```
aws cognito-idp set-user-pool-mfa-config \
    --user-pool-id us-west-2_EXAMPLE \
    --sms-mfa-configuration "SmsAuthenticationMessage=\"Your OTP for MFA or sign-
in: use {####}.\",SmsConfiguration={SnsCallerArn=arn:aws:iam::123456789012:role/
service-role/test-SMS-Role,ExternalId=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111,SnsRegion=us-west-2}" \
    --software-token-mfa-configuration Enabled=true \
    --email-mfa-configuration "Message=\"Your OTP for MFA or sign-in: use
 {####}\",Subject=\"OTP test\"" \
    --mfa-configuration OPTIONAL \
    --web-authn-
configuration RelyingPartyId=auth.example.com,UserVerification=preferred
```

输出：

```
{
    "EmailMfaConfiguration": {
        "Message": "Your OTP for MFA or sign-in: use {####}",
        "Subject": "OTP test"
    },
    "MfaConfiguration": "OPTIONAL",
    "SmsMfaConfiguration": {
        "SmsAuthenticationMessage": "Your OTP for MFA or sign-in: use {####}.",
        "SmsConfiguration": {
            "ExternalId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "SnsCallerArn": "arn:aws:iam::123456789012:role/service-role/test-SMS-
Role",
            "SnsRegion": "us-west-2"
        }
    },
    "SoftwareTokenMfaConfiguration": {
        "Enabled": true
    },
    "WebAuthnConfiguration": {
        "RelyingPartyId": "auth.example.com",
        "UserVerification": "preferred"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Adding MFA 和 Passkey sign-in。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SetUserPoolMfaConfig。

`set-user-settings`

以下代码示例演示了如何使用 `set-user-settings`。

Amazon CLI

设置用户设置

此示例将 MFA 发送首选项设置为 EMAIL。

命令:

```
aws cognito-idp set-user-settings --access-token ACCESS_TOKEN --mfa-
options DeliveryMedium=EMAIL
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetUserSettings。

`sign-up`

以下代码示例演示了如何使用 `sign-up`。

Amazon CLI

注册用户

此示例注册 jane@example.com。

命令:

```
aws cognito-idp sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username jane@example.com --password PASSWORD --user-attributes
 Name="email",Value="jane@example.com" Name="name",Value="Jane"
```

输出：

```
{
  "UserConfirmed": false,
  "UserSub": "e04d60a6-45dc-441c-a40b-e25a787d4862"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SignUp。

start-user-import-job

以下代码示例演示了如何使用 start-user-import-job。

Amazon CLI

启动导入任务

以下 start-user-import-job 示例在所请求的用户池中启动所请求的导入任务。

```
aws cognito-idp start-user-import-job \
    --user-pool-id us-west-2_EXAMPLE \
    --job-id import-mAgUtd8PMm
```

输出：

```
{
    "UserImportJob": {
        "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/example-cloudwatch-
logs-role",
        "CreationDate": 1736442975.904,
        "FailedUsers": 0,
        "ImportedUsers": 0,
        "JobId": "import-mAgUtd8PMm",
        "JobName": "Customer import",
        "PreSignedUrl": "https://aws-cognito-idp-user-import-pdx.s3.us-
west-2.amazonaws.com/123456789012/us-west-2_EXAMPLE/import-mAgUtd8PMm?
X-Amz-Security-Token=[token]&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20241226T193341Z&X-Amz-SignedHeaders=host%3Bx-amz-server-side-encryption&X-Amz-
Expires=899&X-Amz-Credential=[credential]&X-Amz-Signature=[signature]",
        "SkippedUsers": 0,
        "StartDate": 1736443020.081,
        "Status": "Pending",
```

```
        "UserPoolId": "us-west-2_EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Importing users into a user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartUserImportJob。

`start-web-authn-registration`

以下代码示例演示了如何使用 `start-web-authn-registration`。

Amazon CLI

获取已登录用户的通行密钥注册信息

以下 `start-web-authn-registration` 示例为当前用户生成 WebAuthn 注册选项。

```
aws cognito-idp start-web-authn-registration \
    --access-token eyJra456defEXAMPLE
```

输出：

```
{
    "CredentialCreationOptions": {
        "authenticatorSelection": {
            "requireResidentKey": true,
            "residentKey": "required",
            "userVerification": "preferred"
        },
        "challenge": "wxvbDicyqQqvF2EXAMPLE",
        "excludeCredentials": [
            {
                "id": "8LApgk4-lNUFHbhm2w6Und7-
uxcc8coJGsPxiogvHoItc64xWQc3r4CEXAMPLE",
                "type": "public-key"
            }
        ],
        "pubKeyCredParams": [
            {
                "alg": -7,
                "type": "public-key"
```

```
            },
            {
                "alg": -257,
                "type": "public-key"
            }
        ],
        "rp": {
            "id": "auth.example.com",
            "name": "auth.example.com"
        },
        "timeout": 60000,
        "user": {
            "displayName": "testuser",
            "id": "ZWFhZDAyMTktMjExNy00MzlmLThkNDYtNGRiMjBlNEXAMPLE",
            "name": "testuser"
        }
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Passkey sign-in。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartWebAuthnRegistration。

stop-user-import-job

以下代码示例演示了如何使用 stop-user-import-job。

Amazon CLI

停止导入任务

以下 stop-user-import-job 示例在所请求的用户池中停止所请求的正在运行的用户导入任务。

```
aws cognito-idp stop-user-import-job \
    --user-pool-id us-west-2_EXAMPLE \
    --job-id import-mAgUtd8PMm
```

输出：

```
{
```

```
    "UserImportJob": {
        "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/example-cloudwatch-
logs-role",
        "CompletionDate": 1736443496.379,
        "CompletionMessage": "The Import Job was stopped by the developer.",
        "CreationDate": 1736443471.781,
        "FailedUsers": 0,
        "ImportedUsers": 0,
        "JobId": "import-mAgUtd8PMm",
        "JobName": "Customer import",
        "PreSignedUrl": "https://aws-cognito-idp-user-import-pdx.s3.us-
west-2.amazonaws.com/123456789012/us-west-2_EXAMPLE/import-mAgUtd8PMm?
X-Amz-Security-Token=[token]&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20241226T193341Z&X-Amz-SignedHeaders=host%3Bx-amz-server-side-encryption&X-Amz-
Expires=899&X-Amz-Credential=[credential]&X-Amz-Signature=[signature]",
        "SkippedUsers": 0,
        "StartDate": 1736443494.154,
        "Status": "Stopped",
        "UserPoolId": "us-west-2_EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Importing users into a user pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopUserImportJob。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

为用户池添加标签

以下 `tag-resource` 示例将 `administrator` 和 `department` 标签应用于所请求的用户池。

```
aws cognito-idp tag-resource \
    --resource-arn arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE \
    --tags administrator=Jie,tenant=ExampleCorp
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Tagging Amazon Cognito resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从用户池中移除标签

以下 untag-resource 示例从所请求的用户池移除 administrator 和 department 标签。

```
aws cognito-idp untag-resource \
    --resource-arn arn:aws:cognito-idp:us-west-2:767671399759:userpool/us-west-2_l5cxwdm2K \
    --tag-keys administrator tenant
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Tagging Amazon Cognito resources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-auth-event-feedback

以下代码示例演示了如何使用 update-auth-event-feedback。

Amazon CLI

更新身份验证事件反馈

此示例更新授权事件反馈。它将事件标记为“Valid”。

命令:

```
aws cognito-idp update-auth-event-feedback --user-pool-id us-west-2_aaaaaaaaa --username diego@example.com --event-id EVENT_ID --feedback-token FEEDBACK_TOKEN --feedback-value "Valid"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAuthEventFeedback。

update-device-status

以下代码示例演示了如何使用 update-device-status。

Amazon CLI

更新设备状态

此示例将设备状态更新为“not_remembered”。

命令:

```
aws cognito-idp update-device-status --access-token ACCESS_TOKEN --device-
key DEVICE_KEY --device-remembered-status "not_remembered"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeviceStatus。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

更新组

此示例更新 MyGroup 的说明和优先级。

命令:

```
aws cognito-idp update-group --user-pool-id us-west-2_aaaaaaaaa --group-name MyGroup
 --description "New description" --precedence 2
```

输出：

```
{
  "Group": {
      "GroupName": "MyGroup",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "Description": "New description",
      "RoleArn": "arn:aws:iam::111111111111:role/MyRole",
```

```
      "Precedence": 2,
      "LastModifiedDate": 1548800862.812,
      "CreationDate": 1548097827.125
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

update-identity-provider

以下代码示例演示了如何使用 update-identity-provider。

Amazon CLI

更新用户池身份提供者

以下 update-identity-provider 示例更新所请求的用户池中的 OIDC 提供者“MyOIDCIdP”。

```
aws cognito-idp update-identity-provider \
    --cli-input-json file://update-identity-provider.json
```

的内容update-identity-provider.json：

```
{
   "AttributeMapping": {
            "email": "idp_email",
            "email_verified": "idp_email_verified",
            "username": "sub"
    },
    "CreationDate": 1.701129701653E9,
    "IdpIdentifiers": [
        "corp",
        "dev"
    ],
    "LastModifiedDate": 1.701129701653E9,
    "ProviderDetails": {
        "attributes_request_method": "GET",
        "attributes_url": "https://example.com/userInfo",
        "attributes_url_add_attributes": "false",
        "authorize_scopes": "openid profile",
        "authorize_url": "https://example.com/authorize",
        "client_id": "idpexampleclient123",
```

```
        "client_secret": "idpexamplesecret456",
        "jwks_uri": "https://example.com/.well-known/jwks.json",
        "oidc_issuer": "https://example.com",
        "token_url": "https://example.com/token"
    },
    "ProviderName": "MyOIDCIdP",
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

输出：

```
{
    "IdentityProvider": {
        "AttributeMapping": {
            "email": "idp_email",
            "email_verified": "idp_email_verified",
            "username": "sub"
        },
        "CreationDate": 1701129701.653,
        "IdpIdentifiers": [
            "corp",
            "dev"
        ],
        "LastModifiedDate": 1736444278.211,
        "ProviderDetails": {
            "attributes_request_method": "GET",
            "attributes_url": "https://example.com/userInfo",
            "attributes_url_add_attributes": "false",
            "authorize_scopes": "openid profile",
            "authorize_url": "https://example.com/authorize",
            "client_id": "idpexampleclient123",
            "client_secret": "idpexamplesecret456",
            "jwks_uri": "https://example.com/.well-known/jwks.json",
            "oidc_issuer": "https://example.com",
            "token_url": "https://example.com/token"
        },
        "ProviderName": "MyOIDCIdP",
        "ProviderType": "OIDC",
        "UserPoolId": "us-west-2_EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Configuring a domain。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateIdentityProvider。

update-managed-login-branding

以下代码示例演示了如何使用 update-managed-login-branding。

Amazon CLI

更新托管式登录品牌风格

以下 update-managed-login-branding 示例更新了所请求的应用程序客户端品牌风格。

```
aws cognito-idp update-managed-login-branding \
    --cli-input-json file://update-managed-login-branding.json
```

的内容update-managed-login-branding.json：

```
{
    "Assets": [
        {
            "Bytes":
 "PHN2ZyB3aWR0aD0iMjAwMDAiIGhlaWdodD0iNDAwIiB2aWV3Qm94PSIwIDAgMjAwMDAgNDAwIiBmaWxsPSJub25lIi
+CjxyZWN0IHdpZHRoPSIyMDAwMCIgaGVpZ2h0PSI0MDAiIGZpbGw9InVybCgjcGFpbnQwX2xpbmVhcl8xNzI1OV8yMzY
+CjxsaW5lYXJHcmFkaWVudCBpZD0icGFpbnQwX2xpbmVhcl8xNzI1OV8yMzY2NzQiIHgxPSItODk0LjI0OSIgeTE9IjE
+Cjwvc3ZnPgo=",
            "Category": "PAGE_FOOTER_BACKGROUND",
            "ColorMode": "DARK",
            "Extension": "SVG"
        }
    ],
    "ManagedLoginBrandingId": "63f30090-6b1f-4278-b885-2bbb81f8e545",
    "Settings": {
        "categories": {
            "auth": {
                "authMethodOrder": [
                    [
                        {
                            "display": "BUTTON",
                            "type": "FEDERATED"
                        },
                        {
                            "display": "INPUT",
```

```
                        "type": "USERNAME_PASSWORD"
                    }
                ]
            ],
            "federation": {
                "interfaceStyle": "BUTTON_LIST",
                "order": [
                ]
            }
        },
        "form": {
            "displayGraphics": true,
            "instructions": {
                "enabled": false
            },
            "languageSelector": {
                "enabled": false
            },
            "location": {
                "horizontal": "CENTER",
                "vertical": "CENTER"
            },
            "sessionTimerDisplay": "NONE"
        },
        "global": {
            "colorSchemeMode": "LIGHT",
            "pageFooter": {
                "enabled": false
            },
            "pageHeader": {
                "enabled": false
            },
            "spacingDensity": "REGULAR"
        },
        "signUp": {
            "acceptanceElements": [
                {
                    "enforcement": "NONE",
                    "textKey": "en"
                }
            ]
        }
    },
    "componentClasses": {
```

```
            "buttons": {
                "borderRadius": 8.0
            },
            "divider": {
                "darkMode": {
                    "borderColor": "232b37ff"
                },
                "lightMode": {
                    "borderColor": "ebebf0ff"
                }
            },
            "dropDown": {
                "borderRadius": 8.0,
                "darkMode": {
                    "defaults": {
                        "itemBackgroundColor": "192534ff"
                    },
                    "hover": {
                        "itemBackgroundColor": "081120ff",
                        "itemBorderColor": "5f6b7aff",
                        "itemTextColor": "e9ebedff"
                    },
                    "match": {
                        "itemBackgroundColor": "d1d5dbff",
                        "itemTextColor": "89bdeeff"
                    }
                },
                "lightMode": {
                    "defaults": {
                        "itemBackgroundColor": "ffffffff"
                    },
                    "hover": {
                        "itemBackgroundColor": "f4f4f4ff",
                        "itemBorderColor": "7d8998ff",
                        "itemTextColor": "000716ff"
                    },
                    "match": {
                        "itemBackgroundColor": "414d5cff",
                        "itemTextColor": "0972d3ff"
                    }
                }
            },
            "focusState": {
                "darkMode": {
```

```
                "borderColor": "539fe5ff"
            },
            "lightMode": {
                "borderColor": "0972d3ff"
            }
        },
        "idpButtons": {
            "icons": {
                "enabled": true
            }
        },
        "input": {
            "borderRadius": 8.0,
            "darkMode": {
                "defaults": {
                    "backgroundColor": "0f1b2aff",
                    "borderColor": "5f6b7aff"
                },
                "placeholderColor": "8d99a8ff"
            },
            "lightMode": {
                "defaults": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "7d8998ff"
                },
                "placeholderColor": "5f6b7aff"
            }
        },
        "inputDescription": {
            "darkMode": {
                "textColor": "8d99a8ff"
            },
            "lightMode": {
                "textColor": "5f6b7aff"
            }
        },
        "inputLabel": {
            "darkMode": {
                "textColor": "d1d5dbff"
            },
            "lightMode": {
                "textColor": "000716ff"
            }
        },
```

```
            "link": {
                "darkMode": {
                    "defaults": {
                        "textColor": "539fe5ff"
                    },
                    "hover": {
                        "textColor": "89bdeeff"
                    }
                },
                "lightMode": {
                    "defaults": {
                        "textColor": "0972d3ff"
                    },
                    "hover": {
                        "textColor": "033160ff"
                    }
                }
            },
            "optionControls": {
                "darkMode": {
                    "defaults": {
                        "backgroundColor": "0f1b2aff",
                        "borderColor": "7d8998ff"
                    },
                    "selected": {
                        "backgroundColor": "539fe5ff",
                        "foregroundColor": "000716ff"
                    }
                },
                "lightMode": {
                    "defaults": {
                        "backgroundColor": "ffffffff",
                        "borderColor": "7d8998ff"
                    },
                    "selected": {
                        "backgroundColor": "0972d3ff",
                        "foregroundColor": "ffffffff"
                    }
                }
            },
            "statusIndicator": {
                "darkMode": {
                    "error": {
                        "backgroundColor": "1a0000ff",
```

```
                    "borderColor": "eb6f6fff",
                    "indicatorColor": "eb6f6fff"
                },
                "pending": {
                    "indicatorColor": "AAAAAAAA"
                },
                "success": {
                    "backgroundColor": "001a02ff",
                    "borderColor": "29ad32ff",
                    "indicatorColor": "29ad32ff"
                },
                "warning": {
                    "backgroundColor": "1d1906ff",
                    "borderColor": "e0ca57ff",
                    "indicatorColor": "e0ca57ff"
                }
            },
            "lightMode": {
                "error": {
                    "backgroundColor": "fff7f7ff",
                    "borderColor": "d91515ff",
                    "indicatorColor": "d91515ff"
                },
                "pending": {
                    "indicatorColor": "AAAAAAAA"
                },
                "success": {
                    "backgroundColor": "f2fcf3ff",
                    "borderColor": "037f0cff",
                    "indicatorColor": "037f0cff"
                },
                "warning": {
                    "backgroundColor": "fffce9ff",
                    "borderColor": "8d6605ff",
                    "indicatorColor": "8d6605ff"
                }
            }
        }
    },
    "components": {
        "alert": {
            "borderRadius": 12.0,
            "darkMode": {
                "error": {
```

```
                    "backgroundColor": "1a0000ff",
                    "borderColor": "eb6f6fff"
                }
            },
            "lightMode": {
                "error": {
                    "backgroundColor": "fff7f7ff",
                    "borderColor": "d91515ff"
                }
            }
        },
        "favicon": {
            "enabledTypes": [
                "ICO",
                "SVG"
            ]
        },
        "form": {
            "backgroundImage": {
                "enabled": false
            },
            "borderRadius": 8.0,
            "darkMode": {
                "backgroundColor": "0f1b2aff",
                "borderColor": "424650ff"
            },
            "lightMode": {
                "backgroundColor": "ffffffff",
                "borderColor": "c6c6cdff"
            },
            "logo": {
                "enabled": false,
                "formInclusion": "IN",
                "location": "CENTER",
                "position": "TOP"
            }
        },
        "idpButton": {
            "custom": {
            },
            "standard": {
                "darkMode": {
                    "active": {
                        "backgroundColor": "354150ff",
```

```
                        "borderColor": "89bdeeff",
                        "textColor": "89bdeeff"
                    },
                    "defaults": {
                        "backgroundColor": "0f1b2aff",
                        "borderColor": "c6c6cdff",
                        "textColor": "c6c6cdff"
                    },
                    "hover": {
                        "backgroundColor": "192534ff",
                        "borderColor": "89bdeeff",
                        "textColor": "89bdeeff"
                    }
                },
                "lightMode": {
                    "active": {
                        "backgroundColor": "d3e7f9ff",
                        "borderColor": "033160ff",
                        "textColor": "033160ff"
                    },
                    "defaults": {
                        "backgroundColor": "ffffffff",
                        "borderColor": "424650ff",
                        "textColor": "424650ff"
                    },
                    "hover": {
                        "backgroundColor": "f2f8fdff",
                        "borderColor": "033160ff",
                        "textColor": "033160ff"
                    }
                }
            }
        },
        "pageBackground": {
            "darkMode": {
                "color": "0f1b2aff"
            },
            "image": {
                "enabled": true
            },
            "lightMode": {
                "color": "ffffffff"
            }
        },
```

```
            "pageFooter": {
                "backgroundImage": {
                    "enabled": false
                },
                "darkMode": {
                    "background": {
                        "color": "0f141aff"
                    },
                    "borderColor": "424650ff"
                },
                "lightMode": {
                    "background": {
                        "color": "fafafaff"
                    },
                    "borderColor": "d5dbdbff"
                },
                "logo": {
                    "enabled": false,
                    "location": "START"
                }
            },
            "pageHeader": {
                "backgroundImage": {
                    "enabled": false
                },
                "darkMode": {
                    "background": {
                        "color": "0f141aff"
                    },
                    "borderColor": "424650ff"
                },
                "lightMode": {
                    "background": {
                        "color": "fafafaff"
                    },
                    "borderColor": "d5dbdbff"
                },
                "logo": {
                    "enabled": false,
                    "location": "START"
                }
            },
            "pageText": {
                "darkMode": {
```

```
                "bodyColor": "b6bec9ff",
                "descriptionColor": "b6bec9ff",
                "headingColor": "d1d5dbff"
            },
            "lightMode": {
                "bodyColor": "414d5cff",
                "descriptionColor": "414d5cff",
                "headingColor": "000716ff"
            }
        },
        "phoneNumberSelector": {
            "displayType": "TEXT"
        },
        "primaryButton": {
            "darkMode": {
                "active": {
                    "backgroundColor": "539fe5ff",
                    "textColor": "000716ff"
                },
                "defaults": {
                    "backgroundColor": "539fe5ff",
                    "textColor": "000716ff"
                },
                "disabled": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "ffffffff"
                },
                "hover": {
                    "backgroundColor": "89bdeeff",
                    "textColor": "000716ff"
                }
            },
            "lightMode": {
                "active": {
                    "backgroundColor": "033160ff",
                    "textColor": "ffffffff"
                },
                "defaults": {
                    "backgroundColor": "0972d3ff",
                    "textColor": "ffffffff"
                },
                "disabled": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "ffffffff"
```

```
                },
                "hover": {
                    "backgroundColor": "033160ff",
                    "textColor": "ffffffff"
                }
            }
        },
        "secondaryButton": {
            "darkMode": {
                "active": {
                    "backgroundColor": "354150ff",
                    "borderColor": "89bdeeff",
                    "textColor": "89bdeeff"
                },
                "defaults": {
                    "backgroundColor": "0f1b2aff",
                    "borderColor": "539fe5ff",
                    "textColor": "539fe5ff"
                },
                "hover": {
                    "backgroundColor": "192534ff",
                    "borderColor": "89bdeeff",
                    "textColor": "89bdeeff"
                }
            },
            "lightMode": {
                "active": {
                    "backgroundColor": "d3e7f9ff",
                    "borderColor": "033160ff",
                    "textColor": "033160ff"
                },
                "defaults": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "0972d3ff",
                    "textColor": "0972d3ff"
                },
                "hover": {
                    "backgroundColor": "f2f8fdff",
                    "borderColor": "033160ff",
                    "textColor": "033160ff"
                }
            }
        }
    }
```

```
    },
    "UseCognitoProvidedValues": false,
    "UserPoolId": "ca-central-1_EXAMPLE"
}
```

输出：

```
{
    "ManagedLoginBranding": {
        "Assets": [
            {
                "Bytes":
 "PHN2ZyB3aWR0aD0iMjAwMDAiIGhlaWdodD0iNDAwIiB2aWV3Qm94PSIwIDAgMjAwMDAgNDAwIiBmaWxsPSJub25lIi
+CjxyZWN0IHdpZHRoPSIyMDAwMCIgaGVpZ2h0PSI0MDAiIGZpbGw9InVybCgjcGFpbnQwX2xpbmVhcl8xNzI1OV8yMzY
+CjxsaW5lYXJHcmFkaWVudCBpZD0icGFpbnQwX2xpbmVhcl8xNzI1OV8yMzY2NzQiIHgxPSItODk0LjI0OSIgeTE9IjE
+Cjwvc3ZnPgo=",
                "Category": "PAGE_FOOTER_BACKGROUND",
                "ColorMode": "DARK",
                "Extension": "SVG"
            }
        ],
        "CreationDate": 1732138490.642,
        "LastModifiedDate": 1732140420.301,
        "ManagedLoginBrandingId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Settings": {
            "categories": {
                "auth": {
                    "authMethodOrder": [
                        [
                            {
                                "display": "BUTTON",
                                "type": "FEDERATED"
                            },
                            {
                                "display": "INPUT",
                                "type": "USERNAME_PASSWORD"
                            }
                        ]
                    ],
                    "federation": {
                        "interfaceStyle": "BUTTON_LIST",
                        "order": [
                        ]
```

```
                    }
                },
                "form": {
                    "displayGraphics": true,
                    "instructions": {
                        "enabled": false
                    },
                    "languageSelector": {
                        "enabled": false
                    },
                    "location": {
                        "horizontal": "CENTER",
                        "vertical": "CENTER"
                    },
                    "sessionTimerDisplay": "NONE"
                },
                "global": {
                    "colorSchemeMode": "LIGHT",
                    "pageFooter": {
                        "enabled": false
                    },
                    "pageHeader": {
                        "enabled": false
                    },
                    "spacingDensity": "REGULAR"
                },
                "signUp": {
                    "acceptanceElements": [
                        {
                            "enforcement": "NONE",
                            "textKey": "en"
                        }
                    ]
                }
            },
            "componentClasses": {
                "buttons": {
                    "borderRadius": 8.0
                },
                "divider": {
                    "darkMode": {
                        "borderColor": "232b37ff"
                    },
                    "lightMode": {
```

```
                "borderColor": "ebebf0ff"
            }
        },
        "dropDown": {
            "borderRadius": 8.0,
            "darkMode": {
                "defaults": {
                    "itemBackgroundColor": "192534ff"
                },
                "hover": {
                    "itemBackgroundColor": "081120ff",
                    "itemBorderColor": "5f6b7aff",
                    "itemTextColor": "e9ebedff"
                },
                "match": {
                    "itemBackgroundColor": "d1d5dbff",
                    "itemTextColor": "89bdeeff"
                }
            },
            "lightMode": {
                "defaults": {
                    "itemBackgroundColor": "ffffffff"
                },
                "hover": {
                    "itemBackgroundColor": "f4f4f4ff",
                    "itemBorderColor": "7d8998ff",
                    "itemTextColor": "000716ff"
                },
                "match": {
                    "itemBackgroundColor": "414d5cff",
                    "itemTextColor": "0972d3ff"
                }
            }
        },
        "focusState": {
            "darkMode": {
                "borderColor": "539fe5ff"
            },
            "lightMode": {
                "borderColor": "0972d3ff"
            }
        },
        "idpButtons": {
            "icons": {
```

```
                "enabled": true
            }
        },
        "input": {
            "borderRadius": 8.0,
            "darkMode": {
                "defaults": {
                    "backgroundColor": "0f1b2aff",
                    "borderColor": "5f6b7aff"
                },
                "placeholderColor": "8d99a8ff"
            },
            "lightMode": {
                "defaults": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "7d8998ff"
                },
                "placeholderColor": "5f6b7aff"
            }
        },
        "inputDescription": {
            "darkMode": {
                "textColor": "8d99a8ff"
            },
            "lightMode": {
                "textColor": "5f6b7aff"
            }
        },
        "inputLabel": {
            "darkMode": {
                "textColor": "d1d5dbff"
            },
            "lightMode": {
                "textColor": "000716ff"
            }
        },
        "link": {
            "darkMode": {
                "defaults": {
                    "textColor": "539fe5ff"
                },
                "hover": {
                    "textColor": "89bdeeff"
                }
```

```
                    },
                    "lightMode": {
                        "defaults": {
                            "textColor": "0972d3ff"
                        },
                        "hover": {
                            "textColor": "033160ff"
                        }
                    }
                },
                "optionControls": {
                    "darkMode": {
                        "defaults": {
                            "backgroundColor": "0f1b2aff",
                            "borderColor": "7d8998ff"
                        },
                        "selected": {
                            "backgroundColor": "539fe5ff",
                            "foregroundColor": "000716ff"
                        }
                    },
                    "lightMode": {
                        "defaults": {
                            "backgroundColor": "ffffffff",
                            "borderColor": "7d8998ff"
                        },
                        "selected": {
                            "backgroundColor": "0972d3ff",
                            "foregroundColor": "ffffffff"
                        }
                    }
                },
                "statusIndicator": {
                    "darkMode": {
                        "error": {
                            "backgroundColor": "1a0000ff",
                            "borderColor": "eb6f6fff",
                            "indicatorColor": "eb6f6fff"
                        },
                        "pending": {
                            "indicatorColor": "AAAAAAAA"
                        },
                        "success": {
                            "backgroundColor": "001a02ff",
```

```
                        "borderColor": "29ad32ff",
                        "indicatorColor": "29ad32ff"
                    },
                    "warning": {
                        "backgroundColor": "1d1906ff",
                        "borderColor": "e0ca57ff",
                        "indicatorColor": "e0ca57ff"
                    }
                },
                "lightMode": {
                    "error": {
                        "backgroundColor": "fff7f7ff",
                        "borderColor": "d91515ff",
                        "indicatorColor": "d91515ff"
                    },
                    "pending": {
                        "indicatorColor": "AAAAAAAA"
                    },
                    "success": {
                        "backgroundColor": "f2fcf3ff",
                        "borderColor": "037f0cff",
                        "indicatorColor": "037f0cff"
                    },
                    "warning": {
                        "backgroundColor": "fffce9ff",
                        "borderColor": "8d6605ff",
                        "indicatorColor": "8d6605ff"
                    }
                }
            }
        },
        "components": {
            "alert": {
                "borderRadius": 12.0,
                "darkMode": {
                    "error": {
                        "backgroundColor": "1a0000ff",
                        "borderColor": "eb6f6fff"
                    }
                },
                "lightMode": {
                    "error": {
                        "backgroundColor": "fff7f7ff",
                        "borderColor": "d91515ff"
```

```
                    }
                }
            },
            "favicon": {
                "enabledTypes": [
                    "ICO",
                    "SVG"
                ]
            },
            "form": {
                "backgroundImage": {
                    "enabled": false
                },
                "borderRadius": 8.0,
                "darkMode": {
                    "backgroundColor": "0f1b2aff",
                    "borderColor": "424650ff"
                },
                "lightMode": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "c6c6cdff"
                },
                "logo": {
                    "enabled": false,
                    "formInclusion": "IN",
                    "location": "CENTER",
                    "position": "TOP"
                }
            },
            "idpButton": {
                "custom": {
                },
                "standard": {
                    "darkMode": {
                        "active": {
                            "backgroundColor": "354150ff",
                            "borderColor": "89bdeeff",
                            "textColor": "89bdeeff"
                        },
                        "defaults": {
                            "backgroundColor": "0f1b2aff",
                            "borderColor": "c6c6cdff",
                            "textColor": "c6c6cdff"
                        },
```

```
                        "hover": {
                            "backgroundColor": "192534ff",
                            "borderColor": "89bdeeff",
                            "textColor": "89bdeeff"
                        }
                    },
                    "lightMode": {
                        "active": {
                            "backgroundColor": "d3e7f9ff",
                            "borderColor": "033160ff",
                            "textColor": "033160ff"
                        },
                        "defaults": {
                            "backgroundColor": "ffffffff",
                            "borderColor": "424650ff",
                            "textColor": "424650ff"
                        },
                        "hover": {
                            "backgroundColor": "f2f8fdff",
                            "borderColor": "033160ff",
                            "textColor": "033160ff"
                        }
                    }
                }
            },
            "pageBackground": {
                "darkMode": {
                    "color": "0f1b2aff"
                },
                "image": {
                    "enabled": true
                },
                "lightMode": {
                    "color": "ffffffff"
                }
            },
            "pageFooter": {
                "backgroundImage": {
                    "enabled": false
                },
                "darkMode": {
                    "background": {
                        "color": "0f141aff"
                    },
```

```
                    "borderColor": "424650ff"
                },
                "lightMode": {
                    "background": {
                        "color": "fafafaff"
                    },
                    "borderColor": "d5dbdbff"
                },
                "logo": {
                    "enabled": false,
                    "location": "START"
                }
            },
            "pageHeader": {
                "backgroundImage": {
                    "enabled": false
                },
                "darkMode": {
                    "background": {
                        "color": "0f141aff"
                    },
                    "borderColor": "424650ff"
                },
                "lightMode": {
                    "background": {
                        "color": "fafafaff"
                    },
                    "borderColor": "d5dbdbff"
                },
                "logo": {
                    "enabled": false,
                    "location": "START"
                }
            },
            "pageText": {
                "darkMode": {
                    "bodyColor": "b6bec9ff",
                    "descriptionColor": "b6bec9ff",
                    "headingColor": "d1d5dbff"
                },
                "lightMode": {
                    "bodyColor": "414d5cff",
                    "descriptionColor": "414d5cff",
                    "headingColor": "000716ff"
```

```
            }
        },
        "phoneNumberSelector": {
            "displayType": "TEXT"
        },
        "primaryButton": {
            "darkMode": {
                "active": {
                    "backgroundColor": "539fe5ff",
                    "textColor": "000716ff"
                },
                "defaults": {
                    "backgroundColor": "539fe5ff",
                    "textColor": "000716ff"
                },
                "disabled": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "ffffffff"
                },
                "hover": {
                    "backgroundColor": "89bdeeff",
                    "textColor": "000716ff"
                }
            },
            "lightMode": {
                "active": {
                    "backgroundColor": "033160ff",
                    "textColor": "ffffffff"
                },
                "defaults": {
                    "backgroundColor": "0972d3ff",
                    "textColor": "ffffffff"
                },
                "disabled": {
                    "backgroundColor": "ffffffff",
                    "borderColor": "ffffffff"
                },
                "hover": {
                    "backgroundColor": "033160ff",
                    "textColor": "ffffffff"
                }
            }
        },
        "secondaryButton": {
```

```
                    "darkMode": {
                        "active": {
                            "backgroundColor": "354150ff",
                            "borderColor": "89bdeeff",
                            "textColor": "89bdeeff"
                        },
                        "defaults": {
                            "backgroundColor": "0f1b2aff",
                            "borderColor": "539fe5ff",
                            "textColor": "539fe5ff"
                        },
                        "hover": {
                            "backgroundColor": "192534ff",
                            "borderColor": "89bdeeff",
                            "textColor": "89bdeeff"
                        }
                    },
                    "lightMode": {
                        "active": {
                            "backgroundColor": "d3e7f9ff",
                            "borderColor": "033160ff",
                            "textColor": "033160ff"
                        },
                        "defaults": {
                            "backgroundColor": "ffffffff",
                            "borderColor": "0972d3ff",
                            "textColor": "0972d3ff"
                        },
                        "hover": {
                            "backgroundColor": "f2f8fdff",
                            "borderColor": "033160ff",
                            "textColor": "033160ff"
                        }
                    }
                }
            },
            "UseCognitoProvidedValues": false,
            "UserPoolId": "ca-central-1_EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Apply branding to managed login pages。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateManagedLoginBranding。

`update-resource-server`

以下代码示例演示了如何使用 `update-resource-server`。

Amazon CLI

更新资源服务器

此示例更新资源服务器 Weather。它添加了一个新范围。

命令:

```
aws cognito-idp update-resource-server --user-pool-id us-west-2_aaaaaaaaa
 --identifier weather.example.com --name Weather --scopes
 ScopeName=NewScope,ScopeDescription="New scope description"
```

输出：

```
{
  "ResourceServer": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "Identifier": "weather.example.com",
      "Name": "Happy",
      "Scopes": [
          {
              "ScopeName": "NewScope",
              "ScopeDescription": "New scope description"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResourceServer。

update-user-attributes

以下代码示例演示了如何使用 update-user-attributes。

Amazon CLI

更新用户属性

此示例更新用户属性“nickname”。

命令:

```
aws cognito-idp update-user-attributes --access-token ACCESS_TOKEN --user-attributes
 Name="nickname",Value="Dan"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserAttributes。

update-user-pool-client

以下代码示例演示了如何使用 update-user-pool-client。

Amazon CLI

更新应用程序客户端

以下 update-user-pool-client 示例更新了所请求的应用程序客户端的配置。

```
aws cognito-idp update-user-pool-client \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --client-name my-test-app \
    --refresh-token-validity 30 \
    --access-token-validity 60 \
    --id-token-validity 60 \
    --token-validity-units AccessToken=minutes,IdToken=minutes,RefreshToken=days \
    --read-
attributes "address" "birthdate" "email" "email_verified" "family_name" "gender" "locale" "m
 \
    --write-
attributes "address" "birthdate" "email" "family_name" "gender" "locale" "middle_name" "name
 \
```

```
    --explicit-auth-
flows "ALLOW_ADMIN_USER_PASSWORD_AUTH" "ALLOW_CUSTOM_AUTH" "ALLOW_REFRESH_TOKEN_AUTH" "ALLOW
 \
    --supported-identity-providers "MySAML" "COGNITO" "Google" \
    --callback-urls "https://www.example.com" "https://app2.example.com" \
    --logout-urls "https://auth.example.com/login?
client_id=1example23456789&response_type=code&redirect_uri=https%3A%2F
%2Fwww.example.com" "https://example.com/logout" \
    --default-redirect-uri "https://www.example.com" \
    --allowed-o-auth-flows "code" "implicit" \
    --allowed-o-auth-scopes "openid" "profile" "aws.cognito.signin.user.admin" \
    --allowed-o-auth-flows-user-pool-client \
    --prevent-user-existence-errors ENABLED \
    --enable-token-revocation \
    --no-enable-propagate-additional-user-context-data \
    --auth-session-validity 3
```

输出：

```
{
    "UserPoolClient": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ClientName": "my-test-app",
        "ClientId": "1example23456789",
        "LastModifiedDate": "2025-01-31T14:40:12.498000-08:00",
        "CreationDate": "2023-09-13T16:26:34.408000-07:00",
        "RefreshTokenValidity": 30,
        "AccessTokenValidity": 60,
        "IdTokenValidity": 60,
        "TokenValidityUnits": {
            "AccessToken": "minutes",
            "IdToken": "minutes",
            "RefreshToken": "days"
        },
        "ReadAttributes": [
            "website",
            "zoneinfo",
            "address",
            "birthdate",
            "email_verified",
            "gender",
            "profile",
            "phone_number_verified",
```

```
            "preferred_username",
            "locale",
            "middle_name",
            "picture",
            "updated_at",
            "name",
            "nickname",
            "phone_number",
            "family_name",
            "email"
        ],
        "WriteAttributes": [
            "website",
            "zoneinfo",
            "address",
            "birthdate",
            "gender",
            "profile",
            "preferred_username",
            "locale",
            "middle_name",
            "picture",
            "updated_at",
            "name",
            "nickname",
            "phone_number",
            "family_name",
            "email"
        ],
        "ExplicitAuthFlows": [
            "ALLOW_CUSTOM_AUTH",
            "ALLOW_USER_PASSWORD_AUTH",
            "ALLOW_ADMIN_USER_PASSWORD_AUTH",
            "ALLOW_USER_SRP_AUTH",
            "ALLOW_REFRESH_TOKEN_AUTH"
        ],
        "SupportedIdentityProviders": [
            "Google",
            "COGNITO",
            "MySAML"
        ],
        "CallbackURLs": [
            "https://www.example.com",
            "https://app2.example.com"
```

```
        ],
        "LogoutURLs": [
            "https://example.com/logout",
            "https://auth.example.com/login?
client_id=1example23456789&response_type=code&redirect_uri=https%3A%2F
%2Fwww.example.com"
        ],
        "DefaultRedirectURI": "https://www.example.com",
        "AllowedOAuthFlows": [
            "implicit",
            "code"
        ],
        "AllowedOAuthScopes": [
            "aws.cognito.signin.user.admin",
            "openid",
            "profile"
        ],
        "AllowedOAuthFlowsUserPoolClient": true,
        "PreventUserExistenceErrors": "ENABLED",
        "EnableTokenRevocation": true,
        "EnablePropagateAdditionalUserContextData": false,
        "AuthSessionValidity": 3
    }
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Application-specific settings with app clients。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserPoolClient。

update-user-pool-domain

以下代码示例演示了如何使用 update-user-pool-domain。

Amazon CLI

更新自定义域

以下 update-user-pool-domain 示例为所请求的用户池中的自定义域配置品牌版本和证书。

```
aws cognito-idp update-user-pool-domain \
    --user-pool-id ca-central-1_EXAMPLE \
    --domain auth.example.com \
```

```
    --managed-login-version 2 \
    --custom-domain-config CertificateArn=arn:aws:acm:us-
east-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "CloudFrontDomain": "example.cloudfront.net",
    "ManagedLoginVersion": 2
}
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Managed login 和 Configuring a domain。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateUserPoolDomain。

update-user-pool

以下代码示例演示了如何使用 update-user-pool。

Amazon CLI

更新用户池

以下的 update-user-pool 示例使用每个可用配置选项的示例语法修改用户池。要更新用户池，必须指定所有先前配置的选项，否则这些选项将重置为默认值。

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_EXAMPLE \
    --policies PasswordPolicy=
\{MinimumLength=6,RequireUppercase=true,RequireLowercase=true,RequireNumbers=true,RequireSym
 \
    --deletion-protection ACTIVE \
    --lambda-config PreSignUp="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-presignup-
function",PreTokenGeneration="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-pretoken-function" \
    --auto-verified-attributes "phone_number" "email" \
    --verification-message-template \{\"SmsMessage\":\""Your code is
 {####}"\",\"EmailMessage\":\""Your code is {####}"\",\"EmailSubject\":\""Your
 verification code"\",\"EmailMessageByLink\":\""Click {##here##} to verify
 your email address."\",\"EmailSubjectByLink\":\""Your verification link"\",
\"DefaultEmailOption\":\"CONFIRM_WITH_LINK\"\} \
```

```
    --sms-authentication-message "Your code is {####}" \
    --user-attribute-update-settings
 AttributesRequireVerificationBeforeUpdate="email","phone_number" \
    --mfa-configuration "OPTIONAL" \
    --device-
configuration ChallengeRequiredOnNewDevice=true,DeviceOnlyRememberedOnUserPrompt=true
 \
    --email-configuration SourceArn="arn:aws:ses:us-
west-2:123456789012:identity/admin@example.com",ReplyToEmailAddress="amdin
+noreply@example.com",EmailSendingAccount=DEVELOPER,From="admin@amazon.com",ConfigurationSet
configuration-set" \
    --sms-configuration SnsCallerArn="arn:aws:iam::123456789012:role/service-role/
SNS-SMS-Role",ExternalId="12345",SnsRegion="us-west-2" \
    --admin-create-user-config AllowAdminCreateUserOnly=false,InviteMessageTemplate=
\{SMSMessage=\""Welcome {username}. Your confirmation code is
 {####}"\",EmailMessage=\""Welcome {username}. Your confirmation code is
 {####}"\",EmailSubject=\""Welcome to MyMobileGame"\"\} \
    --user-pool-tags "Function"="MyMobileGame","Developers"="Berlin" \
    --admin-create-user-config AllowAdminCreateUserOnly=false,InviteMessageTemplate=
\{SMSMessage=\""Welcome {username}. Your confirmation code is
 {####}"\",EmailMessage=\""Welcome {username}. Your confirmation code is
 {####}"\",EmailSubject=\""Welcome to MyMobileGame"\"\} \
    --user-pool-add-ons AdvancedSecurityMode="AUDIT" \
    --account-recovery-setting RecoveryMechanisms=
\[\{Priority=1,Name="verified_email"\},\{Priority=2,Name="verified_phone_number"\}\]
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Cognito 开发人员指南》中的更新用户池配置。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserPool。

verify-software-token

以下代码示例演示了如何使用 `verify-software-token`。

Amazon CLI

确认 TOTP 身份验证器的注册

以下 `verify-software-token` 示例完成了当前用户的 TOTP 注册。

```
aws cognito-idp verify-software-token \
    --access-token eyJra456defEXAMPLE \
```

```
    --user-code 123456
```

输出：

```
{
    "Status": "SUCCESS"
}
```

有关更多信息，请参阅《Amazon Cognito 开发人员指南》中的向用户池添加 MFA。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 VerifySoftwareToken。

`verify-user-attribute`

以下代码示例演示了如何使用 `verify-user-attribute`。

Amazon CLI

验证属性更改

以下 `verify-user-attribute` 示例验证了对当前用户的电子邮件属性的更改。

```
aws cognito-idp verify-user-attribute \
    --access-token eyJra456defEXAMPLE \
    --attribute-name email \
    --code 123456
```

有关更多信息，请参阅《Amazon Cognito Developer Guide》中的 Configuring email or phone verification。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 VerifyUserAttribute。

使用 Amazon CLI 的 Amazon Comprehend 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Comprehend 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-detect-dominant-language

以下代码示例演示了如何使用 batch-detect-dominant-language。

Amazon CLI

检测多个输入文本的主要语言

以下 batch-detect-dominant-language 示例分析多个输入文本并返回每个文本的主要语言。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend batch-detect-dominant-language \
    --text-list "Physics is the natural science that involves the study of matter and its motion and behavior through space and time, along with related concepts such as energy and force."
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "Languages": [
                {
                    "LanguageCode": "en",
                    "Score": 0.9986501932144165
                }
            ]
        }
    ],
    "ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的主要语言。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectDominantLanguage。

batch-detect-entities

以下代码示例演示了如何使用 batch-detect-entities。

Amazon CLI

检测来自多个输入文本的实体

以下 batch-detect-entities 示例分析多个输入文本并返回每个文本的命名实体。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend batch-detect-entities \
    --language-code en \
    --text-list "Dear Jane, Your AnyCompany Financial Services LLC credit card account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July 31st." "Please send customer feedback to Sunshine Spa, 123 Main St, Anywhere or to Alice at AnySpa@example.com."
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "Entities": [
                {
                    "Score": 0.9985517859458923,
                    "Type": "PERSON",
                    "Text": "Jane",
                    "BeginOffset": 5,
                    "EndOffset": 9
                },
                {
                    "Score": 0.9767839312553406,
                    "Type": "ORGANIZATION",
                    "Text": "AnyCompany Financial Services, LLC",
                    "BeginOffset": 16,
                    "EndOffset": 50
                },
                {
                    "Score": 0.9856694936752319,
                    "Type": "OTHER",
                    "Text": "1111-XXXX-1111-XXXX",
```

```
                    "BeginOffset": 71,
                    "EndOffset": 90
                },
                {
                    "Score": 0.9652159810066223,
                    "Type": "QUANTITY",
                    "Text": ".53",
                    "BeginOffset": 116,
                    "EndOffset": 119
                },
                {
                    "Score": 0.9986667037010193,
                    "Type": "DATE",
                    "Text": "July 31st",
                    "BeginOffset": 135,
                    "EndOffset": 144
                }
            ]
        },
        {
            "Index": 1,
            "Entities": [
                {
                    "Score": 0.720084547996521,
                    "Type": "ORGANIZATION",
                    "Text": "Sunshine Spa",
                    "BeginOffset": 33,
                    "EndOffset": 45
                },
                {
                    "Score": 0.9865870475769043,
                    "Type": "LOCATION",
                    "Text": "123 Main St",
                    "BeginOffset": 47,
                    "EndOffset": 58
                },
                {
                    "Score": 0.5895616412162781,
                    "Type": "LOCATION",
                    "Text": "Anywhere",
                    "BeginOffset": 60,
                    "EndOffset": 68
                },
                {
```

```
                    "Score": 0.6809214353561401,
                    "Type": "PERSON",
                    "Text": "Alice",
                    "BeginOffset": 75,
                    "EndOffset": 80
                },
                {
                    "Score": 0.9979087114334106,
                    "Type": "OTHER",
                    "Text": "AnySpa@example.com",
                    "BeginOffset": 84,
                    "EndOffset": 99
                }
            ]
        }
    ],
    "ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的实体。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectEntities。

batch-detect-key-phrases

以下代码示例演示了如何使用 batch-detect-key-phrases。

Amazon CLI

检测多个文本输入的关键短语

以下 batch-detect-key-phrases 示例分析多个输入文本并返回每个文本的关键名词短语。也会输出每个预测的预训练模型的置信度分数。

```
aws comprehend batch-detect-key-phrases \
    --language-code en \
    --text-list "Hello Zhang Wei, I am John, writing to you about the trip for next Saturday." "Dear Jane, Your AnyCompany Financial Services LLC credit card account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July 31st." "Please send customer feedback to Sunshine Spa, 123 Main St, Anywhere or to Alice at AnySpa@example.com."
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "KeyPhrases": [
                {
                    "Score": 0.99700927734375,
                    "Text": "Zhang Wei",
                    "BeginOffset": 6,
                    "EndOffset": 15
                },
                {
                    "Score": 0.9929308891296387,
                    "Text": "John",
                    "BeginOffset": 22,
                    "EndOffset": 26
                },
                {
                    "Score": 0.9997230172157288,
                    "Text": "the trip",
                    "BeginOffset": 49,
                    "EndOffset": 57
                },
                {
                    "Score": 0.9999470114707947,
                    "Text": "next Saturday",
                    "BeginOffset": 62,
                    "EndOffset": 75
                }
            ]
        },
        {
            "Index": 1,
            "KeyPhrases": [
                {
                    "Score": 0.8358274102210999,
                    "Text": "Dear Jane",
                    "BeginOffset": 0,
                    "EndOffset": 9
                },
                {
                    "Score": 0.989359974861145,
                    "Text": "Your AnyCompany Financial Services",
```

```
                    "BeginOffset": 11,
                    "EndOffset": 45
                },
                {
                    "Score": 0.8812323808670044,
                    "Text": "LLC credit card account 1111-XXXX-1111-XXXX",
                    "BeginOffset": 47,
                    "EndOffset": 90
                },
                {
                    "Score": 0.9999381899833679,
                    "Text": "a minimum payment",
                    "BeginOffset": 95,
                    "EndOffset": 112
                },
                {
                    "Score": 0.9997439980506897,
                    "Text": ".53",
                    "BeginOffset": 116,
                    "EndOffset": 119
                },
                {
                    "Score": 0.996875524520874,
                    "Text": "July 31st",
                    "BeginOffset": 135,
                    "EndOffset": 144
                }
            ]
        },
        {
            "Index": 2,
            "KeyPhrases": [
                {
                    "Score": 0.9990295767784119,
                    "Text": "customer feedback",
                    "BeginOffset": 12,
                    "EndOffset": 29
                },
                {
                    "Score": 0.9994127750396729,
                    "Text": "Sunshine Spa",
                    "BeginOffset": 33,
                    "EndOffset": 45
                },
```

```
                {
                    "Score": 0.9892991185188293,
                    "Text": "123 Main St",
                    "BeginOffset": 47,
                    "EndOffset": 58
                },
                {
                    "Score": 0.9969810843467712,
                    "Text": "Alice",
                    "BeginOffset": 75,
                    "EndOffset": 80
                },
                {
                    "Score": 0.9703696370124817,
                    "Text": "AnySpa@example.com",
                    "BeginOffset": 84,
                    "EndOffset": 99
                }
            ]
        }
    ],
    "ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的关键词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectKeyPhrases。

batch-detect-sentiment

以下代码示例演示了如何使用 batch-detect-sentiment。

Amazon CLI

检测多个输入文本的主导情绪

以下 batch-detect-sentiment 示例分析多个输入文本，并返回每个文本的主导情绪（POSITIVE、NEUTRAL、MIXED 或 NEGATIVE）。

```
aws comprehend batch-detect-sentiment \
    --text-list "That movie was very boring, I can't believe it was over four hours long." "It is a beautiful day for hiking today." "My meal was okay, I'm excited to try other restaurants." \
```

```
    --language-code en
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "Sentiment": "NEGATIVE",
            "SentimentScore": {
                "Positive": 0.00011316669406369328,
                "Negative": 0.9995445609092712,
                "Neutral": 0.00014722718333359808,
                "Mixed": 0.00019498742767609656
            }
        },
        {
            "Index": 1,
            "Sentiment": "POSITIVE",
            "SentimentScore": {
                "Positive": 0.9981263279914856,
                "Negative": 0.00015240783977787942,
                "Neutral": 0.0013876151060685515,
                "Mixed": 0.00033366199932061136
            }
        },
        {
            "Index": 2,
            "Sentiment": "MIXED",
            "SentimentScore": {
                "Positive": 0.15930435061454773,
                "Negative": 0.11471917480230331,
                "Neutral": 0.26897063851356506,
                "Mixed": 0.45700588822364807
            }
        }
    ],
    "ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的情绪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectSentiment。

batch-detect-syntax

以下代码示例演示了如何使用 batch-detect-syntax。

Amazon CLI

检查多个输入文本中单词的语法和语音部分

以下 batch-detect-syntax 示例分析多个输入文本的语法并返回语音的不同部分。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend batch-detect-syntax \
    --text-list "It is a beautiful day." "Can you please pass the salt?" "Please pay
 the bill before the 31st." \
    --language-code en
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "SyntaxTokens": [
                {
                    "TokenId": 1,
                    "Text": "It",
                    "BeginOffset": 0,
                    "EndOffset": 2,
                    "PartOfSpeech": {
                        "Tag": "PRON",
                        "Score": 0.9999740719795227
                    }
                },
                {
                    "TokenId": 2,
                    "Text": "is",
                    "BeginOffset": 3,
                    "EndOffset": 5,
                    "PartOfSpeech": {
                        "Tag": "VERB",
                        "Score": 0.999937117099762
                    }
                },
```

```
                {
                    "TokenId": 3,
                    "Text": "a",
                    "BeginOffset": 6,
                    "EndOffset": 7,
                    "PartOfSpeech": {
                        "Tag": "DET",
                        "Score": 0.9999926686286926
                    }
                },
                {
                    "TokenId": 4,
                    "Text": "beautiful",
                    "BeginOffset": 8,
                    "EndOffset": 17,
                    "PartOfSpeech": {
                        "Tag": "ADJ",
                        "Score": 0.9987891912460327
                    }
                },
                {
                    "TokenId": 5,
                    "Text": "day",
                    "BeginOffset": 18,
                    "EndOffset": 21,
                    "PartOfSpeech": {
                        "Tag": "NOUN",
                        "Score": 0.9999778866767883
                    }
                },
                {
                    "TokenId": 6,
                    "Text": ".",
                    "BeginOffset": 21,
                    "EndOffset": 22,
                    "PartOfSpeech": {
                        "Tag": "PUNCT",
                        "Score": 0.9999974966049194
                    }
                }
            ]
        },
        {
            "Index": 1,
```

```
            "SyntaxTokens": [
                {
                    "TokenId": 1,
                    "Text": "Can",
                    "BeginOffset": 0,
                    "EndOffset": 3,
                    "PartOfSpeech": {
                        "Tag": "AUX",
                        "Score": 0.9999770522117615
                    }
                },
                {
                    "TokenId": 2,
                    "Text": "you",
                    "BeginOffset": 4,
                    "EndOffset": 7,
                    "PartOfSpeech": {
                        "Tag": "PRON",
                        "Score": 0.9999986886978149
                    }
                },
                {
                    "TokenId": 3,
                    "Text": "please",
                    "BeginOffset": 8,
                    "EndOffset": 14,
                    "PartOfSpeech": {
                        "Tag": "INTJ",
                        "Score": 0.9681622385978699
                    }
                },
                {
                    "TokenId": 4,
                    "Text": "pass",
                    "BeginOffset": 15,
                    "EndOffset": 19,
                    "PartOfSpeech": {
                        "Tag": "VERB",
                        "Score": 0.9999874830245972
                    }
                },
                {
                    "TokenId": 5,
                    "Text": "the",
```

```
                    "BeginOffset": 20,
                    "EndOffset": 23,
                    "PartOfSpeech": {
                        "Tag": "DET",
                        "Score": 0.9999827146530151
                    }
                },
                {
                    "TokenId": 6,
                    "Text": "salt",
                    "BeginOffset": 24,
                    "EndOffset": 28,
                    "PartOfSpeech": {
                        "Tag": "NOUN",
                        "Score": 0.9995040893554688
                    }
                },
                {
                    "TokenId": 7,
                    "Text": "?",
                    "BeginOffset": 28,
                    "EndOffset": 29,
                    "PartOfSpeech": {
                        "Tag": "PUNCT",
                        "Score": 0.999998152256012
                    }
                }
            ]
        },
        {
            "Index": 2,
            "SyntaxTokens": [
                {
                    "TokenId": 1,
                    "Text": "Please",
                    "BeginOffset": 0,
                    "EndOffset": 6,
                    "PartOfSpeech": {
                        "Tag": "INTJ",
                        "Score": 0.9997857809066772
                    }
                },
                {
                    "TokenId": 2,
```

```
                    "Text": "pay",
                    "BeginOffset": 7,
                    "EndOffset": 10,
                    "PartOfSpeech": {
                        "Tag": "VERB",
                        "Score": 0.9999252557754517
                    }
                },
                {
                    "TokenId": 3,
                    "Text": "the",
                    "BeginOffset": 11,
                    "EndOffset": 14,
                    "PartOfSpeech": {
                        "Tag": "DET",
                        "Score": 0.9999842643737793
                    }
                },
                {
                    "TokenId": 4,
                    "Text": "bill",
                    "BeginOffset": 15,
                    "EndOffset": 19,
                    "PartOfSpeech": {
                        "Tag": "NOUN",
                        "Score": 0.9999588131904602
                    }
                },
                {
                    "TokenId": 5,
                    "Text": "before",
                    "BeginOffset": 20,
                    "EndOffset": 26,
                    "PartOfSpeech": {
                        "Tag": "ADP",
                        "Score": 0.9958304762840271
                    }
                },
                {
                    "TokenId": 6,
                    "Text": "the",
                    "BeginOffset": 27,
                    "EndOffset": 30,
                    "PartOfSpeech": {
```

```
                    "Tag": "DET",
                    "Score": 0.9999947547912598
                }
            },
            {
                "TokenId": 7,
                "Text": "31st",
                "BeginOffset": 31,
                "EndOffset": 35,
                "PartOfSpeech": {
                    "Tag": "NOUN",
                    "Score": 0.9924124479293823
                }
            },
            {
                "TokenId": 8,
                "Text": ".",
                "BeginOffset": 35,
                "EndOffset": 36,
                "PartOfSpeech": {
                    "Tag": "PUNCT",
                    "Score": 0.9999955892562866
                }
            }
        ]
    }
],
"ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的语法分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectSyntax。

batch-detect-targeted-sentiment

以下代码示例演示了如何使用 batch-detect-targeted-sentiment。

Amazon CLI

检测多个输入文本的情绪和每个命名实体

以下 batch-detect-targeted-sentiment 示例分析多个输入文本，并返回命名实体以及每个实体附带的主导情绪。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend batch-detect-targeted-sentiment \
    --language-code en \
    --text-list "That movie was really boring, the original was way more
 entertaining" "The trail is extra beautiful today." "My meal was just okay."
```

输出：

```
{
    "ResultList": [
        {
            "Index": 0,
            "Entities": [
                {
                    "DescriptiveMentionIndex": [
                        0
                    ],
                    "Mentions": [
                        {
                            "Score": 0.9999009966850281,
                            "GroupScore": 1.0,
                            "Text": "movie",
                            "Type": "MOVIE",
                            "MentionSentiment": {
                                "Sentiment": "NEGATIVE",
                                "SentimentScore": {
                                    "Positive": 0.13887299597263336,
                                    "Negative": 0.8057460188865662,
                                    "Neutral": 0.05525200068950653,
                                    "Mixed": 0.00012799999967683107
                                }
                            },
                            "BeginOffset": 5,
                            "EndOffset": 10
                        }
                    ]
                },
                {
                    "DescriptiveMentionIndex": [
                        0
                    ],
```

```
                "Mentions": [
                    {
                        "Score": 0.9921110272407532,
                        "GroupScore": 1.0,
                        "Text": "original",
                        "Type": "MOVIE",
                        "MentionSentiment": {
                            "Sentiment": "POSITIVE",
                            "SentimentScore": {
                                "Positive": 0.9999989867210388,
                                "Negative": 9.999999974752427e-07,
                                "Neutral": 0.0,
                                "Mixed": 0.0
                            }
                        },
                        "BeginOffset": 34,
                        "EndOffset": 42
                    }
                ]
            }
        ]
    },
    {
        "Index": 1,
        "Entities": [
            {
                "DescriptiveMentionIndex": [
                    0
                ],
                "Mentions": [
                    {
                        "Score": 0.7545599937438965,
                        "GroupScore": 1.0,
                        "Text": "trail",
                        "Type": "OTHER",
                        "MentionSentiment": {
                            "Sentiment": "POSITIVE",
                            "SentimentScore": {
                                "Positive": 1.0,
                                "Negative": 0.0,
                                "Neutral": 0.0,
                                "Mixed": 0.0
                            }
                        },
```

```
                        "BeginOffset": 4,
                        "EndOffset": 9
                    }
                ]
            },
            {
                "DescriptiveMentionIndex": [
                    0
                ],
                "Mentions": [
                    {
                        "Score": 0.9999960064888,
                        "GroupScore": 1.0,
                        "Text": "today",
                        "Type": "DATE",
                        "MentionSentiment": {
                            "Sentiment": "NEUTRAL",
                            "SentimentScore": {
                                "Positive": 9.000000318337698e-06,
                                "Negative": 1.9999999949504854e-06,
                                "Neutral": 0.9999859929084778,
                                "Mixed": 3.999999989900971e-06
                            }
                        },
                        "BeginOffset": 29,
                        "EndOffset": 34
                    }
                ]
            }
        ]
    },
    {
        "Index": 2,
        "Entities": [
            {
                "DescriptiveMentionIndex": [
                    0
                ],
                "Mentions": [
                    {
                        "Score": 0.9999880194664001,
                        "GroupScore": 1.0,
                        "Text": "My",
                        "Type": "PERSON",
```

```
                            "MentionSentiment": {
                                "Sentiment": "NEUTRAL",
                                "SentimentScore": {
                                    "Positive": 0.0,
                                    "Negative": 0.0,
                                    "Neutral": 1.0,
                                    "Mixed": 0.0
                                }
                            },
                            "BeginOffset": 0,
                            "EndOffset": 2
                        }
                    ]
                },
                {
                    "DescriptiveMentionIndex": [
                        0
                    ],
                    "Mentions": [
                        {
                            "Score": 0.9995260238647461,
                            "GroupScore": 1.0,
                            "Text": "meal",
                            "Type": "OTHER",
                            "MentionSentiment": {
                                "Sentiment": "NEUTRAL",
                                "SentimentScore": {
                                    "Positive": 0.04695599898695946,
                                    "Negative": 0.003226999891921878,
                                    "Neutral": 0.6091709733009338,
                                    "Mixed": 0.34064599871635437
                                }
                            },
                            "BeginOffset": 3,
                            "EndOffset": 7
                        }
                    ]
                }
            ]
        }
    ],
    "ErrorList": []
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的目标情绪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDetectTargetedSentiment。

`classify-document`

以下代码示例演示了如何使用 `classify-document`。

Amazon CLI

使用指定模型端点对文档进行分类

以下 `classify-document` 示例对带有自定义模型端点的文档进行分类。此示例中的模型是在包含标记为垃圾邮件、非垃圾邮件或“ham”短信的数据集中训练的。

```
aws comprehend classify-document \
    --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-endpoint/example-classifier-endpoint \
    --text "CONGRATULATIONS! TXT 1235550100 to win $5000"
```

输出：

```
{
    "Classes": [
        {
            "Name": "spam",
            "Score": 0.9998599290847778
        },
        {
            "Name": "ham",
            "Score": 0.00014001205272506922
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义分类。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ClassifyDocument。

`contains-pii-entities`

以下代码示例演示了如何使用 `contains-pii-entities`。

Amazon CLI

分析输入文本中是否存在 PII 信息

以下 contains-pii-entities 示例分析输入文本中是否存在个人身份信息（PII），并返回已识别的 PII 实体类型的标签，例如姓名、地址、银行账号或电话号码。

```
aws comprehend contains-pii-entities \
    --language-code en \
    --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC
 credit card
       account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by
 July 31st. Based on your autopay settings,
       we will withdraw your payment on the due date from your bank account number
 XXXXXX1111 with the routing number XXXXX0000.
       Customer feedback for Sunshine Spa, 100 Main St, Anywhere. Send comments to
 Alice at AnySpa@example.com."
```

输出：

```
{
    "Labels": [
        {
            "Name": "NAME",
            "Score": 1.0
        },
        {
            "Name": "EMAIL",
            "Score": 1.0
        },
        {
            "Name": "BANK_ACCOUNT_NUMBER",
            "Score": 0.9995794296264648
        },
        {
            "Name": "BANK_ROUTING",
            "Score": 0.9173126816749573
        },
        {
            "Name": "CREDIT_DEBIT_NUMBER",
            "Score": 1.0
        }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的个人身份信息（PII）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ContainsPiiEntities。

create-dataset

以下代码示例演示了如何使用 create-dataset。

Amazon CLI

创建飞轮数据集

以下 create-dataset 示例创建一个飞轮数据集。该数据集将用作 --dataset-type 标签指定的其他训练数据。

```
aws comprehend create-dataset \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-entity \
    --dataset-name example-dataset \
    --dataset-type "TRAIN" \
    --input-data-config file://inputConfig.json
```

的内容file://inputConfig.json：

```
{
    "DataFormat": "COMPREHEND_CSV",
    "DocumentClassifierInputDataConfig": {
        "S3Uri": "s3://amzn-s3-demo-bucket/training-data.csv"
    }
}
```

输出：

```
{
    "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-entity/dataset/example-dataset"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataset。

create-document-classifier

以下代码示例演示了如何使用 create-document-classifier。

Amazon CLI

创建文档分类器对文档进行分类

以下 create-document-classifier 示例启动文档分类器模型的训练过程。训练数据文件 training.csv 位于 --input-data-config 标签处。training.csv 是一个两列文档，其中第一列提供标签或分类，第二列提供文档。

```
aws comprehend create-document-classifier \
    --document-classifier-name example-classifier \
    --data-access-arn arn:aws:comprehend:us-west-2:111122223333:pii-entities-detection-job/123456abcdeb0e11022f22a11EXAMPLE \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
    --language-code en
```

输出：

```
{
    "DocumentClassifierArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义分类。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDocumentClassifier。

create-endpoint

以下代码示例演示了如何使用 create-endpoint。

Amazon CLI

为自定义模型创建端点

以下 create-endpoint 示例为之前训练的自定义模型的同步推理创建端点。

```
aws comprehend create-endpoint \
```

```
    --endpoint-name example-classifier-endpoint-1 \
    --model-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier \
    --desired-inference-units 1
```

输出：

```
{
    "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier-
endpoint/example-classifier-endpoint-1"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEndpoint。

create-entity-recognizer

以下代码示例演示了如何使用 create-entity-recognizer。

Amazon CLI

创建自定义实体识别器

以下 create-entity-recognizer 示例启动自定义实体识别器模型的训练过程。此示例使用包含训练文档 raw_text.csv 和 CSV 实体列表 entity_list.csv 的 CSV 文件来训练模型。entity-list.csv 包含以下列：文本和类型。

```
aws comprehend create-entity-recognizer \
    --recognizer-name example-entity-recognizer
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --input-data-config "EntityTypes=[{Type=DEVICE}],Documents={S3Uri=s3://amzn-s3-
demo-bucket/trainingdata/raw_text.csv},EntityList={S3Uri=s3://amzn-s3-demo-bucket/
trainingdata/entity_list.csv}"
    --language-code en
```

输出：

```
{
```

```
    "EntityRecognizerArn": "arn:aws:comprehend:us-west-2:111122223333:example-
entity-recognizer/entityrecognizer1"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义实体识别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEntityRecognizer。

`create-flywheel`

以下代码示例演示了如何使用 `create-flywheel`。

Amazon CLI

创建飞轮

以下 `create-flywheel` 示例创建一个飞轮来编排文档分类或实体识别模型的持续训练。此示例中的飞轮是为了管理 `--active-model-arn` 标签指定的现有训练模型。创建飞轮时，会在 `--input-data-lake` 标签处创建一个数据湖。

```
aws comprehend create-flywheel \
    --flywheel-name example-flywheel \
    --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-
classifier/example-model/version/1 \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --data-lake-s3-uri "s3://amzn-s3-demo-bucket"
```

输出：

```
{
    "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFlywheel。

`delete-document-classifier`

以下代码示例演示了如何使用 `delete-document-classifier`。

Amazon CLI

删除自定义文档分类器

以下 delete-document-classifier 示例删除了自定义文档分类器模型。

```
aws comprehend delete-document-classifier \
    --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDocumentClassifier。

delete-endpoint

以下代码示例演示了如何使用 delete-endpoint。

Amazon CLI

删除自定义模型的端点

以下 delete-endpoint 示例删除指定模型的端点。必须删除所有端点才能删除模型。

```
aws comprehend delete-endpoint \
    --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-endpoint/example-classifier-endpoint-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEndpoint。

delete-entity-recognizer

以下代码示例演示了如何使用 delete-entity-recognizer。

Amazon CLI

删除自定义实体识别器模型

以下 delete-entity-recognizer 示例删除自定义实体识别器模型。

```
aws comprehend delete-entity-recognizer \
    --entity-recognizer-arn arn:aws:comprehend:us-west-2:111122223333:entity-recognizer/example-entity-recognizer-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEntityRecognizer。

delete-flywheel

以下代码示例演示了如何使用 delete-flywheel。

Amazon CLI

删除飞轮

以下 delete-flywheel 示例删除飞轮。与该飞轮关联的数据湖或模型不会删除。

```
aws comprehend delete-flywheel \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-flywheel-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFlywheel。

delete-resource-policy

以下代码示例演示了如何使用 delete-resource-policy。

Amazon CLI

删除基于资源的策略

以下 delete-resource-policy 示例从 Amazon Comprehend 资源中删除基于资源的策略。

```
aws comprehend delete-resource-policy \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier-1/version/1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的在 Amazon 账户之间复制自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourcePolicy。

describe-dataset

以下代码示例演示了如何使用 describe-dataset。

Amazon CLI

描述飞轮数据集

以下 describe-dataset 示例获取飞轮数据集的属性。

```
aws comprehend describe-dataset \
    --dataset-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-
entity/dataset/example-dataset
```

输出：

```
{
    "DatasetProperties": {
        "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-
entity/dataset/example-dataset",
        "DatasetName": "example-dataset",
        "DatasetType": "TRAIN",
        "DatasetS3Uri": "s3://amzn-s3-demo-bucket/flywheel-entity/
schemaVersion=1/12345678A123456Z/datasets/example-dataset/20230616T203710Z/",
        "Status": "CREATING",
        "CreationTime": "2023-06-16T20:37:10.400000+00:00"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDataset。

describe-document-classification-job

以下代码示例演示了如何使用 describe-document-classification-job。

Amazon CLI

描述文档分类作业

以下 describe-document-classification-job 示例将获取异步文档分类作业的属性。

```
aws comprehend describe-document-classification-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "DocumentClassificationJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classification-job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "exampleclassificationjob",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2023-06-14T17:09:51.788000+00:00",
        "EndTime": "2023-06-14T17:15:58.582000+00:00",
        "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/mymodel/version/1",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/jobdata/",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
CLN-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义分类。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDocumentClassificationJob。

describe-document-classifier

以下代码示例演示了如何使用 describe-document-classifier。

Amazon CLI

描述文档分类器

以下 describe-document-classifier 示例将获取自定义文档分类器模型的属性。

```
aws comprehend describe-document-classifier \
    --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier-1
```

输出：

```
{
    "DocumentClassifierProperties": {
        "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/example-classifier-1",
        "LanguageCode": "en",
        "Status": "TRAINED",
        "SubmitTime": "2023-06-13T19:04:15.735000+00:00",
        "EndTime": "2023-06-13T19:42:31.752000+00:00",
        "TrainingStartTime": "2023-06-13T19:08:20.114000+00:00",
        "TrainingEndTime": "2023-06-13T19:41:35.080000+00:00",
        "InputDataConfig": {
            "DataFormat": "COMPREHEND_CSV",
            "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata"
        },
        "OutputDataConfig": {},
        "ClassifierMetadata": {
            "NumberOfLabels": 3,
            "NumberOfTrainedDocuments": 5016,
            "NumberOfTestDocuments": 557,
```

```
            "EvaluationMetrics": {
                "Accuracy": 0.9856,
                "Precision": 0.9919,
                "Recall": 0.9459,
                "F1Score": 0.9673,
                "MicroPrecision": 0.9856,
                "MicroRecall": 0.9856,
                "MicroF1Score": 0.9856,
                "HammingLoss": 0.0144
            }
        },
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
        "Mode": "MULTI_CLASS"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的创建和管理自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDocumentClassifier。

describe-dominant-language-detection-job

以下代码示例演示了如何使用 describe-dominant-language-detection-job。

Amazon CLI

描述主要语言检测作业。

以下 describe-dominant-language-detection-job 示例获取异步主要语言检测作业的属性。

```
aws comprehend describe-dominant-language-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "DominantLanguageDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "languageanalysis1",
```

```
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-09T18:10:38.037000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDominantLanguageDetectionJob。

describe-endpoint

以下代码示例演示了如何使用 describe-endpoint。

Amazon CLI

描述指定端点

以下 describe-endpoint 示例获取指定模型的端点属性。

```
aws comprehend describe-endpoint \
    --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-
endpoint/example-classifier-endpoint
```

输出：

```
{
    "EndpointProperties": {
        "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/example-classifier-endpoint",
        "Status": "IN_SERVICE",
```

```
        "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/
exampleclassifier1",
        "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
        "DesiredInferenceUnits": 1,
        "CurrentInferenceUnits": 1,
        "CreationTime": "2023-06-13T20:32:54.526000+00:00",
        "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpoint。

describe-entities-detection-job

以下代码示例演示了如何使用 describe-entities-detection-job。

Amazon CLI

描述实体检测作业

以下 describe-entities-detection-job 示例获取异步实体检测作业的属性。

```
aws comprehend describe-entities-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "EntitiesDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "example-entity-detector",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2023-06-08T21:30:15.323000+00:00",
        "EndTime": "2023-06-08T21:40:23.509000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
            "InputFormat": "ONE_DOC_PER_LINE"
```

```
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/thefolder/111122223333-
NER-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::12345678012:role/service-role/
AmazonComprehendServiceRole-example-role"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEntitiesDetectionJob。

`describe-entity-recognizer`

以下代码示例演示了如何使用 `describe-entity-recognizer`。

Amazon CLI

描述实体识别器

以下 `describe-entity-recognizer` 示例获取自定义实体识别器模型的属性。

```
aws comprehend describe-entity-recognizer \
    entity-recognizer-arn arn:aws:comprehend:us-west-2:111122223333:entity-
recognizer/business-recongizer-1/version/1
```

输出：

```
{
    "EntityRecognizerProperties": {
        "EntityRecognizerArn": "arn:aws:comprehend:us-west-2:111122223333:entity-
recognizer/business-recongizer-1/version/1",
        "LanguageCode": "en",
        "Status": "TRAINED",
        "SubmitTime": "2023-06-14T20:44:59.631000+00:00",
        "EndTime": "2023-06-14T20:59:19.532000+00:00",
        "TrainingStartTime": "2023-06-14T20:48:52.811000+00:00",
        "TrainingEndTime": "2023-06-14T20:58:11.473000+00:00",
```

```
        "InputDataConfig": {
            "DataFormat": "COMPREHEND_CSV",
            "EntityTypes": [
                {
                    "Type": "BUSINESS"
                }
            ],
            "Documents": {
                "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/dataset/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "EntityList": {
                "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/entity.csv"
            }
        },
        "RecognizerMetadata": {
            "NumberOfTrainedDocuments": 1814,
            "NumberOfTestDocuments": 486,
            "EvaluationMetrics": {
                "Precision": 100.0,
                "Recall": 100.0,
                "F1Score": 100.0
            },
            "EntityTypes": [
                {
                    "Type": "BUSINESS",
                    "EvaluationMetrics": {
                        "Precision": 100.0,
                        "Recall": 100.0,
                        "F1Score": 100.0
                    },
                    "NumberOfTrainMentions": 1520
                }
            ]
        },
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
        "VersionName": "1"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义实体识别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEntityRecognizer。

describe-events-detection-job

以下代码示例演示了如何使用 describe-events-detection-job。

Amazon CLI

描述事件检测作业。

以下 describe-events-detection-job 示例获取异步事件检测作业的属性。

```
aws comprehend describe-events-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "EventsDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:events-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "events_job_1",
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-12T18:45:56.054000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/EventsData",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
EVENTS-123456abcdeb0e11022f22a11EXAMPLE/output/"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
        "TargetEventTypes": [
            "BANKRUPTCY",
            "EMPLOYMENT",
            "CORPORATE_ACQUISITION",
            "CORPORATE_MERGER",
            "INVESTMENT_GENERAL"
        ]
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventsDetectionJob。

describe-flywheel-iteration

以下代码示例演示了如何使用 describe-flywheel-iteration。

Amazon CLI

描述飞轮迭代

以下 describe-flywheel-iteration 示例获取飞轮迭代的属性。

```
aws comprehend describe-flywheel-iteration \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-flywheel \
    --flywheel-iteration-id 20232222AEXAMPLE
```

输出：

```
{
    "FlywheelIterationProperties": {
        "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-entity",
        "FlywheelIterationId": "20232222AEXAMPLE",
        "CreationTime": "2023-06-16T21:10:26.385000+00:00",
        "EndTime": "2023-06-16T23:33:16.827000+00:00",
        "Status": "COMPLETED",
        "Message": "FULL_ITERATION: Flywheel iteration performed all functions successfully.",
        "EvaluatedModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/1",
        "EvaluatedModelMetrics": {
            "AverageF1Score": 0.7742663922375772,
            "AveragePrecision": 0.8287636394041166,
            "AverageRecall": 0.7427084833645399,
            "AverageAccuracy": 0.8795394154118689
        },
        "TrainedModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/Comprehend-Generated-v1-bb52d585",
        "TrainedModelMetrics": {
```

```
            "AverageF1Score": 0.9767700253081214,
            "AveragePrecision": 0.9767700253081214,
            "AverageRecall": 0.9767700253081214,
            "AverageAccuracy": 0.9858281665190434
        },
        "EvaluationManifestS3Prefix": "s3://amzn-s3-demo-destination-bucket/
flywheel-entity/schemaVersion=1/20230616T200543Z/evaluation/20230616T211026Z/"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFlywheelIteration。

describe-flywheel

以下代码示例演示了如何使用 describe-flywheel。

Amazon CLI

描述飞轮

以下 describe-flywheel 示例获取飞轮的属性。在此示例中，与飞轮关联的模型是一个自定义分类器模型，该模型经过训练，可以将文档分类为垃圾邮件、非垃圾邮件或“ham”。

```
aws comprehend describe-flywheel \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel
```

输出：

```
{
    "FlywheelProperties": {
        "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel",
        "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/example-model/version/1",
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
        "TaskConfig": {
            "LanguageCode": "en",
            "DocumentClassificationConfig": {
```

```
                "Mode": "MULTI_CLASS",
                "Labels": [
                    "ham",
                    "spam"
                ]
            }
        },
        "DataLakeS3Uri": "s3://amzn-s3-demo-bucket/example-flywheel/
schemaVersion=1/20230616T200543Z/",
        "DataSecurityConfig": {},
        "Status": "ACTIVE",
        "ModelType": "DOCUMENT_CLASSIFIER",
        "CreationTime": "2023-06-16T20:05:43.242000+00:00",
        "LastModifiedTime": "2023-06-16T20:21:43.567000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFlywheel。

describe-key-phrases-detection-job

以下代码示例演示了如何使用 describe-key-phrases-detection-job。

Amazon CLI

描述关键短语检测作业

以下 describe-key-phrases-detection-job 示例获取异步关键短语检测作业的属性。

```
aws comprehend describe-key-phrases-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "KeyPhrasesDetectionJobProperties": {
        "JobId": "69aa080c00fc68934a6a98f10EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-
job/69aa080c00fc68934a6a98f10EXAMPLE",
        "JobName": "example-key-phrases-detection-job",
        "JobStatus": "COMPLETED",
```

```
        "SubmitTime": 1686606439.177,
        "EndTime": 1686606806.157,
        "InputDataConfig": {
            "S3Uri": "s3://dereksbucket1001/EventsData/",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://dereksbucket1002/testfolder/111122223333-
KP-69aa080c00fc68934a6a98f10EXAMPLE/output/output.tar.gz"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testrole"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeKeyPhrasesDetectionJob。

describe-pii-entities-detection-job

以下代码示例演示了如何使用 describe-pii-entities-detection-job。

Amazon CLI

描述 PII 实体检测作业

以下 describe-pii-entities-detection-job 示例获取异步 PII 实体检测作业的属性。

```
aws comprehend describe-pii-entities-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "PiiEntitiesDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "example-pii-entities-job",
```

```
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-08T21:30:15.323000+00:00",
        "EndTime": "2023-06-08T21:40:23.509000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/thefolder/111122223333-
NER-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::12345678012:role/service-role/
AmazonComprehendServiceRole-example-role"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePiiEntitiesDetectionJob。

describe-resource-policy

以下代码示例演示了如何使用 describe-resource-policy。

Amazon CLI

描述附加到模型的资源策略

以下 describe-resource-policy 示例获取附加到模型的基于资源的策略属性。

```
aws comprehend describe-resource-policy \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1
```

输出：

```
{
    "ResourcePolicy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::444455556666:root\"},\"Action\":
\"comprehend:ImportModel\",\"Resource\":\"*\"}]}",
```

```
    "CreationTime": "2023-06-19T18:44:26.028000+00:00",
    "LastModifiedTime": "2023-06-19T18:53:02.002000+00:00",
    "PolicyRevisionId": "baa675d069d07afaa2aa3106ae280f61"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的在 Amazon 账户之间复制自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResourcePolicy。

describe-sentiment-detection-job

以下代码示例演示了如何使用 describe-sentiment-detection-job。

Amazon CLI

描述情绪检测作业

以下 describe-sentiment-detection-job 示例获取异步情绪检测作业的属性。

```
aws comprehend describe-sentiment-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "SentimentDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "movie_review_analysis",
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/MovieData",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "LanguageCode": "en",
```

```
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSentimentDetectionJob。

describe-targeted-sentiment-detection-job

以下代码示例演示了如何使用 describe-targeted-sentiment-detection-job。

Amazon CLI

描述目标情绪检测作业

以下 describe-targeted-sentiment-detection-job 示例获取异步目标情绪检测作业的属性。

```
aws comprehend describe-targeted-sentiment-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "TargetedSentimentDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "movie_review_analysis",
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/MovieData",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "LanguageCode": "en",
```

```
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTargetedSentimentDetectionJob。

describe-topics-detection-job

以下代码示例演示了如何使用 describe-topics-detection-job。

Amazon CLI

描述主题检测作业

以下 describe-topics-detection-job 示例获取异步主题检测作业的属性。

```
aws comprehend describe-topics-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "TopicsDetectionJobProperties": {
        "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
        "JobName": "example_topics_detection",
        "JobStatus": "IN_PROGRESS",
        "SubmitTime": "2023-06-09T18:44:43.414000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-destination-bucket/testfolder/111122223333-
TOPICS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
        },
        "NumberOfTopics": 10,
```

```
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-examplerole"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTopicsDetectionJob。

`detect-dominant-language`

以下代码示例演示了如何使用 `detect-dominant-language`。

Amazon CLI

检测输入文本的主要语言

以下 `detect-dominant-language` 分析输入文本并识别主要语言。预训练模型的置信度分数也是输出。

```
aws comprehend detect-dominant-language \
    --text "It is a beautiful day in Seattle."
```

输出：

```
{
    "Languages": [
        {
            "LanguageCode": "en",
            "Score": 0.9877256155014038
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的主要语言。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectDominantLanguage。

`detect-entities`

以下代码示例演示了如何使用 `detect-entities`。

Amazon CLI

检测输入文本中的命名实体

以下 detect-entities 示例分析输入文本并返回命名实体。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend detect-entities \
    --language-code en \
    --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC
 credit card \
    account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July
 31st. Based on your autopay settings, \
    we will withdraw your payment on the due date from your bank account number
 XXXXXX1111 with the routing number XXXXX0000. \
    Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to
 Alice at AnySpa@example.com."
```

输出：

```
{
    "Entities": [
        {
            "Score": 0.9994556307792664,
            "Type": "PERSON",
            "Text": "Zhang Wei",
            "BeginOffset": 6,
            "EndOffset": 15
        },
        {
            "Score": 0.9981022477149963,
            "Type": "PERSON",
            "Text": "John",
            "BeginOffset": 22,
            "EndOffset": 26
        },
        {
            "Score": 0.9986887574195862,
            "Type": "ORGANIZATION",
            "Text": "AnyCompany Financial Services, LLC",
            "BeginOffset": 33,
            "EndOffset": 67
        },
```

```
        {
            "Score": 0.9959119558334351,
            "Type": "OTHER",
            "Text": "1111-XXXX-1111-XXXX",
            "BeginOffset": 88,
            "EndOffset": 107
        },
        {
            "Score": 0.9708039164543152,
            "Type": "QUANTITY",
            "Text": ".53",
            "BeginOffset": 133,
            "EndOffset": 136
        },
        {
            "Score": 0.9987268447875977,
            "Type": "DATE",
            "Text": "July 31st",
            "BeginOffset": 152,
            "EndOffset": 161
        },
        {
            "Score": 0.9858865737915039,
            "Type": "OTHER",
            "Text": "XXXXXX1111",
            "BeginOffset": 271,
            "EndOffset": 281
        },
        {
            "Score": 0.9700471758842468,
            "Type": "OTHER",
            "Text": "XXXXX0000",
            "BeginOffset": 306,
            "EndOffset": 315
        },
        {
            "Score": 0.9591118693351746,
            "Type": "ORGANIZATION",
            "Text": "Sunshine Spa",
            "BeginOffset": 340,
            "EndOffset": 352
        },
        {
            "Score": 0.9797496795654297,
```

```
            "Type": "LOCATION",
            "Text": "123 Main St",
            "BeginOffset": 354,
            "EndOffset": 365
        },
        {
            "Score": 0.994929313659668,
            "Type": "PERSON",
            "Text": "Alice",
            "BeginOffset": 394,
            "EndOffset": 399
        },
        {
            "Score": 0.9949769377708435,
            "Type": "OTHER",
            "Text": "AnySpa@example.com",
            "BeginOffset": 403,
            "EndOffset": 418
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的实体。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectEntities。

detect-key-phrases

以下代码示例演示了如何使用 detect-key-phrases。

Amazon CLI

检测输入文本中的关键词

以下 detect-key-phrases 示例分析输入文本并识别关键名词短语。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend detect-key-phrases \
    --language-code en \
    --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC
 credit card \
        account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by
 July 31st. Based on your autopay settings, \
```

```
       we will withdraw your payment on the due date from your bank account number
XXXXXX1111 with the routing number XXXXX0000. \
       Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to
Alice at AnySpa@example.com."
```

输出：

```
{
    "KeyPhrases": [
        {
            "Score": 0.8996376395225525,
            "Text": "Zhang Wei",
            "BeginOffset": 6,
            "EndOffset": 15
        },
        {
            "Score": 0.9992469549179077,
            "Text": "John",
            "BeginOffset": 22,
            "EndOffset": 26
        },
        {
            "Score": 0.988385021686554,
            "Text": "Your AnyCompany Financial Services",
            "BeginOffset": 28,
            "EndOffset": 62
        },
        {
            "Score": 0.8740853071212769,
            "Text": "LLC credit card account 1111-XXXX-1111-XXXX",
            "BeginOffset": 64,
            "EndOffset": 107
        },
        {
            "Score": 0.9999437928199768,
            "Text": "a minimum payment",
            "BeginOffset": 112,
            "EndOffset": 129
        },
        {
            "Score": 0.9998900890350342,
            "Text": ".53",
            "BeginOffset": 133,
```

```
            "EndOffset": 136
        },
        {
            "Score": 0.9979453086853027,
            "Text": "July 31st",
            "BeginOffset": 152,
            "EndOffset": 161
        },
        {
            "Score": 0.9983011484146118,
            "Text": "your autopay settings",
            "BeginOffset": 172,
            "EndOffset": 193
        },
        {
            "Score": 0.9996572136878967,
            "Text": "your payment",
            "BeginOffset": 211,
            "EndOffset": 223
        },
        {
            "Score": 0.9995037317276001,
            "Text": "the due date",
            "BeginOffset": 227,
            "EndOffset": 239
        },
        {
            "Score": 0.9702621698379517,
            "Text": "your bank account number XXXXXX1111",
            "BeginOffset": 245,
            "EndOffset": 280
        },
        {
            "Score": 0.9179925918579102,
            "Text": "the routing number XXXXX0000.Customer feedback",
            "BeginOffset": 286,
            "EndOffset": 332
        },
        {
            "Score": 0.9978160858154297,
            "Text": "Sunshine Spa",
            "BeginOffset": 337,
            "EndOffset": 349
        },
```

```
        {
            "Score": 0.9706913232803345,
            "Text": "123 Main St",
            "BeginOffset": 351,
            "EndOffset": 362
        },
        {
            "Score": 0.9941995143890381,
            "Text": "comments",
            "BeginOffset": 379,
            "EndOffset": 387
        },
        {
            "Score": 0.9759287238121033,
            "Text": "Alice",
            "BeginOffset": 391,
            "EndOffset": 396
        },
        {
            "Score": 0.8376792669296265,
            "Text": "AnySpa@example.com",
            "BeginOffset": 400,
            "EndOffset": 415
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的关键词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectKeyPhrases。

detect-pii-entities

以下代码示例演示了如何使用 detect-pii-entities。

Amazon CLI

检测输入文本中的 PII 实体

以下 detect-pii-entities 示例分析输入文本，并识别包含个人身份信息（PII）的实体。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend detect-pii-entities \
```

```
    --language-code en \
    --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC
credit card \
        account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by
July 31st. Based on your autopay settings, \
        we will withdraw your payment on the due date from your bank account number
XXXXXX1111 with the routing number XXXXX0000. \
        Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to
Alice at AnySpa@example.com."
```

输出：

```
{
    "Entities": [
        {
            "Score": 0.9998322129249573,
            "Type": "NAME",
            "BeginOffset": 6,
            "EndOffset": 15
        },
        {
            "Score": 0.9998878240585327,
            "Type": "NAME",
            "BeginOffset": 22,
            "EndOffset": 26
        },
        {
            "Score": 0.9994089603424072,
            "Type": "CREDIT_DEBIT_NUMBER",
            "BeginOffset": 88,
            "EndOffset": 107
        },
        {
            "Score": 0.9999760985374451,
            "Type": "DATE_TIME",
            "BeginOffset": 152,
            "EndOffset": 161
        },
        {
            "Score": 0.9999449253082275,
            "Type": "BANK_ACCOUNT_NUMBER",
            "BeginOffset": 271,
            "EndOffset": 281
```

```
        },
        {
            "Score": 0.9999847412109375,
            "Type": "BANK_ROUTING",
            "BeginOffset": 306,
            "EndOffset": 315
        },
        {
            "Score": 0.999925434589386,
            "Type": "ADDRESS",
            "BeginOffset": 354,
            "EndOffset": 365
        },
        {
            "Score": 0.9989161491394043,
            "Type": "NAME",
            "BeginOffset": 394,
            "EndOffset": 399
        },
        {
            "Score": 0.9994171857833862,
            "Type": "EMAIL",
            "BeginOffset": 403,
            "EndOffset": 418
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的个人身份信息（PII）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectPiiEntities。

`detect-sentiment`

以下代码示例演示了如何使用 `detect-sentiment`。

Amazon CLI

检测输入文本的情绪

以下 `detect-sentiment` 示例分析输入文本，并返回占主导地位的情绪（`POSITIVE`、`NEUTRAL`、`MIXED` 或`NEGATIVE`）的推断。

```
aws comprehend detect-sentiment \
    --language-code en \
    --text "It is a beautiful day in Seattle"
```

输出：

```
{
    "Sentiment": "POSITIVE",
    "SentimentScore": {
        "Positive": 0.9976957440376282,
        "Negative": 9.653854067437351e-05,
        "Neutral": 0.002169104292988777,
        "Mixed": 3.857641786453314e-05
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的情绪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectSentiment。

detect-syntax

以下代码示例演示了如何使用 detect-syntax。

Amazon CLI

检测输入文本中的语音部分

以下 detect-syntax 示例分析输入文本的语法并返回语音的不同部分。预训练模型的置信度分数也是每个预测的输出。

```
aws comprehend detect-syntax \
    --language-code en \
    --text "It is a beautiful day in Seattle."
```

输出：

```
{
    "SyntaxTokens": [
        {
            "TokenId": 1,
            "Text": "It",
```

```
            "BeginOffset": 0,
            "EndOffset": 2,
            "PartOfSpeech": {
                "Tag": "PRON",
                "Score": 0.9999740719795227
            }
        },
        {
            "TokenId": 2,
            "Text": "is",
            "BeginOffset": 3,
            "EndOffset": 5,
            "PartOfSpeech": {
                "Tag": "VERB",
                "Score": 0.999901294708252
            }
        },
        {
            "TokenId": 3,
            "Text": "a",
            "BeginOffset": 6,
            "EndOffset": 7,
            "PartOfSpeech": {
                "Tag": "DET",
                "Score": 0.9999938607215881
            }
        },
        {
            "TokenId": 4,
            "Text": "beautiful",
            "BeginOffset": 8,
            "EndOffset": 17,
            "PartOfSpeech": {
                "Tag": "ADJ",
                "Score": 0.9987351894378662
            }
        },
        {
            "TokenId": 5,
            "Text": "day",
            "BeginOffset": 18,
            "EndOffset": 21,
            "PartOfSpeech": {
                "Tag": "NOUN",
```

```
                "Score": 0.9999796748161316
            }
        },
        {
            "TokenId": 6,
            "Text": "in",
            "BeginOffset": 22,
            "EndOffset": 24,
            "PartOfSpeech": {
                "Tag": "ADP",
                "Score": 0.9998047947883606
            }
        },
        {
            "TokenId": 7,
            "Text": "Seattle",
            "BeginOffset": 25,
            "EndOffset": 32,
            "PartOfSpeech": {
                "Tag": "PROPN",
                "Score": 0.9940530061721802
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的语法分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectSyntax。

detect-targeted-sentiment

以下代码示例演示了如何使用 detect-targeted-sentiment。

Amazon CLI

检测输入文本中命名实体的目标情绪

以下 detect-targeted-sentiment 示例分析输入文本，并返回命名实体以及与每个实体关联的目标情绪。也会输出每个预测的预训练模型的置信度分数。

```
aws comprehend detect-targeted-sentiment \
    --language-code en \
```

```
    --text "I do not enjoy January because it is too cold but August is the perfect
 temperature"
```

输出：

```
{
    "Entities": [
        {
            "DescriptiveMentionIndex": [
                0
            ],
            "Mentions": [
                {
                    "Score": 0.9999979734420776,
                    "GroupScore": 1.0,
                    "Text": "I",
                    "Type": "PERSON",
                    "MentionSentiment": {
                        "Sentiment": "NEUTRAL",
                        "SentimentScore": {
                            "Positive": 0.0,
                            "Negative": 0.0,
                            "Neutral": 1.0,
                            "Mixed": 0.0
                        }
                    },
                    "BeginOffset": 0,
                    "EndOffset": 1
                }
            ]
        },
        {
            "DescriptiveMentionIndex": [
                0
            ],
            "Mentions": [
                {
                    "Score": 0.9638869762420654,
                    "GroupScore": 1.0,
                    "Text": "January",
                    "Type": "DATE",
                    "MentionSentiment": {
                        "Sentiment": "NEGATIVE",
```

```
                    "SentimentScore": {
                        "Positive": 0.0031610000878572464,
                        "Negative": 0.9967250227928162,
                        "Neutral": 0.00011100000119768083,
                        "Mixed": 1.9999999949504854e-06
                    }
                },
                "BeginOffset": 15,
                "EndOffset": 22
            }
        ]
    },
    {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            {
                "Score": 0.9664419889450073,
                "GroupScore": 1.0,
                "Text": "August",
                "Type": "DATE",
                "MentionSentiment": {
                    "Sentiment": "POSITIVE",
                    "SentimentScore": {
                        "Positive": 0.9999549984931946,
                        "Negative": 3.999999989900971e-06,
                        "Neutral": 4.099999932805076e-05,
                        "Mixed": 0.0
                    }
                },
                "BeginOffset": 50,
                "EndOffset": 56
            }
        ]
    },
    {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
                "Score": 0.9803199768066406,
```

```
                    "GroupScore": 1.0,
                    "Text": "temperature",
                    "Type": "ATTRIBUTE",
                    "MentionSentiment": {
                        "Sentiment": "POSITIVE",
                        "SentimentScore": {
                            "Positive": 1.0,
                            "Negative": 0.0,
                            "Neutral": 0.0,
                            "Mixed": 0.0
                        }
                    },
                    "BeginOffset": 77,
                    "EndOffset": 88
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的目标情绪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectTargetedSentiment。

import-model

以下代码示例演示了如何使用 import-model。

Amazon CLI

导入模型

以下 import-model 示例从不同的 Amazon 账户导入模型。账户 444455556666 中的文档分类器模型具有基于资源的策略，允许账户 111122223333 导入模型。

```
aws comprehend import-model \
    --source-model-arn arn:aws:comprehend:us-west-2:444455556666:document-classifier/example-classifier
```

输出：

```
{
```

```
    "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的在 Amazon 账户之间复制自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportModel。

list-datasets

以下代码示例演示了如何使用 list-datasets。

Amazon CLI

列出所有飞轮数据集

以下 list-datasets 示例列出与飞轮关联的所有数据集。

```
aws comprehend list-datasets \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-
entity
```

输出：

```
{
    "DatasetPropertiesList": [
        {
            "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
flywheel-entity/dataset/example-dataset-1",
            "DatasetName": "example-dataset-1",
            "DatasetType": "TRAIN",
            "DatasetS3Uri": "s3://amzn-s3-demo-bucket/flywheel-entity/
schemaVersion=1/20230616T200543Z/datasets/example-dataset-1/20230616T203710Z/",
            "Status": "CREATING",
            "CreationTime": "2023-06-16T20:37:10.400000+00:00"
        },
        {
            "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
flywheel-entity/dataset/example-dataset-2",
            "DatasetName": "example-dataset-2",
            "DatasetType": "TRAIN",
```

```
            "DatasetS3Uri": "s3://amzn-s3-demo-bucket/flywheel-entity/
schemaVersion=1/20230616T200543Z/datasets/example-dataset-2/20230616T200607Z/",
            "Description": "TRAIN Dataset created by Flywheel creation.",
            "Status": "COMPLETED",
            "NumberOfDocuments": 5572,
            "CreationTime": "2023-06-16T20:06:07.722000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatasets。

list-document-classification-jobs

以下代码示例演示了如何使用 list-document-classification-jobs。

Amazon CLI

列出所有文档分类作业

以下 list-document-classification-jobs 示例列出所有文档分类作业。

```
aws comprehend list-document-classification-jobs
```

输出：

```
{
    "DocumentClassificationJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:1234567890101:document-
classification-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "exampleclassificationjob",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-14T17:09:51.788000+00:00",
            "EndTime": "2023-06-14T17:15:58.582000+00:00",
            "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:1234567890101:document-classifier/mymodel/version/12",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/jobdata/",
                "InputFormat": "ONE_DOC_PER_LINE"
```

```
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/1234567890101-CLN-e758dd56b824aa717ceab551f11749fb/output/output.tar.gz"
            },
            "DataAccessRoleArn": "arn:aws:iam::1234567890101:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
            "JobArn": "arn:aws:comprehend:us-west-2:1234567890101:document-
classification-job/123456abcdeb0e11022f22a1EXAMPLE2",
            "JobName": "exampleclassificationjob2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-14T17:22:39.829000+00:00",
            "EndTime": "2023-06-14T17:28:46.107000+00:00",
            "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:1234567890101:document-classifier/mymodel/version/12",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/jobdata/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/1234567890101-CLN-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
            },
            "DataAccessRoleArn": "arn:aws:iam::1234567890101:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义分类。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDocumentClassificationJobs。

`list-document-classifier-summaries`

以下代码示例演示了如何使用 `list-document-classifier-summaries`。

Amazon CLI

列出所有已创建文档分类器的摘要

以下 list-document-classifier-summaries 示例列出所有已创建文档分类器的摘要。

```
aws comprehend list-document-classifier-summaries
```

输出：

```
{
    "DocumentClassifierSummariesList": [
        {
            "DocumentClassifierName": "example-classifier-1",
            "NumberOfVersions": 1,
            "LatestVersionCreatedAt": "2023-06-13T22:07:59.825000+00:00",
            "LatestVersionName": "1",
            "LatestVersionStatus": "TRAINED"
        },
        {
            "DocumentClassifierName": "example-classifier-2",
            "NumberOfVersions": 2,
            "LatestVersionCreatedAt": "2023-06-13T21:54:59.589000+00:00",
            "LatestVersionName": "2",
            "LatestVersionStatus": "TRAINED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的创建和管理自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDocumentClassifierSummaries。

list-document-classifiers

以下代码示例演示了如何使用 list-document-classifiers。

Amazon CLI

列出所有文档分类器

以下 list-document-classifiers 示例列出所有经过训练和正在训练的文档分类器模型。

```
aws comprehend list-document-classifiers
```

输出：

```
{
    "DocumentClassifierPropertiesList": [
        {
            "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/exampleclassifier1",
            "LanguageCode": "en",
            "Status": "TRAINED",
            "SubmitTime": "2023-06-13T19:04:15.735000+00:00",
            "EndTime": "2023-06-13T19:42:31.752000+00:00",
            "TrainingStartTime": "2023-06-13T19:08:20.114000+00:00",
            "TrainingEndTime": "2023-06-13T19:41:35.080000+00:00",
            "InputDataConfig": {
                "DataFormat": "COMPREHEND_CSV",
                "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata"
            },
            "OutputDataConfig": {},
            "ClassifierMetadata": {
                "NumberOfLabels": 3,
                "NumberOfTrainedDocuments": 5016,
                "NumberOfTestDocuments": 557,
                "EvaluationMetrics": {
                    "Accuracy": 0.9856,
                    "Precision": 0.9919,
                    "Recall": 0.9459,
                    "F1Score": 0.9673,
                    "MicroPrecision": 0.9856,
                    "MicroRecall": 0.9856,
                    "MicroF1Score": 0.9856,
                    "HammingLoss": 0.0144
                }
            },
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testorle",
            "Mode": "MULTI_CLASS"
        },
        {
            "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/exampleclassifier2",
            "LanguageCode": "en",
            "Status": "TRAINING",
            "SubmitTime": "2023-06-13T21:20:28.690000+00:00",
            "InputDataConfig": {
                "DataFormat": "COMPREHEND_CSV",
```

```
                "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata"
            },
            "OutputDataConfig": {},
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testorle",
            "Mode": "MULTI_CLASS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的创建和管理自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDocumentClassifiers。

list-dominant-language-detection-jobs

以下代码示例演示了如何使用 list-dominant-language-detection-jobs。

Amazon CLI

列出所有主要语言检测作业

以下 list-dominant-language-detection-jobs 示例列出所有正在进行和已完成的异步主要语言检测作业。

```
aws comprehend list-dominant-language-detection-jobs
```

输出：

```
{
    "DominantLanguageDetectionJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "languageanalysis1",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T18:10:38.037000+00:00",
            "EndTime": "2023-06-09T18:18:45.498000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
```

```
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/
output.tar.gz"
            },
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "languageanalysis2",
            "JobStatus": "STOPPED",
            "SubmitTime": "2023-06-09T18:16:33.690000+00:00",
            "EndTime": "2023-06-09T18:24:40.608000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/
output.tar.gz"
            },
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDominantLanguageDetectionJobs。

`list-endpoints`

以下代码示例演示了如何使用 `list-endpoints`。

Amazon CLI

列出所有端点

以下 `list-endpoints` 示例列出所有活动的指定模型的端点。

```
aws comprehend list-endpoints
```

输出：

```
{
    "EndpointPropertiesList": [
        {
            "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/ExampleClassifierEndpoint",
            "Status": "IN_SERVICE",
            "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
            "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
            "DesiredInferenceUnits": 1,
            "CurrentInferenceUnits": 1,
            "CreationTime": "2023-06-13T20:32:54.526000+00:00",
            "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"
        },
        {
            "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/ExampleClassifierEndpoint2",
            "Status": "IN_SERVICE",
            "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2",
            "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2",
            "DesiredInferenceUnits": 1,
            "CurrentInferenceUnits": 1,
            "CreationTime": "2023-06-13T20:32:54.526000+00:00",
            "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEndpoints。

list-entities-detection-jobs

以下代码示例演示了如何使用 list-entities-detection-jobs。

Amazon CLI

列出所有实体检测作业

以下 list-entities-detection-jobs 示例列出所有异步实体检测作业。

```
aws comprehend list-entities-detection-jobs
```

输出：

```
{
    "EntitiesDetectionJobPropertiesList": [
        {
            "JobId": "468af39c28ab45b83eb0c4ab9EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/468af39c28ab45b83eb0c4ab9EXAMPLE",
            "JobName": "example-entities-detection",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-08T20:57:46.476000+00:00",
            "EndTime": "2023-06-08T21:05:53.718000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-NER-468af39c28ab45b83eb0c4ab9EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "809691caeaab0e71406f80a28EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/809691caeaab0e71406f80a28EXAMPLE",
```

```
            "JobName": "example-entities-detection-2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-08T21:30:15.323000+00:00",
            "EndTime": "2023-06-08T21:40:23.509000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-NER-809691caeaab0e71406f80a28EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "e00597c36b448b91d70dea165EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/e00597c36b448b91d70dea165EXAMPLE",
            "JobName": "example-entities-detection-3",
            "JobStatus": "STOPPED",
            "SubmitTime": "2023-06-08T22:19:28.528000+00:00",
            "EndTime": "2023-06-08T22:27:33.991000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-NER-e00597c36b448b91d70dea165EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的实体。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntitiesDetectionJobs。

list-entity-recognizer-summaries

以下代码示例演示了如何使用 list-entity-recognizer-summaries。

Amazon CLI

查看所有已创建实体识别器的摘要列表

以下 list-entity-recognizer-summaries 示例列出所有实体识别器摘要。

```
aws comprehend list-entity-recognizer-summaries
```

输出：

```
{
    "EntityRecognizerSummariesList": [
        {
            "RecognizerName": "entity-recognizer-3",
            "NumberOfVersions": 2,
            "LatestVersionCreatedAt": "2023-06-15T23:15:07.621000+00:00",
            "LatestVersionName": "2",
            "LatestVersionStatus": "STOP_REQUESTED"
        },
        {
            "RecognizerName": "entity-recognizer-2",
            "NumberOfVersions": 1,
            "LatestVersionCreatedAt": "2023-06-14T22:55:27.805000+00:00",
            "LatestVersionName": "2"
            "LatestVersionStatus": "TRAINED"
        },
        {
            "RecognizerName": "entity-recognizer-1",
            "NumberOfVersions": 1,
            "LatestVersionCreatedAt": "2023-06-14T20:44:59.631000+00:00",
            "LatestVersionName": "1",
            "LatestVersionStatus": "TRAINED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义实体识别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntityRecognizerSummaries。

list-entity-recognizers

以下代码示例演示了如何使用 list-entity-recognizers。

Amazon CLI

列出所有自定义实体识别器

以下 list-entity-recognizers 示例列出所有已创建自定义实体识别器。

```
aws comprehend list-entity-recognizers
```

输出：

```
{
    "EntityRecognizerPropertiesList": [
        {
            "EntityRecognizerArn": "arn:aws:comprehend:us-
west-2:111122223333:entity-recognizer/EntityRecognizer/version/1",
            "LanguageCode": "en",
            "Status": "TRAINED",
            "SubmitTime": "2023-06-14T20:44:59.631000+00:00",
            "EndTime": "2023-06-14T20:59:19.532000+00:00",
            "TrainingStartTime": "2023-06-14T20:48:52.811000+00:00",
            "TrainingEndTime": "2023-06-14T20:58:11.473000+00:00",
            "InputDataConfig": {
                "DataFormat": "COMPREHEND_CSV",
                "EntityTypes": [
                    {
                        "Type": "BUSINESS"
                    }
                ],
                "Documents": {
                    "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/dataset/",
                    "InputFormat": "ONE_DOC_PER_LINE"
                },
                "EntityList": {
                    "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/entity.csv"
                }
            },
            "RecognizerMetadata": {
                "NumberOfTrainedDocuments": 1814,
                "NumberOfTestDocuments": 486,
                "EvaluationMetrics": {
```

```
                    "Precision": 100.0,
                    "Recall": 100.0,
                    "F1Score": 100.0
                },
                "EntityTypes": [
                    {
                        "Type": "BUSINESS",
                        "EvaluationMetrics": {
                            "Precision": 100.0,
                            "Recall": 100.0,
                            "F1Score": 100.0
                        },
                        "NumberOfTrainMentions": 1520
                    }
                ]
            },
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole",
            "VersionName": "1"
        },
        {
            "EntityRecognizerArn": "arn:aws:comprehend:us-
west-2:111122223333:entity-recognizer/entityrecognizer3",
            "LanguageCode": "en",
            "Status": "TRAINED",
            "SubmitTime": "2023-06-14T22:57:51.056000+00:00",
            "EndTime": "2023-06-14T23:14:13.894000+00:00",
            "TrainingStartTime": "2023-06-14T23:01:33.984000+00:00",
            "TrainingEndTime": "2023-06-14T23:13:02.984000+00:00",
            "InputDataConfig": {
                "DataFormat": "COMPREHEND_CSV",
                "EntityTypes": [
                    {
                        "Type": "DEVICE"
                    }
                ],
                "Documents": {
                    "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/raw_txt.csv",
                    "InputFormat": "ONE_DOC_PER_LINE"
                },
                "EntityList": {
                    "S3Uri": "s3://amzn-s3-demo-bucket/trainingdata/entity_list.csv"
                }
            },
```

```
            "RecognizerMetadata": {
                "NumberOfTrainedDocuments": 4616,
                "NumberOfTestDocuments": 3489,
                "EvaluationMetrics": {
                    "Precision": 98.54227405247813,
                    "Recall": 100.0,
                    "F1Score": 99.26578560939794
                },
                "EntityTypes": [
                    {
                        "Type": "DEVICE",
                        "EvaluationMetrics": {
                            "Precision": 98.54227405247813,
                            "Recall": 100.0,
                            "F1Score": 99.26578560939794
                        },
                        "NumberOfTrainMentions": 2764
                    }
                ]
            },
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义实体识别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntityRecognizers。

list-events-detection-jobs

以下代码示例演示了如何使用 list-events-detection-jobs。

Amazon CLI

列出所有事件检测作业

以下 list-events-detection-jobs 示例列出所有异步事件检测作业。

```
aws comprehend list-events-detection-jobs
```

输出：

```
{
    "EventsDetectionJobPropertiesList": [
        {
            "JobId": "aa9593f9203e84f3ef032ce18EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:1111222233333:events-detection-
job/aa9593f9203e84f3ef032ce18EXAMPLE",
            "JobName": "events_job_1",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-12T19:14:57.751000+00:00",
            "EndTime": "2023-06-12T19:21:04.962000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-source-bucket/EventsData/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/1111222233333-EVENTS-aa9593f9203e84f3ef032ce18EXAMPLE/output/"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::1111222233333:role/service-role/
AmazonComprehendServiceRole-example-role",
            "TargetEventTypes": [
                "BANKRUPTCY",
                "EMPLOYMENT",
                "CORPORATE_ACQUISITION",
                "CORPORATE_MERGER",
                "INVESTMENT_GENERAL"
            ]
        },
        {
            "JobId": "4a990a2f7e82adfca6e171135EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:1111222233333:events-detection-
job/4a990a2f7e82adfca6e171135EXAMPLE",
            "JobName": "events_job_2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-12T19:55:43.702000+00:00",
            "EndTime": "2023-06-12T20:03:49.893000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-source-bucket/EventsData/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
```

```
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/1111222233333-EVENTS-4a990a2f7e82adfca6e171135EXAMPLE/output/"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::1111222233333:role/service-role/
AmazonComprehendServiceRole-example-role",
            "TargetEventTypes": [
                "BANKRUPTCY",
                "EMPLOYMENT",
                "CORPORATE_ACQUISITION",
                "CORPORATE_MERGER",
                "INVESTMENT_GENERAL"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEventsDetectionJobs。

list-flywheel-iteration-history

以下代码示例演示了如何使用 list-flywheel-iteration-history。

Amazon CLI

列出所有飞轮迭代历史记录

以下 list-flywheel-iteration-history 示例列出飞轮的所有迭代。

```
aws comprehend list-flywheel-iteration-history
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel
```

输出：

```
{
    "FlywheelIterationPropertiesList": [
        {
            "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel",
```

```
            "FlywheelIterationId": "20230619TEXAMPLE",
            "CreationTime": "2023-06-19T04:00:32.594000+00:00",
            "EndTime": "2023-06-19T04:00:49.248000+00:00",
            "Status": "COMPLETED",
            "Message": "FULL_ITERATION: Flywheel iteration performed all functions
 successfully.",
            "EvaluatedModelArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/example-classifier/version/1",
            "EvaluatedModelMetrics": {
                "AverageF1Score": 0.7742663922375772,
                "AverageF1Score": 0.9876464664646313,
                "AveragePrecision": 0.9800000253081214,
                "AverageRecall": 0.9445600253081214,
                "AverageAccuracy": 0.9997281665190434
            },
            "EvaluationManifestS3Prefix": "s3://amzn-s3-demo-bucket/example-
flywheel/schemaVersion=1/20230619TEXAMPLE/evaluation/20230619TEXAMPLE/"
        },
        {
            "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-2",
            "FlywheelIterationId": "20230616TEXAMPLE",
            "CreationTime": "2023-06-16T21:10:26.385000+00:00",
            "EndTime": "2023-06-16T23:33:16.827000+00:00",
            "Status": "COMPLETED",
            "Message": "FULL_ITERATION: Flywheel iteration performed all functions
 successfully.",
            "EvaluatedModelArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/spamvshamclassify/version/1",
            "EvaluatedModelMetrics": {
                "AverageF1Score": 0.7742663922375772,
                "AverageF1Score": 0.9767700253081214,
                "AveragePrecision": 0.9767700253081214,
                "AverageRecall": 0.9767700253081214,
                "AverageAccuracy": 0.9858281665190434
            },
            "EvaluationManifestS3Prefix": "s3://amzn-s3-demo-bucket/example-
flywheel-2/schemaVersion=1/20230616TEXAMPLE/evaluation/20230616TEXAMPLE/"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFlywheelIterationHistory。

list-flywheels

以下代码示例演示了如何使用 list-flywheels。

Amazon CLI

列出所有飞轮

以下 list-flywheels 示例列出所有已创建的飞轮。

```
aws comprehend list-flywheels
```

输出：

```
{
    "FlywheelSummaryList": [
        {
            "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-1",
            "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier/version/1",
            "DataLakeS3Uri": "s3://amzn-s3-demo-bucket/example-flywheel-1/
schemaVersion=1/20230616T200543Z/",
            "Status": "ACTIVE",
            "ModelType": "DOCUMENT_CLASSIFIER",
            "CreationTime": "2023-06-16T20:05:43.242000+00:00",
            "LastModifiedTime": "2023-06-19T04:00:43.027000+00:00",
            "LatestFlywheelIteration": "20230619T040032Z"
        },
        {
            "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-2",
            "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2/version/1",
            "DataLakeS3Uri": "s3://amzn-s3-demo-bucket/example-flywheel-2/
schemaVersion=1/20220616T200543Z/",
            "Status": "ACTIVE",
            "ModelType": "DOCUMENT_CLASSIFIER",
            "CreationTime": "2022-06-16T20:05:43.242000+00:00",
            "LastModifiedTime": "2022-06-19T04:00:43.027000+00:00",
```

```
            "LatestFlywheelIteration": "20220619T040032Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFlywheels。

`list-key-phrases-detection-jobs`

以下代码示例演示了如何使用 `list-key-phrases-detection-jobs`。

Amazon CLI

列出所有关键短语检测作业

以下 `list-key-phrases-detection-jobs` 示例列出所有正在进行和已完成的异步关键短语检测作业。

```
aws comprehend list-key-phrases-detection-jobs
```

输出：

```
{
    "KeyPhrasesDetectionJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "keyphrasesanalysis1",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-08T22:31:43.767000+00:00",
            "EndTime": "2023-06-08T22:39:52.565000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-source-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-KP-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
```

```
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a33EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a33EXAMPLE",
            "JobName": "keyphrasesanalysis2",
            "JobStatus": "STOPPED",
            "SubmitTime": "2023-06-08T22:57:52.154000+00:00",
            "EndTime": "2023-06-08T23:05:48.385000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-KP-123456abcdeb0e11022f22a33EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a44EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a44EXAMPLE",
            "JobName": "keyphrasesanalysis3",
            "JobStatus": "FAILED",
            "Message": "NO_READ_ACCESS_TO_INPUT: The provided data access role does
 not have proper access to the input data.",
            "SubmitTime": "2023-06-09T16:47:04.029000+00:00",
            "EndTime": "2023-06-09T16:47:18.413000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-KP-123456abcdeb0e11022f22a44EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListKeyPhrasesDetectionJobs。

`list-pii-entities-detection-jobs`

以下代码示例演示了如何使用 `list-pii-entities-detection-jobs`。

Amazon CLI

列出所有 PII 实体检测作业

以下 `list-pii-entities-detection-jobs` 示例列出所有正在进行和已完成的异步 PII 检测作业。

```
aws comprehend list-pii-entities-detection-jobs
```

输出：

```
{
    "PiiEntitiesDetectionJobPropertiesList": [
        {
            "JobId": "6f9db0c42d0c810e814670ee4EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-
detection-job/6f9db0c42d0c810e814670ee4EXAMPLE",
            "JobName": "example-pii-detection-job",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T21:02:46.241000+00:00",
            "EndTime": "2023-06-09T21:12:52.602000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-source-bucket/111122223333-
PII-6f9db0c42d0c810e814670ee4EXAMPLE/output/"
            },
            "LanguageCode": "en",
```

```
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
            "Mode": "ONLY_OFFSETS"
        },
        {
            "JobId": "d927562638cfa739331a99b3cEXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-
detection-job/d927562638cfa739331a99b3cEXAMPLE",
            "JobName": "example-pii-detection-job-2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T21:20:58.211000+00:00",
            "EndTime": "2023-06-09T21:31:06.027000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/AsyncBatchJobs/",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-PII-d927562638cfa739331a99b3cEXAMPLE/output/"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
            "Mode": "ONLY_OFFSETS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPiiEntitiesDetectionJobs。

`list-sentiment-detection-jobs`

以下代码示例演示了如何使用 `list-sentiment-detection-jobs`。

Amazon CLI

列出所有情绪检测作业

以下 `list-sentiment-detection-jobs` 示例列出所有正在进行和已完成的异步情绪检测作业。

```
aws comprehend list-sentiment-detection-jobs
```

输出：

```
{
    "SentimentDetectionJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "example-sentiment-detection-job",
            "JobStatus": "IN_PROGRESS",
            "SubmitTime": "2023-06-09T22:42:20.545000+00:00",
            "EndTime": "2023-06-09T22:52:27.416000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/MovieData",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-
detection-job/123456abcdeb0e11022f22a1EXAMPLE2",
            "JobName": "example-sentiment-detection-job-2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
            "EndTime": "2023-06-09T23:26:00.168000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/MovieData2",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-TS-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
            },
            "LanguageCode": "en",
```

```
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSentimentDetectionJobs。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出 Amazon Comprehend 资源的标签。

```
aws comprehend list-tags-for-resource \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1
```

输出：

```
{
    "ResourceArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1",
    "Tags": [
        {
            "Key": "Department",
            "Value": "Finance"
        },
        {
            "Key": "location",
            "Value": "Seattle"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-targeted-sentiment-detection-jobs

以下代码示例演示了如何使用 list-targeted-sentiment-detection-jobs。

Amazon CLI

列出所有目标情绪检测作业

以下 list-targeted-sentiment-detection-jobs 示例列出所有正在进行和已完成的异步目标情绪检测作业。

```
aws comprehend list-targeted-sentiment-detection-jobs
```

输出：

```
{
    "TargetedSentimentDetectionJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName": "example-targeted-sentiment-detection-job",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T22:42:20.545000+00:00",
            "EndTime": "2023-06-09T22:52:27.416000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/MovieData",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-IOrole"
        },
        {
            "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a1EXAMPLE2",
```

```
            "JobName": "example-targeted-sentiment-detection-job-2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
            "EndTime": "2023-06-09T23:26:00.168000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/MovieData2",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
testfolder/111122223333-TS-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetedSentimentDetectionJobs。

list-topics-detection-jobs

以下代码示例演示了如何使用 list-topics-detection-jobs。

Amazon CLI

列出所有主题检测作业

以下 list-topics-detection-jobs 示例列出所有正在进行和已完成的异步主题检测作业。

```
aws comprehend list-topics-detection-jobs
```

输出：

```
{
    "TopicsDetectionJobPropertiesList": [
        {
            "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
```

```
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
            "JobName" "topic-analysis-1"
            "JobStatus": "IN_PROGRESS",
            "SubmitTime": "2023-06-09T18:40:35.384000+00:00",
            "EndTime": "2023-06-09T18:46:41.936000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
            },
            "NumberOfTopics": 10,
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a1EXAMPLE2",
            "JobName": "topic-analysis-2",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2023-06-09T18:44:43.414000+00:00",
            "EndTime": "2023-06-09T18:50:50.872000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
            },
            "NumberOfTopics": 10,
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        },
        {
            "JobId": "123456abcdeb0e11022f22a1EXAMPLE3",
            "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a1EXAMPLE3",
            "JobName": "topic-analysis-2",
            "JobStatus": "IN_PROGRESS",
```

```
            "SubmitTime": "2023-06-09T18:50:56.737000+00:00",
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket",
                "InputFormat": "ONE_DOC_PER_LINE"
            },
            "OutputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-destination-bucket/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a1EXAMPLE3/output/output.tar.gz"
            },
            "NumberOfTopics": 10,
            "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTopicsDetectionJobs。

put-resource-policy

以下代码示例演示了如何使用 put-resource-policy。

Amazon CLI

附加基于资源的策略

以下 put-resource-policy 示例将基于资源的策略附加到模型，以便其他 Amazon 账户导入。该策略附加到账户 111122223333 中的模型，并允许账户 444455556666 导入模型。

```
aws comprehend put-resource-policy \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1 \
    --resource-policy '{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Action":"comprehend:ImportModel","Resource":"*","Principal":
{"AWS":["arn:aws:iam::444455556666:root"]}}]}'
```

输出：

```
{
    "PolicyRevisionId": "aaa111d069d07afaa2aa3106aEXAMPLE"
```

```
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的在 Amazon 账户之间复制自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutResourcePolicy。

start-document-classification-job

以下代码示例演示了如何使用 start-document-classification-job。

Amazon CLI

列出文档分类作业

以下 start-document-classification-job 示例以自定义模型启动文档分类作业，该作业对 --input-data-config 标签所指定地址处的所有文件都使用自定义模型。在此示例中，输入 S3 存储桶包含 SampleSMStext1.txt、SampleSMStext2.txt、和 SampleSMStext3.txt。该模型之前曾接受过关于垃圾邮件和非垃圾邮件，或“ham”、短信的文档分类训练。作业完成后，output.tar.gz 将放置在 --output-data-config 标签指定的位置。output.tar.gz 包含 predictions.jsonl，其中列出了每个文档的分类。Json 输出在每个文件的一行上打印，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-document-classification-job \
    --job-name exampleclassificationjob \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket-INPUT/jobdata/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-
classifier/mymodel/version/12
```

的内容SampleSMStext1.txt：

```
"CONGRATULATIONS! TXT 2155550100 to win $5000"
```

的内容SampleSMStext2.txt：

```
"Hi, when do you want me to pick you up from practice?"
```

的内容SampleSMStext3.txt：

```
"Plz send bank account # to 2155550100 to claim prize!!"
```

输出：

```
{
    "JobId": "e758dd56b824aa717ceab551fEXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:document-classification-
job/e758dd56b824aa717ceab551fEXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

的内容predictions.jsonl：

```
{"File": "SampleSMSText1.txt", "Line": "0", "Classes": [{"Name": "spam", "Score":
 0.9999}, {"Name": "ham", "Score": 0.0001}]}
{"File": "SampleSMStext2.txt", "Line": "0", "Classes": [{"Name": "ham", "Score":
 0.9994}, {"Name": "spam", "Score": 0.0006}]}
{"File": "SampleSMSText3.txt", "Line": "0", "Classes": [{"Name": "spam", "Score":
 0.9999}, {"Name": "ham", "Score": 0.0001}]}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义分类。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDocumentClassificationJob。

start-dominant-language-detection-job

以下代码示例演示了如何使用 start-dominant-language-detection-job。

Amazon CLI

启动异步语言检测作业

以下 start-dominant-language-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步语言检测作业。此示例中的 S3 存储桶包含 Sampletext1.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 output.txt，其中包含每个文本文件的主要语言以及每个预测的预训练模型的置信度分数。

```
aws comprehend start-dominant-language-detection-job \
    --job-name example_language_analysis_job \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
```

```
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --language-code en
```

Sampletext1.txt 的内容：

```
"Physics is the natural science that involves the study of matter and its motion and
 behavior through space and time, along with related concepts such as energy and
 force."
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

的内容output.txt：

```
{"File": "Sampletext1.txt", "Languages": [{"LanguageCode": "en", "Score":
 0.9913753867149353}], "Line": 0}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDominantLanguageDetectionJob。

start-entities-detection-job

以下代码示例演示了如何使用 start-entities-detection-job。

Amazon CLI

示例 1：使用预训练模型启动标准实体检测作业

以下 start-entities-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步实体检测作业。此示例中的 S3 存储桶包含

Sampletext1.txt、Sampletext2.txt 和 Sampletext3.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 output.txt，其中列出了在每个文本文件中检测到的所有命名实体，以及预训练模型对每个预测的置信度分数。每个输入文件的 Json 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-entities-detection-job \
    --job-name entitiestest \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --language-code en
```

的内容Sampletext1.txt：

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card
 account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July
 31st."
```

的内容Sampletext2.txt：

```
"Dear Max, based on your autopay settings for your account example1.org account, we
 will withdraw your payment on the due date from your bank account number XXXXXX1111
 with the routing number XXXXX0000. "
```

的内容Sampletext3.txt：

```
"Jane, please submit any customer feedback from this weekend to AnySpa, 123 Main St,
 Anywhere and send comments to Alice at AnySpa@example.com."
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

output.txt 的内容，为便于阅读，采用了行间缩进：


```
{
"Entities": [
    {
    "BeginOffset": 6,
    "EndOffset": 15,
    "Score": 0.9994006636420306,
    "Text": "Zhang Wei",
    "Type": "PERSON"
    },
    {
    "BeginOffset": 22,
    "EndOffset": 26,
    "Score": 0.9976647915128143,
    "Text": "John",
    "Type": "PERSON"
    },
    {
    "BeginOffset": 33,
    "EndOffset": 67,
    "Score": 0.9984608700836206,
    "Text": "AnyCompany Financial Services, LLC",
    "Type": "ORGANIZATION"
    },
    {
    "BeginOffset": 88,
    "EndOffset": 107,
    "Score": 0.9868521019555556,
    "Text": "1111-XXXX-1111-XXXX",
    "Type": "OTHER"
    },
    {
    "BeginOffset": 133,
    "EndOffset": 139,
    "Score": 0.998242565709204,
    "Text": "$24.53",
    "Type": "QUANTITY"
    },
    {
    "BeginOffset": 155,
    "EndOffset": 164,
    "Score": 0.9993039263159287,
    "Text": "July 31st",
```

```
    "Type": "DATE"
    }
],
"File": "SampleText1.txt",
"Line": 0
}
{
"Entities": [
    {
    "BeginOffset": 5,
    "EndOffset": 8,
    "Score": 0.9866232147545232,
    "Text": "Max",
    "Type": "PERSON"
    },
    {
    "BeginOffset": 156,
    "EndOffset": 166,
    "Score": 0.9797723450933329,
    "Text": "XXXXXX1111",
    "Type": "OTHER"
    },
    {
    "BeginOffset": 191,
    "EndOffset": 200,
    "Score": 0.9247838572396843,
    "Text": "XXXXX0000",
    "Type": "OTHER"
    }
],
"File": "SampleText2.txt",
"Line": 0
}
{
 "Entities": [
    {
    "Score": 0.9990532994270325,
    "Type": "PERSON",
    "Text": "Jane",
    "BeginOffset": 0,
    "EndOffset": 4
    },
    {
    "Score": 0.9519651532173157,
```

```
    "Type": "DATE",
    "Text": "this weekend",
    "BeginOffset": 47,
    "EndOffset": 59
    },
    {
    "Score": 0.5566426515579224,
    "Type": "ORGANIZATION",
    "Text": "AnySpa",
    "BeginOffset": 63,
    "EndOffset": 69
    },
    {
    "Score": 0.8059805631637573,
    "Type": "LOCATION",
    "Text": "123 Main St, Anywhere",
    "BeginOffset": 71,
    "EndOffset": 92
    },
    {
    "Score": 0.998830258846283,
    "Type": "PERSON",
    "Text": "Alice",
    "BeginOffset": 114,
    "EndOffset": 119
    },
    {
    "Score": 0.997818112373352,
    "Type": "OTHER",
    "Text": "AnySpa@example.com",
    "BeginOffset": 123,
    "EndOffset": 138
    }
    ],
    "File": "SampleText3.txt",
    "Line": 0
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

示例 2：启动自定义实体检测作业

以下 start-entities-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步自定义实体检测作业。在此示例中，S3 存储桶包含 SampleFeedback1.txt、SampleFeedback2.txt 和 SampleFeedback3.txt。实体识别器模型经过客户支持反馈的训练，可以识别设备名称。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 output.txt，其中列出了在每个文本文件中检测到的所有命名实体，以及预训练模型对每个预测的置信度分数。每个文件的 Json 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-entities-detection-job \
    --job-name customentitiestest \
    --entity-recognizer-arn "arn:aws:comprehend:us-west-2:111122223333:entity-recognizer/entityrecognizer" \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/jobdata/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-IOrole"
```

的内容SampleFeedback1.txt：

```
"I've been on the AnyPhone app have had issues for 24 hours when trying to pay bill. Cannot make payment. Sigh. | Oh man! Lets get that app up and running. DM me, and we can get to work!"
```

的内容SampleFeedback2.txt：

```
"Hi, I have a discrepancy with my new bill. Could we get it sorted out? A rep added stuff I didnt sign up for when I did my AnyPhone 10 upgrade. | We can absolutely get this sorted!"
```

的内容SampleFeedback3.txt：

```
"Is the by 1 get 1 free AnySmartPhone promo still going on? | Hi Christian! It ended yesterday, send us a DM if you have any questions and we can take a look at your options!"
```

输出：

```
{
    "JobId": "019ea9edac758806850fa8a79ff83021",
```

```
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/019ea9edac758806850fa8a79ff83021",
    "JobStatus": "SUBMITTED"
}
```

output.txt 的内容，为便于阅读，采用了行间缩进：

```
{
"Entities": [
    {
    "BeginOffset": 17,
    "EndOffset": 25,
    "Score": 0.9999728210205924,
    "Text": "AnyPhone",
    "Type": "DEVICE"
    }
],
"File": "SampleFeedback1.txt",
"Line": 0
}
{
"Entities": [
    {
    "BeginOffset": 123,
    "EndOffset": 133,
    "Score": 0.9999892116761524,
    "Text": "AnyPhone 10",
    "Type": "DEVICE"
    }
],
"File": "SampleFeedback2.txt",
"Line": 0
}
{
"Entities": [
    {
    "BeginOffset": 23,
    "EndOffset": 35,
    "Score": 0.9999971389852362,
    "Text": "AnySmartPhone",
    "Type": "DEVICE"
    }
],
```

```
"File": "SampleFeedback3.txt",
"Line": 0
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的自定义实体识别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartEntitiesDetectionJob。

start-events-detection-job

以下代码示例演示了如何使用 start-events-detection-job。

Amazon CLI

启动异步事件检测作业

以下 start-events-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步事件检测作业。可能的目标事件类型包括 BANKRUPCTY、EMPLOYMENT、CORPORATE_ACQUISITION、INVESTMENT_GENERAL、CORPORATE_ME 和 STOCK_SPLIT。此示例中的 S3 存储桶包含 SampleText1.txt、SampleText2.txt 和 SampleText3.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 SampleText1.txt.out、SampleText2.txt.out 和 SampleText3.txt.out。每个文件的 JSON 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-events-detection-job \
    --job-name events-detection-1 \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/EventsData" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole \
    --language-code en \
    --target-event-
types "BANKRUPTCY" "EMPLOYMENT" "CORPORATE_ACQUISITION" "CORPORATE_MERGER" "INVESTMENT_GENER
```

的内容SampleText1.txt：

```
"Company AnyCompany grew by increasing sales and through acquisitions. After
 purchasing competing firms in 2020, AnyBusiness, a part of the AnyBusinessGroup,
 gave Jane Does firm a going rate of one cent a gallon or forty-two cents a barrel."
```

的内容SampleText2.txt：

```
"In 2021, AnyCompany officially purchased AnyBusiness for 100 billion dollars,
 surprising and exciting the shareholders."
```

的内容SampleText3.txt：

```
"In 2022, AnyCompany stock crashed 50. Eventually later that year they filed for
 bankruptcy."
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:events-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

SampleText1.txt.out 的内容，为便于阅读，采用了行间缩进：

```
{
    "Entities": [
        {
        "Mentions": [
            {
            "BeginOffset": 8,
            "EndOffset": 18,
            "Score": 0.99977,
            "Text": "AnyCompany",
            "Type": "ORGANIZATION",
            "GroupScore": 1
            },
            {
            "BeginOffset": 112,
            "EndOffset": 123,
            "Score": 0.999747,
            "Text": "AnyBusiness",
            "Type": "ORGANIZATION",
            "GroupScore": 0.979826
            },
            {
```

```
            "BeginOffset": 171,
            "EndOffset": 175,
            "Score": 0.999615,
            "Text": "firm",
            "Type": "ORGANIZATION",
            "GroupScore": 0.871647
            }
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 97,
            "EndOffset": 102,
            "Score": 0.987687,
            "Text": "firms",
            "Type": "ORGANIZATION",
            "GroupScore": 1
            }
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 103,
            "EndOffset": 110,
            "Score": 0.999458,
            "Text": "in 2020",
            "Type": "DATE",
            "GroupScore": 1
            }
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 160,
            "EndOffset": 168,
            "Score": 0.999649,
            "Text": "John Doe",
            "Type": "PERSON",
            "GroupScore": 1
            }
        ]
```

```
        }
    ],
    "Events": [
        {
        "Type": "CORPORATE_ACQUISITION",
        "Arguments": [
            {
            "EntityIndex": 0,
            "Role": "INVESTOR",
            "Score": 0.99977
            }
        ],
        "Triggers": [
            {
            "BeginOffset": 56,
            "EndOffset": 68,
            "Score": 0.999967,
            "Text": "acquisitions",
            "Type": "CORPORATE_ACQUISITION",
            "GroupScore": 1
            }
        ]
        },
        {
        "Type": "CORPORATE_ACQUISITION",
        "Arguments": [
            {
            "EntityIndex": 1,
            "Role": "INVESTEE",
            "Score": 0.987687
            },
            {
            "EntityIndex": 2,
            "Role": "DATE",
            "Score": 0.999458
            },
            {
            "EntityIndex": 3,
            "Role": "INVESTOR",
            "Score": 0.999649
            }
        ],
        "Triggers": [
            {
```

```
            "BeginOffset": 76,
            "EndOffset": 86,
            "Score": 0.999973,
            "Text": "purchasing",
            "Type": "CORPORATE_ACQUISITION",
            "GroupScore": 1
            }
        ]
        }
    ],
    "File": "SampleText1.txt",
    "Line": 0
}
```

的内容SampleText2.txt.out：

```
{
    "Entities": [
        {
        "Mentions": [
            {
            "BeginOffset": 0,
            "EndOffset": 7,
            "Score": 0.999473,
            "Text": "In 2021",
            "Type": "DATE",
            "GroupScore": 1
            }
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 9,
            "EndOffset": 19,
            "Score": 0.999636,
            "Text": "AnyCompany",
            "Type": "ORGANIZATION",
            "GroupScore": 1
            }
        ]
        },
        {
```

```
        "Mentions": [
            {
            "BeginOffset": 45,
            "EndOffset": 56,
            "Score": 0.999712,
            "Text": "AnyBusiness",
            "Type": "ORGANIZATION",
            "GroupScore": 1
            }
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 61,
            "EndOffset": 80,
            "Score": 0.998886,
            "Text": "100 billion dollars",
            "Type": "MONETARY_VALUE",
            "GroupScore": 1
            }
        ]
        }
    ],
    "Events": [
        {
        "Type": "CORPORATE_ACQUISITION",
        "Arguments": [
            {
            "EntityIndex": 3,
            "Role": "AMOUNT",
            "Score": 0.998886
            },
            {
            "EntityIndex": 2,
            "Role": "INVESTEE",
            "Score": 0.999712
            },
            {
            "EntityIndex": 0,
            "Role": "DATE",
            "Score": 0.999473
            },
            {
```

```
            "EntityIndex": 1,
            "Role": "INVESTOR",
            "Score": 0.999636
            }
        ],
        "Triggers": [
            {
            "BeginOffset": 31,
            "EndOffset": 40,
            "Score": 0.99995,
            "Text": "purchased",
            "Type": "CORPORATE_ACQUISITION",
            "GroupScore": 1
            }
        ]
        }
    ],
    "File": "SampleText2.txt",
    "Line": 0
}
```

的内容SampleText3.txt.out：

```
{
    "Entities": [
        {
        "Mentions": [
            {
            "BeginOffset": 9,
            "EndOffset": 19,
            "Score": 0.999774,
            "Text": "AnyCompany",
            "Type": "ORGANIZATION",
            "GroupScore": 1
            },
            {
            "BeginOffset": 66,
            "EndOffset": 70,
            "Score": 0.995717,
            "Text": "they",
            "Type": "ORGANIZATION",
            "GroupScore": 0.997626
            }
```

```
        ]
        },
        {
        "Mentions": [
            {
            "BeginOffset": 50,
            "EndOffset": 65,
            "Score": 0.999656,
            "Text": "later that year",
            "Type": "DATE",
            "GroupScore": 1
            }
        ]
        }
    ],
    "Events": [
        {
        "Type": "BANKRUPTCY",
        "Arguments": [
            {
            "EntityIndex": 1,
            "Role": "DATE",
            "Score": 0.999656
            },
            {
            "EntityIndex": 0,
            "Role": "FILER",
            "Score": 0.995717
            }
        ],
        "Triggers": [
            {
            "BeginOffset": 81,
            "EndOffset": 91,
            "Score": 0.999936,
            "Text": "bankruptcy",
            "Type": "BANKRUPTCY",
            "GroupScore": 1
            }
        ]
        }
    ],
    "File": "SampleText3.txt",
    "Line": 0
```

```
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartEventsDetectionJob。

start-flywheel-iteration

以下代码示例演示了如何使用 start-flywheel-iteration。

Amazon CLI

启动飞轮迭代

以下 start-flywheel-iteration 示例启动飞轮迭代。此操作使用飞轮中的任何新数据集来训练新的模型版本。

```
aws comprehend start-flywheel-iteration \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-flywheel
```

输出：

```
{
    "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-flywheel",
    "FlywheelIterationId": "12345123TEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartFlywheelIteration。

start-key-phrases-detection-job

以下代码示例演示了如何使用 start-key-phrases-detection-job。

Amazon CLI

启动关键短语检测作业

以下 start-key-phrases-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步关键短语检测作业。此示例中的 S3 存储桶包含 Sampletext1.txt、Sampletext2.txt 和 Sampletext3.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 output.txt，其中包含了在每个文本文件中检测到的所有关键短语，以及预训练模型对每个预测的置信度分数。每个文件的 Json 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-key-phrases-detection-job \
    --job-name keyphrasesanalysistest1 \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role" \
    --language-code en
```

的内容Sampletext1.txt：

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card
 account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July
 31st."
```

的内容Sampletext2.txt：

```
"Dear Max, based on your autopay settings for your account Internet.org account, we
 will withdraw your payment on the due date from your bank account number XXXXXX1111
 with the routing number XXXXX0000. "
```

的内容Sampletext3.txt：

```
"Jane, please submit any customer feedback from this weekend to Sunshine Spa, 123
 Main St, Anywhere and send comments to Alice at AnySpa@example.com."
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
```

```
}
```

output.txt 的内容，为便于阅读，采用了行间缩进：

```
{
    "File": "SampleText1.txt",
    "KeyPhrases": [
        {
        "BeginOffset": 6,
        "EndOffset": 15,
        "Score": 0.9748965572679326,
        "Text": "Zhang Wei"
        },
        {
        "BeginOffset": 22,
        "EndOffset": 26,
        "Score": 0.9997344722354619,
        "Text": "John"
        },
        {
        "BeginOffset": 28,
        "EndOffset": 62,
        "Score": 0.9843791074032948,
        "Text": "Your AnyCompany Financial Services"
        },
        {
        "BeginOffset": 64,
        "EndOffset": 107,
        "Score": 0.8976122401721824,
        "Text": "LLC credit card account 1111-XXXX-1111-XXXX"
        },
        {
        "BeginOffset": 112,
        "EndOffset": 129,
        "Score": 0.9999612982629748,
        "Text": "a minimum payment"
        },
        {
        "BeginOffset": 133,
        "EndOffset": 139,
        "Score": 0.99975728947036,
        "Text": "$24.53"
        },
```

```
        {
        "BeginOffset": 155,
        "EndOffset": 164,
        "Score": 0.9940866241449973,
        "Text": "July 31st"
        }
    ],
    "Line": 0
    }
    {
    "File": "SampleText2.txt",
    "KeyPhrases": [
        {
        "BeginOffset": 0,
        "EndOffset": 8,
        "Score": 0.9974021100118472,
        "Text": "Dear Max"
        },
        {
        "BeginOffset": 19,
        "EndOffset": 40,
        "Score": 0.9961120519515884,
        "Text": "your autopay settings"
        },
        {
        "BeginOffset": 45,
        "EndOffset": 78,
        "Score": 0.9980620070116009,
        "Text": "your account Internet.org account"
        },
        {
        "BeginOffset": 97,
        "EndOffset": 109,
        "Score": 0.999919660140754,
        "Text": "your payment"
        },
        {
        "BeginOffset": 113,
        "EndOffset": 125,
        "Score": 0.9998370719754205,
        "Text": "the due date"
        },
        {
        "BeginOffset": 131,
```

```
        "EndOffset": 166,
        "Score": 0.9955068678502509,
        "Text": "your bank account number XXXXXX1111"
        },
        {
        "BeginOffset": 172,
        "EndOffset": 200,
        "Score": 0.8653433315829526,
        "Text": "the routing number XXXXX0000"
        }
    ],
    "Line": 0
    }
    {
    "File": "SampleText3.txt",
    "KeyPhrases": [
        {
        "BeginOffset": 0,
        "EndOffset": 4,
        "Score": 0.9142947833681668,
        "Text": "Jane"
        },
        {
        "BeginOffset": 20,
        "EndOffset": 41,
        "Score": 0.9984325676596763,
        "Text": "any customer feedback"
        },
        {
        "BeginOffset": 47,
        "EndOffset": 59,
        "Score": 0.9998782448150636,
        "Text": "this weekend"
        },
        {
        "BeginOffset": 63,
        "EndOffset": 75,
        "Score": 0.99866741830757,
        "Text": "Sunshine Spa"
        },
        {
        "BeginOffset": 77,
        "EndOffset": 88,
        "Score": 0.9695803485466054,
```

```
        "Text": "123 Main St"
        },
        {
        "BeginOffset": 108,
        "EndOffset": 116,
        "Score": 0.9997065928550928,
        "Text": "comments"
        },
        {
        "BeginOffset": 120,
        "EndOffset": 125,
        "Score": 0.9993466833825161,
        "Text": "Alice"
        },
        {
        "BeginOffset": 129,
        "EndOffset": 144,
        "Score": 0.9654563612885667,
        "Text": "AnySpa@example.com"
        }
    ],
    "Line": 0
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartKeyPhrasesDetectionJob。

start-pii-entities-detection-job

以下代码示例演示了如何使用 start-pii-entities-detection-job。

Amazon CLI

启动异步 PII 检测作业

以下 start-pii-entities-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步个人身份信息（PII）实体检测作业。此示例中的 S3 存储桶包含 Sampletext1.txt、Sampletext2.txt 和 Sampletext3.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 SampleText1.txt.out、SampleText2.txt.out 和 SampleText3.txt.out，列出了每个

文本文件中的命名实体。每个文件的 Json 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-pii-entities-detection-job \
    --job-name entities_test \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --language-code en \
    --mode ONLY_OFFSETS
```

的内容Sampletext1.txt：

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card
 account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July
 31st."
```

的内容Sampletext2.txt：

```
"Dear Max, based on your autopay settings for your account Internet.org account, we
 will withdraw your payment on the due date from your bank account number XXXXXX1111
 with the routing number XXXXX0000. "
```

的内容Sampletext3.txt：

```
"Jane, please submit any customer feedback from this weekend to Sunshine Spa, 123
 Main St, Anywhere and send comments to Alice at AnySpa@example.com."
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

SampleText1.txt.out 的内容，为便于阅读，采用了行间缩进：

```
{
    "Entities": [
        {
        "BeginOffset": 6,
        "EndOffset": 15,
        "Type": "NAME",
        "Score": 0.9998490510222595
        },
        {
        "BeginOffset": 22,
        "EndOffset": 26,
        "Type": "NAME",
        "Score": 0.9998937958019426
        },
        {
        "BeginOffset": 88,
        "EndOffset": 107,
        "Type": "CREDIT_DEBIT_NUMBER",
        "Score": 0.9554297245278491
        },
        {
        "BeginOffset": 155,
        "EndOffset": 164,
        "Type": "DATE_TIME",
        "Score": 0.9999720462925257
        }
    ],
    "File": "SampleText1.txt",
    "Line": 0
}
```


SampleText2.txt.out 的内容，为便于阅读，采用了行间缩进：


```
{
    "Entities": [
        {
        "BeginOffset": 5,
        "EndOffset": 8,
        "Type": "NAME",
        "Score": 0.9994390774924007
        },
        {
        "BeginOffset": 58,
```

```
        "EndOffset": 70,
        "Type": "URL",
        "Score": 0.9999958276922101
        },
        {
        "BeginOffset": 156,
        "EndOffset": 166,
        "Type": "BANK_ACCOUNT_NUMBER",
        "Score": 0.9999721058045592
        },
        {
        "BeginOffset": 191,
        "EndOffset": 200,
        "Type": "BANK_ROUTING",
        "Score": 0.9998968945989909
        }
    ],
    "File": "SampleText2.txt",
    "Line": 0
}
```

SampleText3.txt.out 的内容，为便于阅读，采用了行间缩进：

```
{
    "Entities": [
        {
        "BeginOffset": 0,
        "EndOffset": 4,
        "Type": "NAME",
        "Score": 0.999949934606805
        },
        {
        "BeginOffset": 77,
        "EndOffset": 88,
        "Type": "ADDRESS",
        "Score": 0.9999035300466904
        },
        {
        "BeginOffset": 120,
        "EndOffset": 125,
        "Type": "NAME",
        "Score": 0.9998203838716296
        },
```

```
        {
        "BeginOffset": 129,
        "EndOffset": 144,
        "Type": "EMAIL",
        "Score": 0.9998313473105228
        }
    ],
    "File": "SampleText3.txt",
    "Line": 0
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPiiEntitiesDetectionJob。

start-sentiment-detection-job

以下代码示例演示了如何使用 start-sentiment-detection-job。

Amazon CLI

启动异步情绪分析作业

以下 start-sentiment-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步情绪分析检测作业。此示例中的 S3 存储桶文件夹包含 SampleMovieReview1.txt、SampleMovieReview2.txt 和 SampleMovieReview3.txt。作业完成后，文件夹 output 将放置在 --output-data-config 标签指定的位置。该文件夹包含 output.txt，其中包含了每个文本文件中的主导情绪，以及预训练模型对每个预测的置信度分数。每个文件的 Json 输出打印在一行上，但是为了便于阅读，此处设置了格式。

```
aws comprehend start-sentiment-detection-job \
    --job-name example-sentiment-detection-job \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/MovieData" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role
```

的内容SampleMovieReview1.txt：

```
"The film, AnyMovie2, is fairly predictable and just okay."
```

的内容SampleMovieReview2.txt：

```
"AnyMovie2 is the essential sci-fi film that I grew up watching when I was a kid. I
 highly recommend this movie."
```

的内容SampleMovieReview3.txt：

```
"Don't get fooled by the 'awards' for AnyMovie2. All parts of the film were poorly
 stolen from other modern directors."
```

输出：

```
{
    "JobId": "0b5001e25f62ebb40631a9a1a7fde7b3",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-detection-
job/0b5001e25f62ebb40631a9a1a7fde7b3",
    "JobStatus": "SUBMITTED"
}
```

output.txt 的内容，为便于阅读，采用了行间缩进：

```
{
    "File": "SampleMovieReview1.txt",
        "Line": 0,
        "Sentiment": "MIXED",
        "SentimentScore": {
            "Mixed": 0.6591159105300903,
            "Negative": 0.26492202281951904,
            "Neutral": 0.035430654883384705,
            "Positive": 0.04053137078881264
            }
        }
    {
    "File": "SampleMovieReview2.txt",
        "Line": 0,
        "Sentiment": "POSITIVE",
        "SentimentScore": {
            "Mixed": 0.000008718466233403888,
            "Negative": 0.00006134175055194646,
```

```
            "Neutral": 0.0002941041602753103,
            "Positive": 0.9996358156204224
            }
        }
    {
    "File": "SampleMovieReview3.txt",
        "Line": 0,
        "Sentiment": "NEGATIVE",
        "SentimentScore": {
            "Mixed": 0.004146667663007975,
            "Negative": 0.9645107984542847,
            "Neutral": 0.016559595242142677,
            "Positive": 0.014782938174903393
        }
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSentimentDetectionJob。

start-targeted-sentiment-detection-job

以下代码示例演示了如何使用 start-targeted-sentiment-detection-job。

Amazon CLI

启动异步目标情绪分析作业

以下 start-targeted-sentiment-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步目标情绪分析检测作业。此示例中的 S3 存储桶文件夹包含 SampleMovieReview1.txt、SampleMovieReview2.txt 和 SampleMovieReview3.txt。作业完成后，output.tar.gz 将放置在 --output-data-config 标签指定的位置。output.tar.gz 包含文件 SampleMovieReview1.txt.out、SampleMovieReview2.txt.out 和 SampleMovieReview3.txt.out，每个文件都包含单个输入文本文件的所有命名实体和关联情绪。

```
aws comprehend start-targeted-sentiment-detection-job \
    --job-name targeted_movie_review_analysis1 \
    --language-code en \
```

```
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/MovieData" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role
```

的内容SampleMovieReview1.txt：

```
"The film, AnyMovie, is fairly predictable and just okay."
```

的内容SampleMovieReview2.txt：

```
"AnyMovie is the essential sci-fi film that I grew up watching when I was a kid. I
 highly recommend this movie."
```

的内容SampleMovieReview3.txt：

```
"Don't get fooled by the 'awards' for AnyMovie. All parts of the film were poorly
 stolen from other modern directors."
```

输出：

```
{
    "JobId": "0b5001e25f62ebb40631a9a1a7fde7b3",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/0b5001e25f62ebb40631a9a1a7fde7b3",
    "JobStatus": "SUBMITTED"
}
```

SampleMovieReview1.txt.out 的内容，为便于阅读，采用了行间缩进：

```
{
    "Entities": [
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 4,
            "EndOffset": 8,
            "Score": 0.994972,
```

```
            "GroupScore": 1,
            "Text": "film",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            }
        ]
        },
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 10,
            "EndOffset": 18,
            "Score": 0.631368,
            "GroupScore": 1,
            "Text": "AnyMovie",
            "Type": "ORGANIZATION",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                "Mixed": 0.001729,
                "Negative": 0.000001,
                "Neutral": 0.000318,
                "Positive": 0.997952
                }
            }
            }
        ]
        }
    ],
    "File": "SampleMovieReview1.txt",
    "Line": 0
}
```

SampleMovieReview2.txt.out 的内容，为便于阅读，采用了行间缩进：


```
{
    "Entities": [
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 0,
            "EndOffset": 8,
            "Score": 0.854024,
            "GroupScore": 1,
            "Text": "AnyMovie",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 0.000007,
                "Positive": 0.999993
                }
            }
            },
            {
            "BeginOffset": 104,
            "EndOffset": 109,
            "Score": 0.999129,
            "GroupScore": 0.502937,
            "Text": "movie",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 0,
                "Positive": 1
                }
            }
            },
            {
```

```
            "BeginOffset": 33,
            "EndOffset": 37,
            "Score": 0.999823,
            "GroupScore": 0.999252,
            "Text": "film",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 0.000001,
                "Positive": 0.999999
                }
            }
            }
        ]
        },
        {
        "DescriptiveMentionIndex": [
            0,
            1,
            2
        ],
        "Mentions": [
            {
            "BeginOffset": 43,
            "EndOffset": 44,
            "Score": 0.999997,
            "GroupScore": 1,
            "Text": "I",
            "Type": "PERSON",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            },
            {
            "BeginOffset": 80,
```

```
            "EndOffset": 81,
            "Score": 0.999996,
            "GroupScore": 0.52523,
            "Text": "I",
            "Type": "PERSON",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            },
            {
            "BeginOffset": 67,
            "EndOffset": 68,
            "Score": 0.999994,
            "GroupScore": 0.999499,
            "Text": "I",
            "Type": "PERSON",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            }
        ]
        },
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 75,
            "EndOffset": 78,
            "Score": 0.999978,
            "GroupScore": 1,
```

```
            "Text": "kid",
            "Type": "PERSON",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            }
        ]
        }
    ],
    "File": "SampleMovieReview2.txt",
    "Line": 0
}
```

SampleMovieReview3.txt.out 的内容，为便于阅读，采用了行间缩进：

```
{
    "Entities": [
        {
        "DescriptiveMentionIndex": [
            1
        ],
        "Mentions": [
            {
            "BeginOffset": 64,
            "EndOffset": 68,
            "Score": 0.992953,
            "GroupScore": 0.999814,
            "Text": "film",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0.000004,
                "Negative": 0.010425,
                "Neutral": 0.989543,
                "Positive": 0.000027
                }
```

```
            }
            },
            {
            "BeginOffset": 37,
            "EndOffset": 45,
            "Score": 0.999782,
            "GroupScore": 1,
            "Text": "AnyMovie",
            "Type": "ORGANIZATION",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                "Mixed": 0.000095,
                "Negative": 0.039847,
                "Neutral": 0.000673,
                "Positive": 0.959384
                }
            }
            }
        ]
        },
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 47,
            "EndOffset": 50,
            "Score": 0.999991,
            "GroupScore": 1,
            "Text": "All",
            "Type": "QUANTITY",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0.000001,
                "Negative": 0.000001,
                "Neutral": 0.999998,
                "Positive": 0
                }
            }
            }
        ]
```

```
        },
        {
        "DescriptiveMentionIndex": [
            0
        ],
        "Mentions": [
            {
            "BeginOffset": 106,
            "EndOffset": 115,
            "Score": 0.542083,
            "GroupScore": 1,
            "Text": "directors",
            "Type": "PERSON",
            "MentionSentiment": {
                "Sentiment": "NEUTRAL",
                "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
                }
            }
            }
        ]
        }
    ],
    "File": "SampleMovieReview3.txt",
    "Line": 0
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartTargetedSentimentDetectionJob。

`start-topics-detection-job`

以下代码示例演示了如何使用 `start-topics-detection-job`。

Amazon CLI

启动主题检测分析作业

以下 start-topics-detection-job 示例为位于 --input-data-config 标签指定地址的所有文件启动异步主题检测作业。作业完成后，文件夹 output 将放置在 --ouput-data-config 标签指定的位置。output 包含 topic-terms.csv 和 doc-topics.csv。第一个输出文件 topic-terms.csv 是集合中的主题列表。对于每个主题，默认情况下，该列表按权重排列主题列出根据其的热门术语。第二个文件 doc-topics.csv 列出了与主题相关的文档以及与该主题相关的文档比例。

```
aws comprehend start-topics-detection-job \
    --job-name example_topics_detection_job \
    --language-code en \
    --input-data-config "S3Uri=s3://amzn-s3-demo-bucket/" \
    --output-data-config "S3Uri=s3://amzn-s3-demo-destination-bucket/testfolder/" \
    --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
    --language-code en
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobStatus": "SUBMITTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的主题建模。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartTopicsDetectionJob。

stop-dominant-language-detection-job

以下代码示例演示了如何使用 stop-dominant-language-detection-job。

Amazon CLI

停止异步主要语言检测作业

以下 stop-dominant-language-detection-job 示例停止正在进行的异步主要语言检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-dominant-language-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDominantLanguageDetectionJob。

stop-entities-detection-job

以下代码示例演示了如何使用 stop-entities-detection-job。

Amazon CLI

停止异步实体检测作业

以下 stop-entities-detection-job 示例停止正在进行的异步实体检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-entities-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopEntitiesDetectionJob。

stop-events-detection-job

以下代码示例演示了如何使用 stop-events-detection-job。

Amazon CLI

停止异步事件检测作业

以下 stop-events-detection-job 示例停止正在进行的异步事件检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-events-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopEventsDetectionJob。

stop-key-phrases-detection-job

以下代码示例演示了如何使用 stop-key-phrases-detection-job。

Amazon CLI

停止异步关键短语检测作业

以下 stop-key-phrases-detection-job 示例停止正在进行的异步关键短语检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-key-phrases-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopKeyPhrasesDetectionJob。

stop-pii-entities-detection-job

以下代码示例演示了如何使用 stop-pii-entities-detection-job。

Amazon CLI

停止异步 PII 实体检测作业

以下 stop-pii-entities-detection-job 示例停止正在进行的异步 PII 实体检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-pii-entities-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopPiiEntitiesDetectionJob。

stop-sentiment-detection-job

以下代码示例演示了如何使用 stop-sentiment-detection-job。

Amazon CLI

停止异步情绪检测作业

以下 stop-sentiment-detection-job 示例停止正在进行的异步情绪检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-sentiment-detection-job \
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopSentimentDetectionJob。

stop-targeted-sentiment-detection-job

以下代码示例演示了如何使用 stop-targeted-sentiment-detection-job。

Amazon CLI

停止异步目标情绪检测作业

以下 stop-targeted-sentiment-detection-job 示例停止正在进行的异步目标情绪检测作业。如果当前作业状态为 IN_PROGRESS，则该作业被标记为终止并进入 STOP_REQUESTED 状态。如果作业在可以停止之前就完成了，则会进入 COMPLETED 状态。

```
aws comprehend stop-targeted-sentiment-detection-job \
```

```
    --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

输出：

```
{
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE,
    "JobStatus": "STOP_REQUESTED"
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的 Amazon Comprehend 洞察的异步分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopTargetedSentimentDetectionJob。

stop-training-document-classifier

以下代码示例演示了如何使用 stop-training-document-classifier。

Amazon CLI

停止训练文档分类器模型

以下 stop-training-document-classifier 示例停止训练正在进行的文档分类器模型。

```
aws comprehend stop-training-document-classifier
    --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的创建和管理自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopTrainingDocumentClassifier。

stop-training-entity-recognizer

以下代码示例演示了如何使用 stop-training-entity-recognizer。

Amazon CLI

停止训练实体识别器模型

以下 stop-training-entity-recognizer 示例停止训练正在进行的实体识别器模型。

```
aws comprehend stop-training-entity-recognizer
    --entity-recognizer-arn "arn:aws:comprehend:us-west-2:111122223333:entity-recognizer/examplerecognizer1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的创建和管理自定义模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopTrainingEntityRecognizer。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

示例 1：标记资源

以下 tag-resource 示例为 Amazon Comprehend 资源添加一个标签。

```
aws comprehend tag-resource \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/1 \
    --tags Key=Location,Value=Seattle
```

此命令没有输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的标记资源。

示例 2：为资源添加多个标记

以下 tag-resource 示例为 Amazon Comprehend 资源添加多个标签。

```
aws comprehend tag-resource \
    --resource-arn "arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/1" \
    --tags Key=location,Value=Seattle Key=Department,Value=Finance
```

此命令没有输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：从资源中移除单个标签

以下 untag-resource 示例从 Amazon Comprehend 资源中移除一个标签。

```
aws comprehend untag-resource \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/1
    --tag-keys Location
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的标记资源。

示例 2：从资源中删除多个标签

以下 untag-resource 示例从 Amazon Comprehend 资源中移除多个标签。

```
aws comprehend untag-resource \
    --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/1
    --tag-keys Location Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-endpoint

以下代码示例演示了如何使用 update-endpoint。

Amazon CLI

示例 1：更新端点的推理单元

以下 update-endpoint 示例更新有关端点的信息。在此示例中，增加了推理单元的数量。

```
aws comprehend update-endpoint \
    --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-endpoint/example-classifier-endpoint
    --desired-inference-units 2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

示例 2：更新端点的活动模型

以下 update-endpoint 示例更新有关端点的信息。在此示例中，更改了活动模型。

```
aws comprehend update-endpoint \
    --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-endpoint/example-classifier-endpoint
    --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier-new
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的管理 Amazon Comprehend 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEndpoint。

update-flywheel

以下代码示例演示了如何使用 update-flywheel。

Amazon CLI

更新飞轮配置

以下 update-flywheel 示例更新飞轮配置。在此示例中，更新了飞轮的活动模型。

```
aws comprehend update-flywheel \
    --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-flywheel-1 \
    --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/new-example-classifier-model
```

输出：

```
{
    "FlywheelProperties": {
        "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-entity",
        "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/example-classifier/version/new-example-classifier-model",
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-example-role",
        "TaskConfig": {
            "LanguageCode": "en",
            "DocumentClassificationConfig": {
                "Mode": "MULTI_CLASS"
            }
        },
        "DataLakeS3Uri": "s3://amzn-s3-demo-bucket/flywheel-entity/schemaVersion=1/20230616T200543Z/",
        "DataSecurityConfig": {},
        "Status": "ACTIVE",
        "ModelType": "DOCUMENT_CLASSIFIER",
        "CreationTime": "2023-06-16T20:05:43.242000+00:00",
        "LastModifiedTime": "2023-06-19T04:00:43.027000+00:00",
        "LatestFlywheelIteration": "20230619T040032Z"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend 开发人员指南》中的飞轮概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFlywheel。

使用 Amazon CLI 的 Amazon Comprehend Medical 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Comprehend Medical 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

describe-entities-detection-v2-job

以下代码示例演示了如何使用 describe-entities-detection-v2-job。

Amazon CLI

描述实体检测作业

以下 describe-entities-detection-v2-job 示例显示与异步实体检测作业关联的属性。

```
aws comprehendmedical describe-entities-detection-v2-job \
    --job-id "ab9887877365fe70299089371c043b96"
```

输出：

```
{
    "ComprehendMedicalAsyncJobProperties": {
        "JobId": "ab9887877365fe70299089371c043b96",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2020-03-18T21:20:15.614000+00:00",
        "EndTime": "2020-03-18T21:27:07.350000+00:00",
        "ExpirationTime": "2020-07-16T21:20:15+00:00",
        "InputDataConfig": {
            "S3Bucket": "comp-med-input",
            "S3Key": ""
        },
        "OutputDataConfig": {
            "S3Bucket": "comp-med-output",
            "S3Key": "867139942017-EntitiesDetection-
ab9887877365fe70299089371c043b96/"
        },
```

```
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
        "ModelVersion": "DetectEntitiesModelV20190930"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEntitiesDetectionV2Job。

describe-icd10-cm-inference-job

以下代码示例演示了如何使用 describe-icd10-cm-inference-job。

Amazon CLI

描述 ICD-10-CM 推理作业

以下 describe-icd10-cm-inference-job 示例描述具有指定作业 ID 的请求推理作业的属性。

```
aws comprehendmedical describe-icd10-cm-inference-job \
    --job-id "5780034166536cdb52ffa3295a1b00a7"
```

输出：

```
{
    "ComprehendMedicalAsyncJobProperties": {
        "JobId": "5780034166536cdb52ffa3295a1b00a7",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2020-05-18T21:20:15.614000+00:00",
        "EndTime": "2020-05-18T21:27:07.350000+00:00",
        "ExpirationTime": "2020-09-16T21:20:15+00:00",
        "InputDataConfig": {
            "S3Bucket": "comp-med-input",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
        "OutputDataConfig": {
            "S3Bucket": "comp-med-output",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
```

```
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
        "ModelVersion":  "0.1.0"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIcd10CmInferenceJob。

describe-phi-detection-job

以下代码示例演示了如何使用 describe-phi-detection-job。

Amazon CLI

描述 PHI 检测作业

以下 describe-phi-detection-job 示例显示与异步受保护健康信息（PHI）检测作业关联的属性。

```
aws comprehendmedical describe-phi-detection-job \
    --job-id "4750034166536cdb52ffa3295a1b00a3"
```

输出：

```
{
    "ComprehendMedicalAsyncJobProperties": {
        "JobId": "4750034166536cdb52ffa3295a1b00a3",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
        "EndTime": "2020-03-19T20:45:07.894000+00:00",
        "ExpirationTime": "2020-07-17T20:38:37+00:00",
        "InputDataConfig": {
            "S3Bucket": "comp-med-input",
            "S3Key": ""
        },
        "OutputDataConfig": {
            "S3Bucket": "comp-med-output",
            "S3Key": "867139942017-PHIDetection-4750034166536cdb52ffa3295a1b00a3/"
        },
        "LanguageCode": "en",
```

```
        "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
        "ModelVersion": "PHIModelV20190903"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePhiDetectionJob。

describe-rx-norm-inference-job

以下代码示例演示了如何使用 describe-rx-norm-inference-job。

Amazon CLI

描述 RxNorm 推理作业

以下 describe-rx-norm-inference-job 示例描述具有指定作业 ID 的请求推理作业的属性。

```
aws comprehendmedical describe-rx-norm-inference-job \
    --job-id "eg8199877365fc70299089371c043b96"
```

输出：

```
{
    "ComprehendMedicalAsyncJobProperties": {
        "JobId": "g8199877365fc70299089371c043b96",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2020-05-18T21:20:15.614000+00:00",
        "EndTime": "2020-05-18T21:27:07.350000+00:00",
        "ExpirationTime": "2020-09-16T21:20:15+00:00",
        "InputDataConfig": {
            "S3Bucket": "comp-med-input",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
        "OutputDataConfig": {
            "S3Bucket": "comp-med-output",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
```

```
        "ModelVersion": "0.0.0"
    }
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRxNormInferenceJob。

describe-snomedct-inference-job

以下代码示例演示了如何使用 describe-snomedct-inference-job。

Amazon CLI

描述 SNOMED CT 推理作业

以下 describe-snomedct-inference-job 示例描述具有指定作业 ID 的请求推理作业的属性。

```
aws comprehendmedical describe-snomedct-inference-job \
    --job-id "2630034166536cdb52ffa3295a1b00a7"
```

输出：

```
{
    "ComprehendMedicalAsyncJobProperties": {
        "JobId": "2630034166536cdb52ffa3295a1b00a7",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2021-12-18T21:20:15.614000+00:00",
        "EndTime": "2021-12-18T21:27:07.350000+00:00",
        "ExpirationTime": "2022-05-16T21:20:15+00:00",
        "InputDataConfig": {
            "S3Bucket": "comp-med-input",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
        "OutputDataConfig": {
            "S3Bucket": "comp-med-output",
            "S3Key": "AKIAIOSFODNN7EXAMPLE"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
        "ModelVersion":  "0.1.0"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnomedctInferenceJob。

`detect-entities-v2`

以下代码示例演示了如何使用 `detect-entities-v2`。

Amazon CLI

示例 1：直接检测文本中的实体

以下 `detect-entities-v2` 示例显示检测到的实体，并直接从输入文本中根据类型对其进行标记。

```
aws comprehendmedical detect-entities-v2 \
    --text "Sleeping trouble on present dosage of Clonidine. Severe rash on face and
 leg, slightly itchy."
```

输出：

```
{
    "Id": 0,
    "BeginOffset": 38,
    "EndOffset": 47,
    "Score": 0.9942955374717712,
    "Text": "Clonidine",
    "Category": "MEDICATION",
    "Type": "GENERIC_NAME",
    "Traits": []
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的检测实体版本 2。

示例 2：检测文件路径中的实体

以下 `detect-entities-v2` 示例显示检测到的实体，并根据文件路径中的类型对其进行标记。

```
aws comprehendmedical detect-entities-v2 \
```

```
    --text file://medical_entities.txt
```

的内容medical_entities.txt：

```
{
    "Sleeping trouble on present dosage of Clonidine. Severe rash on face and leg,
 slightly itchy."
}
```

输出：

```
{
    "Id": 0,
    "BeginOffset": 38,
    "EndOffset": 47,
    "Score": 0.9942955374717712,
    "Text": "Clonidine",
    "Category": "MEDICATION",
    "Type": "GENERIC_NAME",
    "Traits": []
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的检测实体版本 2。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectEntitiesV2。

detect-phi

以下代码示例演示了如何使用 detect-phi。

Amazon CLI

示例 1：直接检测文本中的受保护健康信息（PHI）

以下 detect-phi 示例直接显示输入文本中检测到的受保护健康信息（PHI）实体。

```
aws comprehendmedical detect-phi \
    --text "Patient Carlos Salazar presented with rash on his upper extremities and
 dry cough. He lives at 100 Main Street, Anytown, USA where he works from his home
 as a carpenter."
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 8,
            "EndOffset": 21,
            "Score": 0.9914507269859314,
            "Text": "Carlos Salazar",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "NAME",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 94,
            "EndOffset": 109,
            "Score": 0.871849775314331,
            "Text": "100 Main Street, Anytown, USA",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "ADDRESS",
            "Traits": []
        },
        {
            "Id": 2,
            "BeginOffset": 145,
            "EndOffset": 154,
            "Score": 0.8302185535430908,
            "Text": "carpenter",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "PROFESSION",
            "Traits": []
        }
    ],
    "ModelVersion": "0.0.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的检测 PHI。

示例 2：直接检测文件路径中的受保护健康信息（PHI）

以下 `detect-phi` 示例显示从文件路径中检测到的受保护健康信息（PHI）实体。

```
aws comprehendmedical detect-phi \
```

```
    --text file://phi.txt
```

的内容phi.txt：

```
"Patient Carlos Salazar presented with a rash on his upper extremities and a dry
 cough. He lives at 100 Main Street, Anytown, USA, where he works from his home as a
 carpenter."
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 8,
            "EndOffset": 21,
            "Score": 0.9914507269859314,
            "Text": "Carlos Salazar",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "NAME",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 94,
            "EndOffset": 109,
            "Score": 0.871849775314331,
            "Text": "100 Main Street, Anytown, USA",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "ADDRESS",
            "Traits": []
        },
        {
            "Id": 2,
            "BeginOffset": 145,
            "EndOffset": 154,
            "Score": 0.8302185535430908,
            "Text": "carpenter",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "PROFESSION",
            "Traits": []
        }
    ],
```

```
    "ModelVersion": "0.0.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的检测 PHI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectPhi。

`infer-icd10-cm`

以下代码示例演示了如何使用 `infer-icd10-cm`。

Amazon CLI

示例 1：直接检测文本中的医疗条件实体并将其链接到 ICD-10-CM 本体

以下 `infer-icd10-cm` 示例标记检测到的医疗条件实体，并将这些实体与 2019 年版《国际疾病分类临床修订版（ICD-10-CM）》中的代码关联。

```
aws comprehendmedical infer-icd10-cm \
    --text "The patient complains of abdominal pain, has a long-standing history of
 diabetes treated with Micronase daily."
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "Text": "abdominal pain",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9475538730621338,
            "BeginOffset": 28,
            "EndOffset": 42,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.6724207401275635
                }
            ],
            "ICD10CMConcepts": [
                {
```

```
                    "Description": "Unspecified abdominal pain",
                    "Code": "R10.9",
                    "Score": 0.6904221177101135
                },
                {
                    "Description": "Epigastric pain",
                    "Code": "R10.13",
                    "Score": 0.1364113688468933
                },
                {
                    "Description": "Generalized abdominal pain",
                    "Code": "R10.84",
                    "Score": 0.12508003413677216
                },
                {
                    "Description": "Left lower quadrant pain",
                    "Code": "R10.32",
                    "Score": 0.10063883662223816
                },
                {
                    "Description": "Lower abdominal pain, unspecified",
                    "Code": "R10.30",
                    "Score": 0.09933677315711975
                }
            ]
        },
        {
            "Id": 1,
            "Text": "diabetes",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9899052977561951,
            "BeginOffset": 75,
            "EndOffset": 83,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "DIAGNOSIS",
                    "Score": 0.9258432388305664
                }
            ],
            "ICD10CMConcepts": [
                {
                    "Description": "Type 2 diabetes mellitus without complications",
```

```
                    "Code": "E11.9",
                    "Score": 0.7158446311950684
                },
                {
                    "Description": "Family history of diabetes mellitus",
                    "Code": "Z83.3",
                    "Score": 0.5704703330993652
                },
                {
                    "Description": "Family history of other endocrine, nutritional
 and metabolic diseases",
                    "Code": "Z83.49",
                    "Score": 0.19856023788452148
                },
                {
                    "Description": "Type 1 diabetes mellitus with ketoacidosis
 without coma",
                    "Code": "E10.10",
                    "Score": 0.13285516202449799
                },
                {
                    "Description": "Type 2 diabetes mellitus with hyperglycemia",
                    "Code": "E11.65",
                    "Score": 0.0993388369679451
                }
            ]
        }
    ],
    "ModelVersion": "0.1.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的 Infer ICD10-CM。

示例 2：检测文件路径中的医疗条件实体并将其链接到 ICD-10-CM 本体

以下 `infer-icd-10-cm` 示例标记检测到的医疗条件实体，并将这些实体与 2019 年版《国际疾病分类临床修订版（ICD-10-CM）》中的代码关联。

```
aws comprehendmedical infer-icd10-cm \
    --text file://icd10cm.txt
```

的内容`icd10cm.txt`：

```
{
    "The patient complains of abdominal pain, has a long-standing history of
 diabetes treated with Micronase daily."
}
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "Text": "abdominal pain",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9475538730621338,
            "BeginOffset": 28,
            "EndOffset": 42,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.6724207401275635
                }
            ],
            "ICD10CMConcepts": [
                {
                    "Description": "Unspecified abdominal pain",
                    "Code": "R10.9",
                    "Score": 0.6904221177101135
                },
                {
                    "Description": "Epigastric pain",
                    "Code": "R10.13",
                    "Score": 0.1364113688468933
                },
                {
                    "Description": "Generalized abdominal pain",
                    "Code": "R10.84",
                    "Score": 0.12508003413677216
                },
                {
                    "Description": "Left lower quadrant pain",
                    "Code": "R10.32",
```

```
                    "Score": 0.10063883662223816
                },
                {
                    "Description": "Lower abdominal pain, unspecified",
                    "Code": "R10.30",
                    "Score": 0.09933677315711975
                }
            ]
        },
        {
            "Id": 1,
            "Text": "diabetes",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9899052977561951,
            "BeginOffset": 75,
            "EndOffset": 83,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "DIAGNOSIS",
                    "Score": 0.9258432388305664
                }
            ],
            "ICD10CMConcepts": [
                {
                    "Description": "Type 2 diabetes mellitus without complications",
                    "Code": "E11.9",
                    "Score": 0.7158446311950684
                },
                {
                    "Description": "Family history of diabetes mellitus",
                    "Code": "Z83.3",
                    "Score": 0.5704703330993652
                },
                {
                    "Description": "Family history of other endocrine, nutritional
and metabolic diseases",
                    "Code": "Z83.49",
                    "Score": 0.19856023788452148
                },
                {
                    "Description": "Type 1 diabetes mellitus with ketoacidosis
without coma",
```

```
                    "Code": "E10.10",
                    "Score": 0.13285516202449799
                },
                {
                    "Description": "Type 2 diabetes mellitus with hyperglycemia",
                    "Code": "E11.65",
                    "Score": 0.0993388369679451
                }
            ]
        }
    ],
    "ModelVersion": "0.1.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的 Infer-ICD10-CM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InferIcd10Cm。

`infer-rx-norm`

以下代码示例演示了如何使用 `infer-rx-norm`。

Amazon CLI

示例 1：直接检测文本中的药物实体并将其链接到 RxNorm

以下 `infer-rx-norm` 示例显示和标记检测到的药物实体，并将这些实体与美国国家医学图书馆 RxNorm 数据库中的概念标识符（RxCUI）关联。

```
aws comprehendmedical infer-rx-norm \
    --text "Patient reports taking Levothyroxine 125 micrograms p.o. once daily, but
 denies taking Synthroid."
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "Text": "Levothyroxine",
            "Category": "MEDICATION",
            "Type": "GENERIC_NAME",
            "Score": 0.9996285438537598,
```

```
            "BeginOffset": 23,
            "EndOffset": 36,
            "Attributes": [
                {
                    "Type": "DOSAGE",
                    "Score": 0.9892290830612183,
                    "RelationshipScore": 0.9997978806495667,
                    "Id": 1,
                    "BeginOffset": 37,
                    "EndOffset": 51,
                    "Text": "125 micrograms",
                    "Traits": []
                },
                {
                    "Type": "ROUTE_OR_MODE",
                    "Score": 0.9988924860954285,
                    "RelationshipScore": 0.998291552066803,
                    "Id": 2,
                    "BeginOffset": 52,
                    "EndOffset": 56,
                    "Text": "p.o.",
                    "Traits": []
                },
                {
                    "Type": "FREQUENCY",
                    "Score": 0.9953463673591614,
                    "RelationshipScore": 0.9999889135360718,
                    "Id": 3,
                    "BeginOffset": 57,
                    "EndOffset": 67,
                    "Text": "once daily",
                    "Traits": []
                }
            ],
            "Traits": [],
            "RxNormConcepts": [
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet",
                    "Code": "966224",
                    "Score": 0.9912070631980896
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Capsule",
                    "Code": "966405",
```

```
                    "Score": 0.8698278665542603
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Synthroid]",
                    "Code": "966191",
                    "Score": 0.7448257803916931
                },
                {
                    "Description": "levothyroxine",
                    "Code": "10582",
                    "Score": 0.7050482630729675
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Levoxyl]",
                    "Code": "966190",
                    "Score": 0.6921631693840027
                }
            ]
        },
        {
            "Id": 4,
            "Text": "Synthroid",
            "Category": "MEDICATION",
            "Type": "BRAND_NAME",
            "Score": 0.9946461319923401,
            "BeginOffset": 86,
            "EndOffset": 95,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "NEGATION",
                    "Score": 0.5167351961135864
                }
            ],
            "RxNormConcepts": [
                {
                    "Description": "Synthroid",
                    "Code": "224920",
                    "Score": 0.9462039470672607
                },
                {
```

```
                    "Description": "Levothyroxine Sodium 0.088 MG Oral Tablet
 [Synthroid]",
                    "Code": "966282",
                    "Score": 0.8309829235076904
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
 [Synthroid]",
                    "Code": "966191",
                    "Score": 0.4945160448551178
                },
                {
                    "Description": "Levothyroxine Sodium 0.05 MG Oral Tablet
 [Synthroid]",
                    "Code": "966247",
                    "Score": 0.3674522042274475
                },
                {
                    "Description": "Levothyroxine Sodium 0.025 MG Oral Tablet
 [Synthroid]",
                    "Code": "966158",
                    "Score": 0.2588822841644287
                }
            ]
        }
    ],
    "ModelVersion": "0.0.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的推理 RxNorm。

示例 2：检测文件路径中的药物实体并将其链接到 RxNorm

以下 `infer-rx-norm` 示例显示和标记检测到的药物实体，并将这些实体与美国国家医学图书馆 RxNorm 数据库中的概念标识符（RxCUI）关联。

```
aws comprehendmedical infer-rx-norm \
    --text file://rxnorm.txt
```

的内容`rxnorm.txt`：

```
{
```

```
    "Patient reports taking Levothyroxine 125 micrograms p.o. once daily, but denies
 taking Synthroid."
}
```

输出：

```
{
    "Entities": [
        {
            "Id": 0,
            "Text": "Levothyroxine",
            "Category": "MEDICATION",
            "Type": "GENERIC_NAME",
            "Score": 0.9996285438537598,
            "BeginOffset": 23,
            "EndOffset": 36,
            "Attributes": [
                {
                    "Type": "DOSAGE",
                    "Score": 0.9892290830612183,
                    "RelationshipScore": 0.9997978806495667,
                    "Id": 1,
                    "BeginOffset": 37,
                    "EndOffset": 51,
                    "Text": "125 micrograms",
                    "Traits": []
                },
                {
                    "Type": "ROUTE_OR_MODE",
                    "Score": 0.9988924860954285,
                    "RelationshipScore": 0.998291552066803,
                    "Id": 2,
                    "BeginOffset": 52,
                    "EndOffset": 56,
                    "Text": "p.o.",
                    "Traits": []
                },
                {
                    "Type": "FREQUENCY",
                    "Score": 0.9953463673591614,
                    "RelationshipScore": 0.9999889135360718,
                    "Id": 3,
                    "BeginOffset": 57,
```

```
                    "EndOffset": 67,
                    "Text": "once daily",
                    "Traits": []
                }
            ],
            "Traits": [],
            "RxNormConcepts": [
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet",
                    "Code": "966224",
                    "Score": 0.9912070631980896
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Capsule",
                    "Code": "966405",
                    "Score": 0.8698278665542603
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Synthroid]",
                    "Code": "966191",
                    "Score": 0.7448257803916931
                },
                {
                    "Description": "levothyroxine",
                    "Code": "10582",
                    "Score": 0.7050482630729675
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Levoxyl]",
                    "Code": "966190",
                    "Score": 0.6921631693840027
                }
            ]
        },
        {
            "Id": 4,
            "Text": "Synthroid",
            "Category": "MEDICATION",
            "Type": "BRAND_NAME",
            "Score": 0.9946461319923401,
            "BeginOffset": 86,
            "EndOffset": 95,
```

```
            "Attributes": [],
            "Traits": [
                {
                    "Name": "NEGATION",
                    "Score": 0.5167351961135864
                }
            ],
            "RxNormConcepts": [
                {
                    "Description": "Synthroid",
                    "Code": "224920",
                    "Score": 0.9462039470672607
                },
                {
                    "Description": "Levothyroxine Sodium 0.088 MG Oral Tablet
 [Synthroid]",
                    "Code": "966282",
                    "Score": 0.8309829235076904
                },
                {
                    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
 [Synthroid]",
                    "Code": "966191",
                    "Score": 0.4945160448551178
                },
                {
                    "Description": "Levothyroxine Sodium 0.05 MG Oral Tablet
 [Synthroid]",
                    "Code": "966247",
                    "Score": 0.3674522042274475
                },
                {
                    "Description": "Levothyroxine Sodium 0.025 MG Oral Tablet
 [Synthroid]",
                    "Code": "966158",
                    "Score": 0.2588822841644287
                }
            ]
        }
    ],
    "ModelVersion": "0.0.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的推理 RxNorm。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InferRxNorm。

`infer-snomedct`

以下代码示例演示了如何使用 `infer-snomedct`。

Amazon CLI

示例：直接检测文本中的实体并将其链接到 SNOMED CT 本体

以下 `infer-snomedct` 示例演示如何检测医学实体，并将其与 2021-03 版《医学系统命名法，临床术语（SNOMED CT）》中的概念关联。

```
aws comprehendmedical infer-snomedct \
    --text "The patient complains of abdominal pain, has a long-standing history of
 diabetes treated with Micronase daily."
```

输出：

```
{
    "Entities": [
        {
            "Id": 3,
            "BeginOffset": 26,
            "EndOffset": 40,
            "Score": 0.9598260521888733,
            "Text": "abdominal pain",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.6819021701812744
                }
            ]
        },
        {
            "Id": 4,
            "BeginOffset": 73,
            "EndOffset": 81,
```

```
            "Score": 0.9905840158462524,
            "Text": "diabetes",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Traits": [
                {
                    "Name": "DIAGNOSIS",
                    "Score": 0.9255214333534241
                }
            ]
        },
        {
            "Id": 1,
            "BeginOffset": 95,
            "EndOffset": 104,
            "Score": 0.6371926665306091,
            "Text": "Micronase",
            "Category": "MEDICATION",
            "Type": "BRAND_NAME",
            "Traits": [],
            "Attributes": [
                {
                    "Type": "FREQUENCY",
                    "Score": 0.9761165380477905,
                    "RelationshipScore": 0.9984188079833984,
                    "RelationshipType": "FREQUENCY",
                    "Id": 2,
                    "BeginOffset": 105,
                    "EndOffset": 110,
                    "Text": "daily",
                    "Category": "MEDICATION",
                    "Traits": []
                }
            ]
        }
    ],
    "UnmappedAttributes": [],
    "ModelVersion": "1.0.0"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的 InferSNOMEDCT。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InferSnomedct。

list-entities-detection-v2-jobs

以下代码示例演示了如何使用 list-entities-detection-v2-jobs。

Amazon CLI

列出实体检测作业

以下 list-entities-detection-v2-jobs 示例列出当前异步检测作业。

```
aws comprehendmedical list-entities-detection-v2-jobs
```

输出：

```
{
    "ComprehendMedicalAsyncJobPropertiesList": [
        {
            "JobId": "ab9887877365fe70299089371c043b96",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
            "EndTime": "2020-03-19T20:45:07.894000+00:00",
            "ExpirationTime": "2020-07-17T20:38:37+00:00",
            "InputDataConfig": {
                "S3Bucket": "comp-med-input",
                "S3Key": ""
            },
            "OutputDataConfig": {
                "S3Bucket": "comp-med-output",
                "S3Key": "867139942017-EntitiesDetection-
ab9887877365fe70299089371c043b96/"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
            "ModelVersion": "DetectEntitiesModelV20190930"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntitiesDetectionV2Jobs。

list-icd10-cm-inference-jobs

以下代码示例演示了如何使用 list-icd10-cm-inference-jobs。

Amazon CLI

列出所有当前 ICD-10-CM 推理作业

以下示例演示 list-icd10-cm-inference-jobs 操作如何返回当前异步 ICD-10-CM 批量推理作业的列表。

```
aws comprehendmedical list-icd10-cm-inference-jobs
```

输出：

```
{
    "ComprehendMedicalAsyncJobPropertiesList": [
        {
            "JobId": "5780034166536cdb52ffa3295a1b00a7",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
            "EndTime": "2020-05-19T20:45:07.894000+00:00",
            "ExpirationTime": "2020-09-17T20:38:37+00:00",
            "InputDataConfig": {
                "S3Bucket": "comp-med-input",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "OutputDataConfig": {
                "S3Bucket": "comp-med-output",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
            "ModelVersion":  "0.1.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIcd10CmInferenceJobs。

list-phi-detection-jobs

以下代码示例演示了如何使用 list-phi-detection-jobs。

Amazon CLI

列出受保护健康信息（PHI）检测作业

以下 list-phi-detection-jobs 示例列出当前受保护健康信息（PHI）检测作业

```
aws comprehendmedical list-phi-detection-jobs
```

输出：

```
{
    "ComprehendMedicalAsyncJobPropertiesList": [
        {
            "JobId": "4750034166536cdb52ffa3295a1b00a3",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
            "EndTime": "2020-03-19T20:45:07.894000+00:00",
            "ExpirationTime": "2020-07-17T20:38:37+00:00",
            "InputDataConfig": {
                "S3Bucket": "comp-med-input",
                "S3Key": ""
            },
            "OutputDataConfig": {
                "S3Bucket": "comp-med-output",
                "S3Key": "867139942017-
PHIDetection-4750034166536cdb52ffa3295a1b00a3/"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
            "ModelVersion": "PHIModelV20190903"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPhiDetectionJobs。

list-rx-norm-inference-jobs

以下代码示例演示了如何使用 list-rx-norm-inference-jobs。

Amazon CLI

列出所有当前的 Rx-Norm 推理作业

以下示例演示 list-rx-norm-inference-jobs 如何返回当前异步 Rx-Norm 批量推理作业的列表。

```
aws comprehendmedical list-rx-norm-inference-jobs
```

输出：

```
{
    "ComprehendMedicalAsyncJobPropertiesList": [
        {
            "JobId": "4980034166536cfb52gga3295a1b00a3",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
            "EndTime": "2020-05-19T20:45:07.894000+00:00",
            "ExpirationTime": "2020-09-17T20:38:37+00:00",
            "InputDataConfig": {
                "S3Bucket": "comp-med-input",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "OutputDataConfig": {
                "S3Bucket": "comp-med-output",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
            "ModelVersion": "0.0.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRxNormInferenceJobs。

`list-snomedct-inference-jobs`

以下代码示例演示了如何使用 `list-snomedct-inference-jobs`。

Amazon CLI

列出所有 SNOMED CT 推理作业

以下示例演示 `list-snomedct-inference-jobs` 操作如何返回当前异步 SNOMED CT 批量推理作业的列表。

```
aws comprehendmedical list-snomedct-inference-jobs
```

输出：

```
{
    "ComprehendMedicalAsyncJobPropertiesList": [
        {
            "JobId": "5780034166536cdb52ffa3295a1b00a7",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
            "EndTime": "2020-05-19T20:45:07.894000+00:00",
            "ExpirationTime": "2020-09-17T20:38:37+00:00",
            "InputDataConfig": {
                "S3Bucket": "comp-med-input",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "OutputDataConfig": {
                "S3Bucket": "comp-med-output",
                "S3Key": "AKIAIOSFODNN7EXAMPLE"
            },
            "LanguageCode": "en",
            "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
            "ModelVersion":  "0.1.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSnomedctInferenceJobs。

start-entities-detection-v2-job

以下代码示例演示了如何使用 start-entities-detection-v2-job。

Amazon CLI

启动实体检测作业

以下 start-entities-detection-v2-job 示例启动异步实体检测作业。

```
aws comprehendmedical start-entities-detection-v2-job \
    --input-data-config "S3Bucket=comp-med-input" \
    --output-data-config "S3Bucket=comp-med-output" \
    --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
    --language-code en
```

输出：

```
{
    "JobId": "ab9887877365fe70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartEntitiesDetectionV2Job。

start-icd10-cm-inference-job

以下代码示例演示了如何使用 start-icd10-cm-inference-job。

Amazon CLI

启动 ICD-10-CM 推理作业

以下 start-icd10-cm-inference-job 示例启动 ICD-10-CM 推理批量分析作业。

```
aws comprehendmedical start-icd10-cm-inference-job \
    --input-data-config "S3Bucket=comp-med-input" \
    --output-data-config "S3Bucket=comp-med-output" \
    --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
```

```
    --language-code en
```

输出：

```
{
    "JobId": "ef7289877365fc70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartIcd10CmInferenceJob。

start-phi-detection-job

以下代码示例演示了如何使用 start-phi-detection-job。

Amazon CLI

启动 PHI 检测作业

以下 start-phi-detection-job 示例启动异步 PHI 实体检测作业。

```
aws comprehendmedical start-phi-detection-job \
    --input-data-config "S3Bucket=comp-med-input" \
    --output-data-config "S3Bucket=comp-med-output" \
    --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
    --language-code en
```

输出：

```
{
    "JobId": "ab9887877365fe70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPhiDetectionJob。

start-rx-norm-inference-job

以下代码示例演示了如何使用 start-rx-norm-inference-job。

Amazon CLI

启动 RxNorm 推理作业

以下 `start-rx-norm-inference-job` 示例启动 RxNorm 推理批量分析作业。

```
aws comprehendmedical start-rx-norm-inference-job \
    --input-data-config "S3Bucket=comp-med-input" \
    --output-data-config "S3Bucket=comp-med-output" \
    --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
    --language-code en
```

输出：

```
{
    "JobId": "eg8199877365fc70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartRxNormInferenceJob。

start-snomedct-inference-job

以下代码示例演示了如何使用 `start-snomedct-inference-job`。

Amazon CLI

启动 SNOMED CT 推理作业

以下 `start-snomedct-inference-job` 示例启动 SNOMED CT 推理批量分析作业。

```
aws comprehendmedical start-snomedct-inference-job \
    --input-data-config "S3Bucket=comp-med-input" \
    --output-data-config "S3Bucket=comp-med-output" \
    --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
    --language-code en
```

输出：

```
{
    "JobId": "dg7289877365fc70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSnomedctInferenceJob。

stop-entities-detection-v2-job

以下代码示例演示了如何使用 stop-entities-detection-v2-job。

Amazon CLI

停止实体检测作业

以下 stop-entities-detection-v2-job 示例停止异步实体检测作业。

```
aws comprehendmedical stop-entities-detection-v2-job \
    --job-id "ab9887877365fe70299089371c043b96"
```

输出：

```
{
    "JobId": "ab9887877365fe70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopEntitiesDetectionV2Job。

stop-icd10-cm-inference-job

以下代码示例演示了如何使用 stop-icd10-cm-inference-job。

Amazon CLI

停止 ICD-10-CM 推理作业

以下 stop-icd10-cm-inference-job 示例停止 ICD-10-CM 推理批量分析作业。

```
aws comprehendmedical stop-icd10-cm-inference-job \
    --job-id "4750034166536cdb52ffa3295a1b00a3"
```

输出：

```
{
    "JobId": "ef7289877365fc70299089371c043b96",
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopIcd10CmInferenceJob。

stop-phi-detection-job

以下代码示例演示了如何使用 stop-phi-detection-job。

Amazon CLI

停止受保护健康信息（PHI）检测作业

以下 stop-phi-detection-job 示例停止异步受保护健康信息（PHI）检测作业。

```
aws comprehendmedical stop-phi-detection-job \
    --job-id "4750034166536cdb52ffa3295a1b00a3"
```

输出：

```
{
    "JobId": "ab9887877365fe70299089371c043b96"
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的批处理 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopPhiDetectionJob。

stop-rx-norm-inference-job

以下代码示例演示了如何使用 stop-rx-norm-inference-job。

Amazon CLI

停止 RxNorm 推理作业

以下 stop-rx-norm-inference-job 示例停止 ICD-10-CM 推理批量分析作业。

```
aws comprehendmedical stop-rx-norm-inference-job \
    --job-id "eg8199877365fc70299089371c043b96"
```

输出：

```
{
    "JobId": "eg8199877365fc70299089371c043b96",
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopRxNormInferenceJob。

stop-snomedct-inference-job

以下代码示例演示了如何使用 stop-snomedct-inference-job。

Amazon CLI

停止 SNOMED CT 推理作业

以下 stop-snomedct-inference-job 示例停止 SNOMED CT 推理批量分析作业。

```
aws comprehendmedical stop-snomedct-inference-job \
    --job-id "8750034166436cdb52ffa3295a1b00a1"
```

输出：

```
{
    "JobId": "8750034166436cdb52ffa3295a1b00a1",
}
```

有关更多信息，请参阅《Amazon Comprehend Medical 开发人员指南》中的本体链接批量分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopSnomedctInferenceJob。

Amazon Config使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Config 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-config-rule`

以下代码示例演示了如何使用 `delete-config-rule`。

Amazon CLI

删除 Amazon Config 规则

以下命令将删除名为 `MyConfigRule` 的 Amazon Config 规则：

```
aws configservice delete-config-rule --config-rule-name MyConfigRule
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConfigRule。

`delete-delivery-channel`

以下代码示例演示了如何使用 `delete-delivery-channel`。

Amazon CLI

删除传输通道

以下命令删除默认的传输通道：

```
aws configservice delete-delivery-channel --delivery-channel-name default
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeliveryChannel。

delete-evaluation-results

以下代码示例演示了如何使用 delete-evaluation-results。

Amazon CLI

手动删除评估结果

以下命令删除 Amazon 托管规则 s3-bucket-versioning-enabled 的当前评估结果：

```
aws configservice delete-evaluation-results --config-rule-name s3-bucket-versioning-enabled
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEvaluationResults。

deliver-config-snapshot

以下代码示例演示了如何使用 deliver-config-snapshot。

Amazon CLI

传输配置快照

以下命令将配置快照传输到属于默认传输通道的 Amazon S3 存储桶：

```
aws configservice deliver-config-snapshot --delivery-channel-name default
```

输出：

```
{
    "configSnapshotId": "d0333b00-a683-44af-921e-examplefb794"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeliverConfigSnapshot。

describe-compliance-by-config-rule

以下代码示例演示了如何使用 describe-compliance-by-config-rule。

Amazon CLI

获取您的 Amazon Config 规则的合规性信息

以下命令返回一个或多个 Amazon 资源违反的每个 Amazon Config 规则的合规性信息：

```
aws configservice describe-compliance-by-config-rule --compliance-
types NON_COMPLIANT
```

在输出中，每个 CappedCount 属性的值都表示有多少资源不符合相关规则。例如，以下输出表明 3 个资源不符合名为 InstanceTypesAreT2micro 的规则。

输出：

```
{
    "ComplianceByConfigRules": [
        {
            "Compliance": {
                "ComplianceContributorCount": {
                    "CappedCount": 3,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            },
            "ConfigRuleName": "InstanceTypesAreT2micro"
        },
        {
            "Compliance": {
                "ComplianceContributorCount": {
                    "CappedCount": 10,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            },
            "ConfigRuleName": "RequiredTagsForVolumes"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeComplianceByConfigRule。

describe-compliance-by-resource

以下代码示例演示了如何使用 describe-compliance-by-resource。

Amazon CLI

获取 Amazon 资源的合规性信息

以下命令返回由 Amazon Config 记录且违反一条或多条规则的每个 EC2 实例的合规性信息：

```
aws configservice describe-compliance-by-resource --resource-type AWS::EC2::Instance
 --compliance-types NON_COMPLIANT
```

在输出中，每个 CappedCount 属性的值都表示该资源违反了多少规则。例如，以下输出表明实例 i-1a2b3c4d 违反了 2 条规则。

输出：

```
{
    "ComplianceByResources": [
        {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-1a2b3c4d",
            "Compliance": {
                "ComplianceContributorCount": {
                    "CappedCount": 2,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            }
        },
        {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-2a2b3c4d ",
            "Compliance": {
                "ComplianceContributorCount": {
```

```
                    "CappedCount": 3,
                    "CapExceeded": false
                },
                "ComplianceType": "NON_COMPLIANT"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeComplianceByResource。

describe-config-rule-evaluation-status

以下代码示例演示了如何使用 describe-config-rule-evaluation-status。

Amazon CLI

获取 Amazon Config 规则的状态信息

以下命令将返回名为 MyConfigRule 的 Amazon Config 规则的状态信息：

```
aws configservice describe-config-rule-evaluation-status --config-rule-
names MyConfigRule
```

输出：

```
{
    "ConfigRulesEvaluationStatus": [
        {
            "ConfigRuleArn": "arn:aws:config:us-east-1:123456789012:config-rule/
config-rule-abcdef",
            "FirstActivatedTime": 1450311703.844,
            "ConfigRuleId": "config-rule-abcdef",
            "LastSuccessfulInvocationTime": 1450314643.156,
            "ConfigRuleName": "MyConfigRule"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigRuleEvaluationStatus。

describe-config-rules

以下代码示例演示了如何使用 describe-config-rules。

Amazon CLI

获取 Amazon Config 规则的详细信息

以下命令将返回名为 InstanceTypesAreT2micro 的 Amazon Config 规则的详细信息：

```
aws configservice describe-config-rules --config-rule-names InstanceTypesAreT2micro
```

输出：

```
{
    "ConfigRules": [
        {
            "ConfigRuleState": "ACTIVE",
            "Description": "Evaluates whether EC2 instances are the t2.micro type.",
            "ConfigRuleName": "InstanceTypesAreT2micro",
            "ConfigRuleArn": "arn:aws:config:us-east-1:123456789012:config-rule/
config-rule-abcdef",
            "Source": {
                "Owner": "CUSTOM_LAMBDA",
                "SourceIdentifier": "arn:aws:lambda:us-
east-1:123456789012:function:InstanceTypeCheck",
                "SourceDetails": [
                    {
                        "EventSource": "aws.config",
                        "MessageType": "ConfigurationItemChangeNotification"
                    }
                ]
            },
            "InputParameters": "{\"desiredInstanceType\":\"t2.micro\"}",
            "Scope": {
                "ComplianceResourceTypes": [
                    "AWS::EC2::Instance"
                ]
            },
            "ConfigRuleId": "config-rule-abcdef"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigRules。

describe-configuration-recorder-status

以下代码示例演示了如何使用 describe-configuration-recorder-status。

Amazon CLI

获取配置记录器的状态信息

以下命令返回默认配置记录器的状态。

```
aws configservice describe-configuration-recorder-status
```

输出：

```
{
    "ConfigurationRecordersStatus": [
        {
            "name": "default",
            "lastStatus": "SUCCESS",
            "recording": true,
            "lastStatusChangeTime": 1452193834.344,
            "lastStartTime": 1441039997.819,
            "lastStopTime": 1441039992.835
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigurationRecorderStatus。

describe-configuration-recorders

以下代码示例演示了如何使用 describe-configuration-recorders。

Amazon CLI

获取有关配置记录器的详细信息

以下命令返回有关默认配置记录器的详细信息：

```
aws configservice describe-configuration-recorders
```

输出：

```
{
    "ConfigurationRecorders": [
        {
            "recordingGroup": {
                "allSupported": true,
                "resourceTypes": [],
                "includeGlobalResourceTypes": true
            },
            "roleARN": "arn:aws:iam::123456789012:role/config-ConfigRole-
A1B2C3D4E5F6",
            "name": "default"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigurationRecorders。

describe-delivery-channel-status

以下代码示例演示了如何使用 describe-delivery-channel-status。

Amazon CLI

获取传输通道的状态信息

以下命令返回传输通道的状态：

```
aws configservice describe-delivery-channel-status
```

输出：

```
{
    "DeliveryChannelsStatus": [
        {
            "configStreamDeliveryInfo": {
                "lastStatusChangeTime": 1452193834.381,
                "lastStatus": "SUCCESS"
            },
```

```
            "configHistoryDeliveryInfo": {
                "lastSuccessfulTime": 1450317838.412,
                "lastStatus": "SUCCESS",
                "lastAttemptTime": 1450317838.412
            },
            "configSnapshotDeliveryInfo": {
                "lastSuccessfulTime": 1452185597.094,
                "lastStatus": "SUCCESS",
                "lastAttemptTime": 1452185597.094
            },
            "name": "default"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDeliveryChannelStatus。

describe-delivery-channels

以下代码示例演示了如何使用 describe-delivery-channels。

Amazon CLI

获取有关传输通道的详细信息

以下命令返回有关传输通道的详细信息：

```
aws configservice describe-delivery-channels
```

输出：

```
{
    "DeliveryChannels": [
        {
            "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
            "name": "default",
            "s3BucketName": "config-bucket-123456789012"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDeliveryChannels。

get-compliance-details-by-config-rule

以下代码示例演示了如何使用 get-compliance-details-by-config-rule。

Amazon CLI

获取 Amazon Config 规则的评估结果

以下命令返回所有不符合名为 InstanceTypesAreT2micro 的 Amazon Config 规则的资源的评估结果：

```
aws configservice get-compliance-details-by-config-rule --config-rule-
name InstanceTypesAreT2micro --compliance-types NON_COMPLIANT
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1450314635.065,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-1a2b3c4d",
                    "ConfigRuleName": "InstanceTypesAreT2micro"
                }
            },
            "ResultRecordedTime": 1450314645.261,
            "ConfigRuleInvokedTime": 1450314642.948,
            "ComplianceType": "NON_COMPLIANT"
        },
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1450314635.065,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-2a2b3c4d",
                    "ConfigRuleName": "InstanceTypesAreT2micro"
                }
            },
            "ResultRecordedTime": 1450314645.18,
            "ConfigRuleInvokedTime": 1450314642.902,
            "ComplianceType": "NON_COMPLIANT"
```

```
        },
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1450314635.065,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-3a2b3c4d",
                    "ConfigRuleName": "InstanceTypesAreT2micro"
                }
            },
            "ResultRecordedTime": 1450314643.346,
            "ConfigRuleInvokedTime": 1450314643.124,
            "ComplianceType": "NON_COMPLIANT"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComplianceDetailsByConfigRule。

get-compliance-details-by-resource

以下代码示例演示了如何使用 get-compliance-details-by-resource。

Amazon CLI

获取 Amazon 资源的评估结果

以下命令返回 EC2 实例 i-1a2b3c4d 不符合的每条规则的评估结果：

```
aws configservice get-compliance-details-by-resource --resource-
type AWS::EC2::Instance --resource-id i-1a2b3c4d --compliance-types NON_COMPLIANT
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1450314635.065,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
```

```
                    "ResourceId": "i-1a2b3c4d",
                    "ConfigRuleName": "InstanceTypesAreT2micro"
                }
            },
            "ResultRecordedTime": 1450314643.288,
            "ConfigRuleInvokedTime": 1450314643.034,
            "ComplianceType": "NON_COMPLIANT"
        },
        {
            "EvaluationResultIdentifier": {
                "OrderingTimestamp": 1450314635.065,
                "EvaluationResultQualifier": {
                    "ResourceType": "AWS::EC2::Instance",
                    "ResourceId": "i-1a2b3c4d",
                    "ConfigRuleName": "RequiredTagForEC2Instances"
                }
            },
            "ResultRecordedTime": 1450314645.261,
            "ConfigRuleInvokedTime": 1450314642.948,
            "ComplianceType": "NON_COMPLIANT"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComplianceDetailsByResource。

get-compliance-summary-by-config-rule

以下代码示例演示了如何使用 get-compliance-summary-by-config-rule。

Amazon CLI

获取您的 Amazon Config 规则的合规性摘要

以下命令返回合规和不合规的规则数量。

```
aws configservice get-compliance-summary-by-config-rule
```

在输出中，每个 CappedCount 属性的值都表示有多少规则合规或不合规。

输出：

```
{
    "ComplianceSummary": {
        "NonCompliantResourceCount": {
            "CappedCount": 3,
            "CapExceeded": false
        },
        "ComplianceSummaryTimestamp": 1452204131.493,
        "CompliantResourceCount": {
            "CappedCount": 2,
            "CapExceeded": false
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComplianceSummaryByConfigRule。

get-compliance-summary-by-resource-type

以下代码示例演示了如何使用 get-compliance-summary-by-resource-type。

Amazon CLI

获取所有资源类型的合规性摘要

以下命令返回合规和不合规的 Amazon 资源数量。

```
aws configservice get-compliance-summary-by-resource-type
```

在输出中，每个 CappedCount 属性的值都表示有多少资源合规或不合规。

输出：

```
{
    "ComplianceSummariesByResourceType": [
        {
            "ComplianceSummary": {
                "NonCompliantResourceCount": {
                    "CappedCount": 16,
                    "CapExceeded": false
                },
```

```
                "ComplianceSummaryTimestamp": 1453237464.543,
                "CompliantResourceCount": {
                    "CappedCount": 10,
                    "CapExceeded": false
                }
            }
        }
    ]
}
```

获取特定资源类型的合规性摘要

该命令返回合规和不合规的 EC2 实例数量。

```
aws configservice get-compliance-summary-by-resource-type --resource-
types AWS::EC2::Instance
```

在输出中，每个 CappedCount 属性的值都表示有多少资源合规或不合规。

输出：

```
{
    "ComplianceSummariesByResourceType": [
        {
            "ResourceType": "AWS::EC2::Instance",
            "ComplianceSummary": {
                "NonCompliantResourceCount": {
                    "CappedCount": 3,
                    "CapExceeded": false
                },
                "ComplianceSummaryTimestamp": 1452204923.518,
                "CompliantResourceCount": {
                    "CappedCount": 7,
                    "CapExceeded": false
                }
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComplianceSummaryByResourceType。

get-resource-config-history

以下代码示例演示了如何使用 get-resource-config-history。

Amazon CLI

获取 Amazon 资源的配置历史记录

以下命令返回 ID 为 i-1a2b3c4d 的 EC2 实例的配置项目列表：

```
aws configservice get-resource-config-history --resource-type AWS::EC2::Instance --
resource-id i-1a2b3c4d
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceConfigHistory。

get-status

以下代码示例演示了如何使用 get-status。

Amazon CLI

获取 Amazon Config 的状态

以下命令返回传输通道和配置记录器的状态。

```
aws configservice get-status
```

输出：

```
Configuration Recorders:

name: default
recorder: ON
last status: SUCCESS

Delivery Channels:

name: default
last stream delivery status: SUCCESS
last history delivery status: SUCCESS
last snapshot delivery status: SUCCESS
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStatus。

`list-discovered-resources`

以下代码示例演示了如何使用 `list-discovered-resources`。

Amazon CLI

列出 Amazon Config 已发现的资源

以下命令列出 Amazon Config 已发现的 EC2 实例：

```
aws configservice list-discovered-resources --resource-type AWS::EC2::Instance
```

输出：

```
{
    "resourceIdentifiers": [
        {
            "resourceType": "AWS::EC2::Instance",
            "resourceId": "i-1a2b3c4d"
        },
        {
            "resourceType": "AWS::EC2::Instance",
            "resourceId": "i-2a2b3c4d"
        },
        {
            "resourceType": "AWS::EC2::Instance",
            "resourceId": "i-3a2b3c4d"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDiscoveredResources。

`put-config-rule`

以下代码示例演示了如何使用 `put-config-rule`。

Amazon CLI

添加 Amazon 托管 Config 规则

以下命令提供用于添加 Amazon 托管 Config 规则的 JSON 代码：

```
aws configservice put-config-rule --config-rule file://
RequiredTagsForEC2Instances.json
```

RequiredTagsForEC2Instances.json 是一个包含规则配置的 JSON 文件：

```
{
  "ConfigRuleName": "RequiredTagsForEC2Instances",
  "Description": "Checks whether the CostCenter and Owner tags are applied to EC2
 instances.",
  "Scope": {
    "ComplianceResourceTypes": [
      "AWS::EC2::Instance"
    ]
  },
  "Source": {
    "Owner": "AWS",
    "SourceIdentifier": "REQUIRED_TAGS"
  },
  "InputParameters": "{\"tag1Key\":\"CostCenter\",\"tag2Key\":\"Owner\"}"
}
```

对于 ComplianceResourceTypes 属性，此 JSON 代码将范围限制为 AWS::EC2::Instance 类型的资源，因此，Amazon Config 将仅根据规则评估 EC2 实例。由于该规则是托管规则，因此 Owner 属性设置为，AWSSourceIdentifier 属性设置为规则标识符。REQUIRED_TAGS对于 InputParameters 属性，指定了规则所需的标签键 CostCenter 和 Owner。

如果命令成功，则 Amazon Config 不返回任何输出。要验证规则配置，请运行 describe-config-rules 命令并指定规则名称。

添加客户托管的 Config 规则

以下命令提供用于添加客户托管 Config 规则的 JSON 代码：

```
aws configservice put-config-rule --config-rule file://InstanceTypesAreT2micro.json
```

InstanceTypesAreT2micro.json 是一个包含规则配置的 JSON 文件：

```
{
  "ConfigRuleName": "InstanceTypesAreT2micro",
```

```
  "Description": "Evaluates whether EC2 instances are the t2.micro type.",
  "Scope": {
    "ComplianceResourceTypes": [
      "AWS::EC2::Instance"
    ]
  },
  "Source": {
    "Owner": "CUSTOM_LAMBDA",
    "SourceIdentifier": "arn:aws:lambda:us-
east-1:123456789012:function:InstanceTypeCheck",
    "SourceDetails": [
      {
        "EventSource": "aws.config",
        "MessageType": "ConfigurationItemChangeNotification"
      }
    ]
  },
  "InputParameters": "{\"desiredInstanceType\":\"t2.micro\"}"
}
```

对于 ComplianceResourceTypes 属性，此 JSON 代码将范围限制为 AWS::EC2::Instance 类型的资源，因此，Amazon Config 将仅根据规则评估 EC2 实例。由于此规则是客户托管规则，因此，Owner 属性设置为，CUSTOM_LAMBDASourceIdentifier 属性设置为 Amazon Lambda 函数的 ARN。SourceDetails 对象为必填项。当 Amazon Config 调用 Amazon Lambda 函数来根据规则评估资源时，为 InputParameters 属性指定的参数将传递给该函数。

如果命令成功，则 Amazon Config 不返回任何输出。要验证规则配置，请运行 describe-config-rules 命令并指定规则名称。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutConfigRule。

put-configuration-recorder

以下代码示例演示了如何使用 put-configuration-recorder。

Amazon CLI

示例 1：记录所有支持的资源

以下命令创建一个配置记录器，用于跟踪对所有支持的资源类型（包括全局资源类型）的更改：

```
aws configservice put-configuration-recorder \
```

```
    --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/
config-role \
    --recording-group allSupported=true,includeGlobalResourceTypes=true
```

如果命令成功，则 Amazon Config 不返回任何输出。要验证配置记录器的设置，请运行 describe-configuration-recorders 命令。

示例 2：记录指定资源的类型

以下命令创建一个配置记录器，该记录器仅跟踪在 JSON 文件中为 --recording-group 选项指定的资源类型的更改：

```
aws configservice put-configuration-recorder \
    --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/
config-role \
    --recording-group file://recordingGroup.json
```

recordingGroup.json 是一个 JSON 文件，它指定了 Amazon Config 将记录的资源类型：

```
{
    "allSupported": false,
    "includeGlobalResourceTypes": false,
    "resourceTypes": [
        "AWS::EC2::EIP",
        "AWS::EC2::Instance",
        "AWS::EC2::NetworkAcl",
        "AWS::EC2::SecurityGroup",
        "AWS::CloudTrail::Trail",
        "AWS::EC2::Volume",
        "AWS::EC2::VPC",
        "AWS::IAM::User",
        "AWS::IAM::Policy"
    ]
}
```

您必须将 allSupported 和 includeGlobalResourceTypes 选项设置为 false 或者忽略它们，才可以为 resourceTypes 键指定资源类型。

如果命令成功，则 Amazon Config 不返回任何输出。要验证配置记录器的设置，请运行 describe-configuration-recorders 命令。

示例 3：选择除指定资源类型之外的所有支持的资源

以下命令创建一个配置记录器，该记录器仅跟踪当前和未来支持的所有资源类型的更改，不包括在 JSON 文件中为 --recording-group 选项指定的资源类型：

```
aws configservice put-configuration-recorder \
    --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/config-role \
    --recording-group file://recordingGroup.json
```

recordingGroup.json 是一个 JSON 文件，它指定了 Amazon Config 将记录的资源类型：

```
{
    "allSupported": false,
    "exclusionByResourceTypes": {
        "resourceTypes": [
        "AWS::Redshift::ClusterSnapshot",
        "AWS::RDS::DBClusterSnapshot",
        "AWS::CloudFront::StreamingDistribution"
    ]
    },
        "includeGlobalResourceTypes": false,
        "recordingStrategy": {
        "useOnly": "EXCLUSION_BY_RESOURCE_TYPES"
    },
}
```

在指定不记录的资源类型之前：1) 必须将 allSupported 和 includeGlobalResourceTypes 选项设置为 false 或将其省略；2) 必须将 RecordingStrategy 的 useOnly 字段设置为 EXCLUSION_BY_RESOURCE_TYPES。

如果命令成功，则 Amazon Config 不返回任何输出。要验证配置记录器的设置，请运行 describe-configuration-recorders 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutConfigurationRecorder。

put-delivery-channel

以下代码示例演示了如何使用 put-delivery-channel。

Amazon CLI

创建传输通道

以下命令以 JSON 代码的形式提供传输通道的设置：

```
aws configservice put-delivery-channel --delivery-channel file://
deliveryChannel.json
```

deliveryChannel.json 文件指定了传输通道的属性：

```
{
    "name": "default",
    "s3BucketName": "config-bucket-123456789012",
    "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
    "configSnapshotDeliveryProperties": {
        "deliveryFrequency": "Twelve_Hours"
    }
}
```

此示例设置了以下属性：

name——传输通道的名称。默认情况下，Amazon Config 会将名称 default 分配给新的传输通道。您无法使用 put-delivery-channel 命令更新传输通道名称。有关更改名称的步骤，请参阅“重命名传输通道”。s3BucketName- Amazon Config 向其传输配置快照和配置历史记录文件的 Amazon S3 存储桶的名称。如果您指定的存储桶属于其他 Amazon 账户，则该存储桶必须拥有授予 Amazon Config 访问权限的策略。有关更多信息，请参阅 Amazon S3 存储桶的权限。

snsTopicARN - Amazon SNS 主题的 Amazon 资源名称（ARN），Amazon Config 会向其发送配置更改通知。如果您从其他账户选择主题，则该主题必须拥有授予 Amazon Config 访问权限的策略。有关更多信息，请参阅 Amazon SNS 主题的权限。

configSnapshotDeliveryProperties - 包含 deliveryFrequency 属性，该属性设置 Amazon Config 传输配置快照的频率以及它为定期 Config 规则调用评估的频率。

如果命令成功，则 Amazon Config 不返回任何输出。要验证您的传输通道的设置，请运行 describe-delivery-channels 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutDeliveryChannel。

start-config-rules-evaluation

以下代码示例演示了如何使用 start-config-rules-evaluation。

Amazon CLI

对 Amazon Config 规则运行按需评估

以下命令启动对两个 Amazon 托管规则的评估：

```
aws configservice start-config-rules-evaluation --config-rule-names s3-bucket-versioning-enabled cloudtrail-enabled
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartConfigRulesEvaluation。

start-configuration-recorder

以下代码示例演示了如何使用 start-configuration-recorder。

Amazon CLI

启动配置记录器

以下命令启动默认配置记录器：

```
aws configservice start-configuration-recorder --configuration-recorder-name default
```

如果命令成功，则 Amazon Config 不返回任何输出。要验证 Amazon Config 是否正在记录您的资源，请运行 get-status 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartConfigurationRecorder。

stop-configuration-recorder

以下代码示例演示了如何使用 stop-configuration-recorder。

Amazon CLI

停止配置记录器

以下命令停止默认配置记录器：

```
aws configservice stop-configuration-recorder --configuration-recorder-name default
```

如果命令成功，则 Amazon Config 不返回任何输出。要验证 Amazon Config 是否未在记录您的资源，请运行 get-status 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopConfigurationRecorder。

`subscribe`

以下代码示例演示了如何使用 `subscribe`。

Amazon CLI

订阅 Amazon Config

以下命令创建默认传输通道和配置记录器。该命令还指定 Amazon Config 将向其传输配置信息的 Amazon S3 存储桶和 Amazon SNS 主题：

```
aws configservice subscribe --s3-bucket config-bucket-123456789012
 --sns-topic arn:aws:sns:us-east-1:123456789012:config-topic --iam-
role arn:aws:iam::123456789012:role/ConfigRole-A1B2C3D4E5F6
```

输出：

```
Using existing S3 bucket: config-bucket-123456789012
Using existing SNS topic: arn:aws:sns:us-east-1:123456789012:config-topic
Subscribe succeeded:

Configuration Recorders: [
    {
        "recordingGroup": {
            "allSupported": true,
            "resourceTypes": [],
            "includeGlobalResourceTypes": false
        },
        "roleARN": "arn:aws:iam::123456789012:role/ConfigRole-A1B2C3D4E5F6",
        "name": "default"
    }
]

Delivery Channels: [
    {
        "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
        "name": "default",
        "s3BucketName": "config-bucket-123456789012"
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 Subscribe。

使用 Amazon CLI 的 Amazon Connect 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Connect 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-user`

以下代码示例演示了如何使用 `create-user`。

Amazon CLI

创建用户

以下 `create-user` 示例向指定 Amazon Connect 实例添加带有指定属性的用户。

```
aws connect create-user \
    --username Mary \
    --password Pass@Word1 \
    --identity-info FirstName=Mary,LastName=Major \
    --phone-
config PhoneType=DESK_PHONE,AutoAccept=true,AfterContactWorkTimeLimit=60,DeskPhoneNumber=
+15555551212 \
    --security-profile-id 12345678-1111-2222-aaaa-a1b2c3d4f5g7 \
    --routing-profile-id 87654321-9999-3434-abcd-x1y2z3a1b2c3 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "UserId": "87654321-2222-1234-1234-111234567891",
    "UserArn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111/agent/87654321-2222-1234-1234-111234567891"
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的添加用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

以下 delete-user 示例从指定的 Amazon Connect 实例中删除指定用户。

```
aws connect delete-user \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --user-id 87654321-2222-1234-1234-111234567891
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的管理用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-user-hierarchy-group

以下代码示例演示了如何使用 describe-user-hierarchy-group。

Amazon CLI

显示层次结构组的详细信息

以下 describe-user-hierarchy-group 示例显示指定 Amazon Connect 层次结构组的详细信息。

```
aws connect describe-user-hierarchy-group \
```

```
    --hierarchy-group-id 12345678-1111-2222-800e-aaabbb555gg \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
     "HierarchyGroup": {
         "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
         "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111/agent-group/12345678-1111-2222-800e-a2b3c4d5f6g7",
         "Name": "Example Corporation",
         "LevelId": "1",
         "HierarchyPath": {
             "LevelOne": {
                 "Id": "abcdefgh-3333-4444-8af3-201123456789",
                 "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group/
abcdefgh-3333-4444-8af3-201123456789",
                 "Name": "Example Corporation"
             }
         }
     }
 }
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的设置座席层次结构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserHierarchyGroup。

describe-user-hierarchy-structure

以下代码示例演示了如何使用 describe-user-hierarchy-structure。

Amazon CLI

显示层次结构的详细信息

以下 describe-user-hierarchy-structure 示例显示指定 Amazon Connect 实例层次结构的详细信息。

```
aws connect describe-user-hierarchy-group \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
     "HierarchyStructure": {
         "LevelOne": {
             "Id": "12345678-1111-2222-800e-aaabbb555gg",
             "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/1",
             "Name": "Corporation"
         },
         "LevelTwo": {
             "Id": "87654321-2222-3333-ac99-123456789102",
             "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/2",
             "Name": "Services Division"
         },
         "LevelThree": {
             "Id": "abcdefgh-3333-4444-8af3-201123456789",
             "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/3",
             "Name": "EU Site"
         }
     }
 }
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的设置座席层次结构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserHierarchyStructure。

describe-user

以下代码示例演示了如何使用 describe-user。

Amazon CLI

显示用户的详细信息

以下 describe-user 示例显示指定 Amazon Connect 用户的详细信息。

```
aws connect describe-user \
    --user-id 0c245dc0-0cf5-4e37-800e-2a7481cc8a60
    --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

输出：

```
{
    "User": {
        "Id": "0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
        "Username": "Jane",
        "IdentityInfo": {
            "FirstName": "Jane",
            "LastName": "Doe",
            "Email": "example.com"
        },
        "PhoneConfig": {
            "PhoneType": "SOFT_PHONE",
            "AutoAccept": false,
            "AfterContactWorkTimeLimit": 0,
            "DeskPhoneNumber": ""
        },
        "DirectoryUserId": "8b444cf6-b368-4f29-ba18-07af27405658",
        "SecurityProfileIds": [
            "b6f85a42-1dc5-443b-b621-de0abf70c9cf"
        ],
        "RoutingProfileId": "0be36ee9-2b5f-4ef4-bcf7-87738e5be0e5",
        "Tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的管理用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUser。

get-contact-attributes

以下代码示例演示了如何使用 get-contact-attributes。

Amazon CLI

检索联系人的属性

以下 get-contact-attributes 示例检索为指定的 Amazon Connect 联系人设置的属性。

```
aws connect get-contact-attributes \
```

```
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --initial-contact-id 12345678-1111-2222-800e-a2b3c4d5f6g7
```

输出：

```
{
    "Attributes": {
        "greetingPlayed": "true"
    }
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的使用 Amazon Connect 联系人属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContactAttributes。

list-contact-flows

以下代码示例演示了如何使用 list-contact-flows。

Amazon CLI

列出实例中的联系流

以下 list-contact-flows 示例列出指定 Amazon Connect 实例中的联系流。

```
aws connect list-contact-flows \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
     "ContactFlowSummaryList": [
         {
             "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
             "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/12345678-1111-2222-800e-
a2b3c4d5f6g7",
             "Name": "Default queue transfer",
             "ContactFlowType": "QUEUE_TRANSFER"
         },
         {
             "Id": "87654321-2222-3333-ac99-123456789102",
```

```
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/87654321-2222-3333-
ac99-123456789102",
            "Name": "Default agent hold",
            "ContactFlowType": "AGENT_HOLD"
        },
        {
            "Id": "abcdefgh-3333-4444-8af3-201123456789",
                "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/
abcdefgh-3333-4444-8af3-201123456789",
            "Name": "Default customer hold",
            "ContactFlowType": "CUSTOMER_HOLD"
        },
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的创建 Amazon Connect 联系流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContactFlows。

`list-hours-of-operations`

以下代码示例演示了如何使用 `list-hours-of-operations`。

Amazon CLI

列出实例的运行时间

以下 `list-hours-of-operations` 示例列出指定 Amazon Connect 实例的运行时间。

```
aws connect list-hours-of-operations \
    --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

输出：

```
{
    "HoursOfOperationSummaryList": [
        {
            "Id": "d69f1f84-7457-4924-8fbe-e64875546259",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/operating-hours/d69f1f84-7457-4924-8fbe-e64875546259",
            "Name": "Basic Hours"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的设置队列的运行时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHoursOfOperations。

`list-phone-numbers`

以下代码示例演示了如何使用 `list-phone-numbers`。

Amazon CLI

列出实例中的电话号码

以下 `list-phone-numbers` 示例列出指定 Amazon Connect 实例中的电话号码。

```
aws connect list-phone-numbers \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "PhoneNumberSummaryList": [
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/phone-number/xyz80zxy-xyz1-80zx-
zx80-11111EXAMPLE",
            "PhoneNumber": "+17065551212",
            "PhoneNumberType": "DID",
            "PhoneNumberCountryCode": "US"
        },
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/phone-number/ccc0ccc-xyz1-80zx-
zx80-22222EXAMPLE",
            "PhoneNumber": "+18555551212",
            "PhoneNumberType": "TOLL_FREE",
            "PhoneNumberCountryCode": "US"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的设置联络中心的电话号码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPhoneNumbers。

`list-queues`

以下代码示例演示了如何使用 `list-queues`。

Amazon CLI

列出实例中的队列

以下 `list-queues` 示例列出指定 Amazon Connect 实例中的队列。

```
aws connect list-queues \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "QueueSummaryList": [
        {
            "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/12345678-1111-2222-800e-
a2b3c4d5f6g7",
            "QueueType": "AGENT"
        },
        {
            "Id": "87654321-2222-3333-ac99-123456789102",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/87654321-2222-3333-
ac99-123456789102",
            "QueueType": "AGENT"
        },
        {
            "Id": "abcdefgh-3333-4444-8af3-201123456789",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/
abcdefgh-3333-4444-8af3-201123456789",
```

```
            "QueueType": "AGENT"
        },
        {
            "Id": "hgfedcba-4444-5555-a31f-123456789102",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/hgfedcba-4444-5555-a31f-123456789102",
            "Name": "BasicQueue",
            "QueueType": "STANDARD"
        },
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的创建队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueues。

`list-routing-profiles`

以下代码示例演示了如何使用 `list-routing-profiles`。

Amazon CLI

列出实例中的路由配置文件

以下 `list-routing-profiles` 示例列出指定 Amazon Connect 实例中的路由配置文件。

```
aws connect list-routing-profiles \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RoutingProfileSummaryList": [
        {
            "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/routing-profile/12345678-1111-2222-800e-
a2b3c4d5f6g7",
            "Name": "Basic Routing Profile"
        },
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的创建路由配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoutingProfiles。

list-security-profiles

以下代码示例演示了如何使用 list-security-profiles。

Amazon CLI

列出实例中的安全配置文件

以下 list-security-profiles 示例列出指定 Amazon Connect 实例中的安全配置文件。

```
aws connect list-security-profiles \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "SecurityProfileSummaryList": [
        {
            "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/12345678-1111-2222-800e-
a2b3c4d5f6g7",
            "Name": "CallCenterManager"
        },
        {
            "Id": "87654321-2222-3333-ac99-123456789102",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/87654321-2222-3333-
ac99-123456789102",
            "Name": "QualityAnalyst"
        },
        {
            "Id": "abcdefgh-3333-4444-8af3-201123456789",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/
abcdefgh-3333-4444-8af3-201123456789",
            "Name": "Agent"
        },
        {
```

```
            "Id": "12345678-1111-2222-800e-x2y3c4d5fzzzz",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/12345678-1111-2222-800e-
x2y3c4d5fzzzz",
            "Name": "Admin"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的分配权限：安全配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecurityProfiles。

`list-user-hierarchy-groups`

以下代码示例演示了如何使用 `list-user-hierarchy-groups`。

Amazon CLI

列出实例中的用户层次结构组

以下 `list-user-hierarchy-groups` 示例列出指定 Amazon Connect 实例中的用户层次结构组。

```
aws connect list-user-hierarchy-groups \
    --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

输出：

```
{
    "UserHierarchyGroupSummaryList": [
        {
            "Id": "0e2f6d1d-b3ca-494b-8dbc-ba81d9f8182a",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent-group/0e2f6d1d-b3ca-494b-8dbc-ba81d9f8182a",
            "Name": "Example Corporation"
        },
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的设置座席层次结构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUserHierarchyGroups。

list-users

以下代码示例演示了如何使用 list-users。

Amazon CLI

列出实例中的用户层次结构组

以下 list-users 示例列出指定 Amazon Connect 实例中的用户。

```
aws connect list-users \
    --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

输出：

```
{
    "UserSummaryList": [
        {
            "Id": "0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
            "Username": "Jane"
        },
        {
            "Id": "46f0c67c-3fc7-4806-ac99-403798788c14",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/46f0c67c-3fc7-4806-ac99-403798788c14",
            "Username": "Paulo"
        },
        {
            "Id": "55a83578-95e1-4710-8af3-2b7afe310e48",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/55a83578-95e1-4710-8af3-2b7afe310e48",
            "Username": "JohnD"
        },
        {
            "Id": "703e27b5-c9f0-4f1f-a239-64ccbb160125",
            "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/703e27b5-c9f0-4f1f-a239-64ccbb160125",
            "Username": "JohnS"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Connect 管理员指南》中的添加用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

update-contact-attributes

以下代码示例演示了如何使用 update-contact-attributes。

Amazon CLI

更新联系人的属性

以下 update-contact-attributes 示例更新指定 Amazon Connect 用户的 greetingPlayed 属性。

```
aws connect update-contact-attributes \
    --initial-contact-id 11111111-2222-3333-4444-12345678910 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --attributes greetingPlayed=false
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的使用 Amazon Connect 联系人属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContactAttributes。

update-user-hierarchy

以下代码示例演示了如何使用 update-user-hierarchy。

Amazon CLI

更新用户的层次结构

以下 update-user-hierarchy 示例更新指定 Amazon Connect 用户的座席层次结构。

```
aws connect update-user-hierarchy \
    --hierarchy-group-id 12345678-a1b2-c3d4-e5f6-123456789abc \
    --user-id 87654321-2222-1234-1234-111234567891 \
```

```
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的配置座席设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserHierarchy。

update-user-identity-info

以下代码示例演示了如何使用 update-user-identity-info。

Amazon CLI

更新用户的身份信息

以下 update-user-identity-info 示例更新指定 Amazon Connect 用户的身份信息。

```
aws connect update-user-identity-info \
    --identity-info FirstName=Mary,LastName=Major,Email=marym@example.com \
    --user-id 87654321-2222-1234-1234-111234567891 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的配置座席设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserIdentityInfo。

update-user-phone-config

以下代码示例演示了如何使用 update-user-phone-config。

Amazon CLI

更新用户的电话配置

以下 update-user-phone-config 示例更新指定用户的电话配置。

```
aws connect update-user-phone-config \
    --phone-
config PhoneType=SOFT_PHONE,AutoAccept=false,AfterContactWorkTimeLimit=60,DeskPhoneNumber=
+18005551212 \
```

```
    --user-id 12345678-4444-3333-2222-111122223333 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的配置座席设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserPhoneConfig。

update-user-routing-profile

以下代码示例演示了如何使用 update-user-routing-profile。

Amazon CLI

更新用户的路由配置文件

以下 update-user-routing-profile 示例更新指定 Amazon Connect 用户的路由配置文件。

```
aws connect update-user-routing-profile \
    --routing-profile-id 12345678-1111-3333-2222-4444EXAMPLE \
    --user-id 87654321-2222-1234-1234-111234567891 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的配置座席设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserRoutingProfile。

update-user-security-profiles

以下代码示例演示了如何使用 update-user-security-profiles。

Amazon CLI

更新用户的安全配置文件

以下 update-user-security-profiles 示例更新指定 Amazon Connect 用户的安全配置文件。

```
aws connect update-user-security-profiles \
```

```
    --security-profile-ids 12345678-1234-1234-1234-1234567892111 \
    --user-id 87654321-2222-1234-1234-111234567891 \
    --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Connect 管理员指南》中的分配权限：安全配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUserSecurityProfiles。

Amazon 成本和使用情况报告使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon 成本和使用情况报告 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-report-definition`

以下代码示例演示了如何使用 `delete-report-definition`。

Amazon CLI

删除 Amazon 成本和使用情况报告

此示例删除 Amazon 成本和使用情况报告。

命令:

```
aws cur --region us-east-1 delete-report-definition --report-name "ExampleReport"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReportDefinition。

describe-report-definitions

以下代码示例演示了如何使用 describe-report-definitions。

Amazon CLI

检索 Amazon 成本和使用情况报告列表

此示例描述账户拥有的 Amazon 成本和使用情况报告列表。

命令:

```
aws cur --region us-east-1 describe-report-definitions --max-items 5
```

输出：

```
  {
"ReportDefinitions": [
  {
      "ReportName": "ExampleReport",
      "Compression": "ZIP",
      "S3Region": "us-east-1",
      "Format": "textORcsv",
      "S3Prefix": "exampleprefix",
      "S3Bucket": "example-s3-bucket",
      "TimeUnit": "DAILY",
      "AdditionalArtifacts": [
          "REDSHIFT",
          "QUICKSIGHT"
      ],
      "AdditionalSchemaElements": [
          "RESOURCES"
      ]
  }
]
  }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReportDefinitions。

put-report-definition

以下代码示例演示了如何使用 put-report-definition。

Amazon CLI

创建 Amazon 成本和使用情况报告

以下 put-report-definition 示例创建每日 Amazon 成本和使用情况报告，您可以将其上传到 Amazon Redshift 或 Amazon QuickSight。

```
aws cur put-report-definition --report-definition file://report-definition.json
```

的内容report-definition.json：

```
{
    "ReportName": "ExampleReport",
    "TimeUnit": "DAILY",
    "Format": "textORcsv",
    "Compression": "ZIP",
    "AdditionalSchemaElements": [
        "RESOURCES"
    ],
    "S3Bucket": "example-s3-bucket",
    "S3Prefix": "exampleprefix",
    "S3Region": "us-east-1",
    "AdditionalArtifacts": [
        "REDSHIFT",
        "QUICKSIGHT"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutReportDefinition。

使用 Amazon CLI 的 Cost Explorer Service 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Cost Explorer Service 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

get-cost-and-usage

以下代码示例演示了如何使用 get-cost-and-usage。

Amazon CLI

检索账户在 2017 年 9 月的 S3 使用情况

以下 get-cost-and-usage 示例检索账户在 2017 年 9 月的 S3 使用情况。

```
aws ce get-cost-and-usage \
    --time-period Start=2017-09-01,End=2017-10-01 \
    --granularity MONTHLY \
    --metrics "BlendedCost" "UnblendedCost" "UsageQuantity" \
    --group-by Type=DIMENSION,Key=SERVICE Type=TAG,Key=Environment \
    --filter file://filters.json
```

的内容filters.json：

```
{
    "Dimensions": {
        "Key": "SERVICE",
        "Values": [
            "Amazon Simple Storage Service"
        ]
    }
}
```

输出：

```
{
    "GroupDefinitions": [
        {
            "Type": "DIMENSION",
            "Key": "SERVICE"
        },
        {
            "Type": "TAG",
```

```
            "Key": "Environment"
        }
    ],
    "ResultsByTime": [
        {
            "Estimated": false,
            "TimePeriod": {
                "Start": "2017-09-01",
                "End": "2017-10-01"
            },
            "Total": {},
            "Groups": [
                {
                    "Keys": [
                        "Amazon Simple Storage Service",
                        "Environment$"
                    ],
                    "Metrics": {
                        "BlendedCost": {
                            "Amount": "40.3527508453",
                            "Unit": "USD"
                        },
                        "UnblendedCost": {
                            "Amount": "40.3543773134",
                            "Unit": "USD"
                        },
                        "UsageQuantity": {
                            "Amount": "9312771.098461578",
                            "Unit": "N/A"
                        }
                    }
                },
                {
                    "Keys": [
                        "Amazon Simple Storage Service",
                        "Environment$Dev"
                    ],
                    "Metrics": {
                        "BlendedCost": {
                            "Amount": "0.2682364644",
                            "Unit": "USD"
                        },
                        "UnblendedCost": {
                            "Amount": "0.2682364644",
```

```
                        "Unit": "USD"
                    },
                    "UsageQuantity": {
                        "Amount": "22403.4395271182",
                        "Unit": "N/A"
                    }
                }
            }
        ]
    }
]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCostAndUsage。

get-dimension-values

以下代码示例演示了如何使用 get-dimension-values。

Amazon CLI

检索值为“Elastic”的维度 SERVICE 的标签

此示例检索 2017 年 1 月 1 日至 2017 年 5 月 18 日的值为“Elastic”的维度 SERVICE 标签。

命令:

```
aws ce get-dimension-values --search-string Elastic --time-
period Start=2017-01-01,End=2017-05-18 --dimension SERVICE
```

输出：

```
{
   "TotalSize": 6,
   "DimensionValues": [
       {
           "Attributes": {},
           "Value": "Amazon ElastiCache"
       },
       {
           "Attributes": {},
           "Value": "EC2 - Other"
```

```
        },
        {
            "Attributes": {},
            "Value": "Amazon Elastic Compute Cloud - Compute"
        },
        {
            "Attributes": {},
            "Value": "Amazon Elastic Load Balancing"
        },
        {
            "Attributes": {},
            "Value": "Amazon Elastic MapReduce"
        },
        {
            "Attributes": {},
            "Value": "Amazon Elasticsearch Service"
        }
    ],
    "ReturnSize": 6
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDimensionValues。

get-reservation-coverage

以下代码示例演示了如何使用 get-reservation-coverage。

Amazon CLI

检索 us-east-1 区域中 EC2 t2.nano 实例的预留覆盖范围

此示例检索 2017 年 7 月至 9 月 us-east-1 区域中 EC2 t2.nano 实例的预留覆盖范围。

命令:

```
aws ce get-reservation-coverage --time-period Start=2017-07-01,End=2017-10-01 --
group-by Type=Dimension,Key=REGION --filter file://filters.json
```

filters.json：

```
{
    "And": [
```

```
    {
      "Dimensions": {
        "Key": "INSTANCE_TYPE",
        "Values": [
          "t2.nano"
        ]
      },
      "Dimensions": {
        "Key": "REGION",
        "Values": [
          "us-east-1"
        ]
      }
    }
  ]
}
```

输出：

```
{
   "TotalSize": 6,
   "DimensionValues": [
       {
           "Attributes": {},
           "Value": "Amazon ElastiCache"
       },
       {
           "Attributes": {},
           "Value": "EC2 - Other"
       },
       {
           "Attributes": {},
           "Value": "Amazon Elastic Compute Cloud - Compute"
       },
       {
           "Attributes": {},
           "Value": "Amazon Elastic Load Balancing"
       },
       {
           "Attributes": {},
           "Value": "Amazon Elastic MapReduce"
       },
       {
```

```
            "Attributes": {},
            "Value": "Amazon Elasticsearch Service"
        }
    ],
    "ReturnSize": 6
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReservationCoverage。

get-reservation-purchase-recommendation

以下代码示例演示了如何使用 get-reservation-purchase-recommendation。

Amazon CLI

检索为期三年的部分预付 EC2 RI 的预留建议

以下 get-reservation-purchase-recommendation 示例根据过去 60 天的 EC2 使用情况，检索为期三年的部分预付 EC2 实例的建议。

```
aws ce get-reservation-purchase-recommendation \
    --service "Amazon Redshift" \
    --lookback-period-in-days SIXTY_DAYS \
    --term-in-years THREE_YEARS \
    --payment-option PARTIAL_UPFRONT
```

输出：

```
{
    "Recommendations": [],
    "Metadata": {
        "GenerationTimestamp": "2018-08-08T15:20:57Z",
        "RecommendationId": "00d59dde-a1ad-473f-8ff2-iexample3330b"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReservationPurchaseRecommendation。

get-reservation-utilization

以下代码示例演示了如何使用 get-reservation-utilization。

Amazon CLI

检索您账户的预留利用率

以下 get-reservation-utilization 示例检索账户在 2018-03-01 到 2018-08-01 期间所有 t2.nano 实例类型的 RI 利用率。

```
aws ce get-reservation-utilization \
    --time-period Start=2018-03-01,End=2018-08-01 \
    --filter file://filters.json
```

的内容filters.json：

```
{
    "Dimensions": {
        "Key": "INSTANCE_TYPE",
        "Values": [
            "t2.nano"
        ]
    }
}
```

输出：

```
{
    "Total": {
        "TotalAmortizedFee": "0",
        "UtilizationPercentage": "0",
        "PurchasedHours": "0",
        "NetRISavings": "0",
        "TotalActualHours": "0",
        "AmortizedRecurringFee": "0",
        "UnusedHours": "0",
        "TotalPotentialRISavings": "0",
        "OnDemandCostOfRIHoursUsed": "0",
        "AmortizedUpfrontFee": "0"
    },
    "UtilizationsByTime": []
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReservationUtilization。

get-tags

以下代码示例演示了如何使用 get-tags。

Amazon CLI

检索成本分配标签的键和值

此示例检索密钥为“Project”且值包含“secretProject”的所有成本分配标签。

命令:

```
aws ce get-tags --search-string secretProject --time-
period Start=2017-01-01,End=2017-05-18 --tag-key Project
```

输出：

```
{
  "ReturnSize": 2,
  "Tags": [
    "secretProject1",
    "secretProject2"
  ],
  "TotalSize": 2
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTags。

使用 Amazon CLI 的 Firehose 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Firehose 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`list-delivery-streams`

以下代码示例演示了如何使用 `list-delivery-streams`。

Amazon CLI

列出可用的传输流

以下 `list-delivery-streams` 示例列出了您 Amazon 账户中的可用传输流。

```
aws firehose list-delivery-streams
```

输出：

```
{
    "DeliveryStreamNames": [
        "my-stream"
    ],
    "HasMoreDeliveryStreams": false
}
```

有关更多信息，请参阅 Amazon Kinesis Data Firehose 开发人员指南中的创建 Amazon Kinesis Data Firehose 传输流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeliveryStreams。

`put-record-batch`

以下代码示例演示了如何使用 `put-record-batch`。

Amazon CLI

将多条记录写入流中

以下 put-record-batch 示例将三条记录写入流中。数据以 Base64 格式编码。

```
aws firehose put-record-batch \
    --delivery-stream-name my-stream \
    --records file://records.json
```

的内容myfile.json：

```
[
    {"Data": "Rmlyc3QgdGhpbmc="},
    {"Data": "U2Vjb25kIHRoaW5n"},
    {"Data": "VGhpcmQgdGhpbmc="}
]
```

输出：

```
{
    "FailedPutCount": 0,
    "Encrypted": false,
    "RequestResponses": [
        {
            "RecordId": "9D2OJ6t2EqCTZTXwGzeSv/EVHxRoRCw89xd+o3+sXg8DhYOaWKPSmZy/
CGlRVEys1u1xbeKh6VofEYKkoeiDrcjrxhQp9iF7sUW7pujiMEQ5LzlrzCkGosxQn
+3boDnURDEaD42V7GiixpOyLJkYZcae1i7HzlCEoy9LJhMr8EjDSi4Om/9Vc2uhwwuAtGE0XKpxJ2WD7ZRWtAnYlKAnv
        },
        {
            "RecordId": "jFirejqxCLlK5xjH/UNmlMVcjktEN76I7916X9PaZ
+PVaOSXDfU1WGOqEZhxq2js7xcZ552eoeDxsuTU1MSq9nZTbVfb6cQTIXnm/GsuF37Uhg67GKmR5z90l6XKJ
+/+pDloFv7Hh9a3oUS6wYm3DcNRLTHHAimANp1PhkQvWpvLRfzbuCUkBphR2QVzhP9OiHLbzGwy8/
DfH8sqWEUYASNJKS8GXP5s"
        },
        {
            "RecordId":
 "oy0amQ40o5Y2YV4vxzufdcMOOw6n3EPr3tpPJGoYVNKH4APPVqNcbUgefo1stEFRg4hTLrf2k6eliHu/9+YJ5R3iie
DTBt3qBlmTj7Xq8SKVbO1S7YvMTpWkMKA86f8JfmT8BMKoMb4XZS/sOkQLe+qh0sYKXWl"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Kinesis Data Firehose 开发人员指南》中的将数据发送到 Amazon Kinesis Data Firehose 传输流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRecordBatch。

put-record

以下代码示例演示了如何使用 put-record。

Amazon CLI

将记录写入流

以下 put-record 示例将数据写入流中。数据以 Base64 格式编码。

```
aws firehose put-record \
    --delivery-stream-name my-stream \
    --record '{"Data":"SGVsbG8gd29ybGQ="}'
```

输出：

```
{
    "RecordId": "RjB5K/nnoGFHqwTsZlNd/
TTqvjE8V5dsyXZTQn2JXrdpMTOwssyEb6nfC8fwf1whhwnItt4mvrn+gsqeK5jB7QjuLg283+Ps4Sz/
j1Xujv31iDhnPdaLw4BOyM9Amv7PcCuB2079RuM0NhoakbyUymlwY8yt20G8X2420wu1jlFafhci4erAt7QhDEvpwuK8
    "Encrypted": false
}
```

有关更多信息，请参阅《Amazon Kinesis Data Firehose 开发人员指南》中的将数据发送到 Amazon Kinesis Data Firehose 传输流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRecord。

使用 Amazon CLI 的 Amazon Data Lifecycle Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Data Lifecycle Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-default-role`

以下代码示例演示了如何使用 `create-default-role`。

Amazon CLI

为 Amazon DLM 创建所需的 IAM 角色

以下 `dlm create-default-role` 示例创建用于管理快照的 AmazonDataLifecycleManagerDefaultRole 默认角色。

```
aws dlm create-default-role \
    --resource-type snapshot
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的 Amazon Data Lifecycle Manager 的默认服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDefaultRole。

`create-lifecycle-policy`

以下代码示例演示了如何使用 `create-lifecycle-policy`。

Amazon CLI

创建生命周期策略

以下 `create-lifecycle-policy` 示例创建一个生命周期策略，在指定时间创建卷的每日快照。将指定的标签添加到快照中，标签也将从卷中复制并添加到快照中。如果创建的新快照超过指定的最大计数，则将删除最早的快照。

```
aws dlm create-lifecycle-policy \
    --description "My first policy" \
    --state ENABLED \
```

```
    --execution-role-arn arn:aws:iam::12345678910:role/
AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

的内容policyDetails.json：

```
{
    "ResourceTypes": [
        "VOLUME"
    ],
    "TargetTags": [
        {
            "Key": "costCenter",
            "Value": "115"
        }
    ],
    "Schedules":[
        {
            "Name": "DailySnapshots",
            "CopyTags": true,
            "TagsToAdd": [
                {
                    "Key": "type",
                     "Value": "myDailySnapshot"
                }
            ],
            "CreateRule": {
                "Interval": 24,
                "IntervalUnit": "HOURS",
                "Times": [
                    "03:00"
                ]
            },
            "RetainRule": {
                "Count":5
            }
        }
    ]
}
```

输出：

```
{
```

```
    "PolicyId": "policy-0123456789abcdef0"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLifecyclePolicy。

delete-lifecycle-policy

以下代码示例演示了如何使用 delete-lifecycle-policy。

Amazon CLI

删除生命周期策略

以下示例删除指定的生命周期策略：

```
aws dlm delete-lifecycle-policy --policy-id policy-0123456789abcdef0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLifecyclePolicy。

get-lifecycle-policies

以下代码示例演示了如何使用 get-lifecycle-policies。

Amazon CLI

获取生命周期策略的摘要

以下 get-lifecycle-policies 示例列出所有生命周期策略。

```
aws dlm get-lifecycle-policies
```

输出：

```
{
    "Policies": [
        {
            "PolicyId": "policy-0123456789abcdef0",
            "Description": "My first policy",
            "State": "ENABLED"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLifecyclePolicies。

get-lifecycle-policy

以下代码示例演示了如何使用 get-lifecycle-policy。

Amazon CLI

描述生命周期策略

以下 get-lifecycle-policy 示例显示指定生命周期策略的详细信息。

```
aws dlm get-lifecycle-policy \
    --policy-id policy-0123456789abcdef0
```

输出：

```
{
    "Policy": {
        "PolicyId": "policy-0123456789abcdef0",
        "Description": "My policy",
        "State": "ENABLED",
        "ExecutionRoleArn": "arn:aws:iam::123456789012:role/
AWSDataLifecycleManagerDefaultRole",
        "DateCreated": "2019-08-08T17:45:42Z",
        "DateModified": "2019-08-08T17:45:42Z",
        "PolicyDetails": {
            "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
            "ResourceTypes": [
                "VOLUME"
            ],
            "TargetTags": [
              {
                  "Key": "costCenter",
                  "Value": "115"
              }
            ],
            "Schedules": [
              {
                  "Name": "DailySnapshots",
```

```
                "CopyTags": true,
                "TagsToAdd": [
                  {
                      "Key": "type",
                      "Value": "myDailySnapshot"
                  }
                ],
                "CreateRule": {
                  "Interval": 24,
                  "IntervalUnit": "HOURS",
                  "Times": [
                      "03:00"
                  ]
                },
                "RetainRule": {
                  "Count": 5
                }
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLifecyclePolicy。

update-lifecycle-policy

以下代码示例演示了如何使用 update-lifecycle-policy。

Amazon CLI

示例 1：启用生命周期策略

以下 update-lifecycle-policy 示例启用指定的生命周期策略。

```
aws dlm update-lifecycle-policy \
    --policy-id policy-0123456789abcdef0 \
    --state ENABLED
```

示例 2：禁用生命周期策略

以下 update-lifecycle-policy 示例禁用指定的生命周期策略。

```
aws dlm update-lifecycle-policy \
    --policy-id policy-0123456789abcdef0 \
    --state DISABLED
```

示例 3：更新生命周期策略的详细信息

以下 update-lifecycle-policy 示例更新指定生命周期策略的目标标签。

```
aws dlm update-lifecycle-policy \
    --policy-id policy-0123456789abcdef0
    --policy-details file://policyDetails.json
```

policyDetails.json 的内容。该命令不会更改此文件中未引用的其他详细信息。

```
{
    "TargetTags": [
        {
            "Key": "costCenter",
            "Value": "120"
        },
        {
            "Key": "project",
            "Value": "lima"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLifecyclePolicy。

Amazon Data Pipeline使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Data Pipeline 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`activate-pipeline`

以下代码示例演示了如何使用 `activate-pipeline`。

Amazon CLI

激活管道

此示例激活指定的管道：

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

要在特定的日期和时间激活管道，请使用以下命令：

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE --start-
timestamp 2015-04-07T00:00:00Z
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ActivatePipeline。

`add-tags`

以下代码示例演示了如何使用 `add-tags`。

Amazon CLI

向管道添加标签

此示例向指定管道添加指定标签：

```
aws datapipeline add-tags --pipeline-id df-00627471SOVYZEXAMPLE --
tags key=environment,value=production key=owner,value=sales
```

要查看标签，请使用 describe-pipelines 命令。例如，示例命令中添加的标签在 describe-pipelines 的输出中如下所示：

```
{
    ...
        "tags": [
            {
                "value": "production",
                "key": "environment"
            },
            {
                "value": "sales",
                "key": "owner"
            }
        ]
    ...
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTags。

create-pipeline

以下代码示例演示了如何使用 create-pipeline。

Amazon CLI

创建管道

此示例创建一个管道：

```
aws datapipeline create-pipeline --name my-pipeline --unique-id my-pipeline-token
```

下面是示例输出：

```
{
    "pipelineId": "df-00627471SOVYZEXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePipeline。

deactivate-pipeline

以下代码示例演示了如何使用 deactivate-pipeline。

Amazon CLI

停用管道

此示例停用指定的管道：

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

要仅在所有正在运行的活动完成后停用管道，请使用以下命令：

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE --no-cancel-active
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeactivatePipeline。

delete-pipeline

以下代码示例演示了如何使用 delete-pipeline。

Amazon CLI

删除管道

此示例删除指定的管道：

```
aws datapipeline delete-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePipeline。

describe-pipelines

以下代码示例演示了如何使用 describe-pipelines。

Amazon CLI

描述您的管道

此示例描述指定的管道：

```
aws datapipeline describe-pipelines --pipeline-ids df-00627471SOVYZEXAMPLE
```

下面是示例输出：


```
{
  "pipelineDescriptionList": [
      {
          "fields": [
              {
                  "stringValue": "PENDING",
                  "key": "@pipelineState"
              },
              {
                  "stringValue": "my-pipeline",
                  "key": "name"
              },
              {
                  "stringValue": "2015-04-07T16:05:58",
                  "key": "@creationTime"
              },
              {
                  "stringValue": "df-00627471SOVYZEXAMPLE",
                  "key": "@id"
              },
              {
                  "stringValue": "123456789012",
                  "key": "pipelineCreator"
              },
              {
                  "stringValue": "PIPELINE",
                  "key": "@sphere"
              },
              {
                  "stringValue": "123456789012",
                  "key": "@userId"
              },
              {
                  "stringValue": "123456789012",
                  "key": "@accountId"
              },
              {
                  "stringValue": "my-pipeline-token",
                  "key": "uniqueId"
              }
          ],
          "pipelineId": "df-00627471SOVYZEXAMPLE",
```

```
            "name": "my-pipeline",
            "tags": []
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePipelines。

get-pipeline-definition

以下代码示例演示了如何使用 get-pipeline-definition。

Amazon CLI

获取管道定义

此示例获取指定管道的管道定义：

```
aws datapipeline get-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE
```

下面是示例输出：

```
{
  "parameters": [
      {
          "type": "AWS::S3::ObjectKey",
          "id": "myS3OutputLoc",
          "description": "S3 output folder"
      },
      {
          "default": "s3://us-east-1.elasticmapreduce.samples/pig-apache-logs/data",
          "type": "AWS::S3::ObjectKey",
          "id": "myS3InputLoc",
          "description": "S3 input folder"
      },
      {
          "default": "grep -rc \"GET\" ${INPUT1_STAGING_DIR}/* >
 ${OUTPUT1_STAGING_DIR}/output.txt",
          "type": "String",
          "id": "myShellCmd",
          "description": "Shell command to run"
      }
  ],
```

```
  "objects": [
      {
          "type": "Ec2Resource",
          "terminateAfter": "20 Minutes",
          "instanceType": "t1.micro",
          "id": "EC2ResourceObj",
          "name": "EC2ResourceObj"
      },
      {
          "name": "Default",
          "failureAndRerunMode": "CASCADE",
          "resourceRole": "DataPipelineDefaultResourceRole",
          "schedule": {
              "ref": "DefaultSchedule"
          },
          "role": "DataPipelineDefaultRole",
          "scheduleType": "cron",
          "id": "Default"
      },
      {
          "directoryPath": "#{myS3OutputLoc}/#{format(@scheduledStartTime, 'YYYY-MM-
dd-HH-mm-ss')}",
          "type": "S3DataNode",
          "id": "S3OutputLocation",
          "name": "S3OutputLocation"
      },
      {
          "directoryPath": "#{myS3InputLoc}",
          "type": "S3DataNode",
          "id": "S3InputLocation",
          "name": "S3InputLocation"
      },
      {
          "startAt": "FIRST_ACTIVATION_DATE_TIME",
          "name": "Every 15 minutes",
          "period": "15 minutes",
          "occurrences": "4",
          "type": "Schedule",
          "id": "DefaultSchedule"
      },
      {
          "name": "ShellCommandActivityObj",
          "command": "#{myShellCmd}",
          "output": {
```

```
                "ref": "S3OutputLocation"
            },
            "input": {
                "ref": "S3InputLocation"
            },
            "stage": "true",
            "type": "ShellCommandActivity",
            "id": "ShellCommandActivityObj",
            "runsOn": {
                "ref": "EC2ResourceObj"
            }
        }
    ],
    "values": {
        "myS3OutputLoc": "s3://amzn-s3-demo-bucket/",
        "myS3InputLoc": "s3://us-east-1.elasticmapreduce.samples/pig-apache-logs/
data",
        "myShellCmd": "grep -rc \"GET\" ${INPUT1_STAGING_DIR}/* >
 ${OUTPUT1_STAGING_DIR}/output.txt"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPipelineDefinition。

`list-pipelines`

以下代码示例演示了如何使用 `list-pipelines`。

Amazon CLI

列出您的管道

此示例列出您的管道：

```
aws datapipeline list-pipelines
```

下面是示例输出：

```
{
    "pipelineIdList": [
        {
            "id": "df-00627471SOVYZEXAMPLE",
```

```
            "name": "my-pipeline"
        },
        {
            "id": "df-09028963KNVMREXAMPLE",
            "name": "ImportDDB"
        },
        {
            "id": "df-0870198233ZYVEXAMPLE",
            "name": "CrossRegionDDB"
        },
        {
            "id": "df-00189603TB4MZEXAMPLE",
            "name": "CopyRedshift"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPipelines。

list-runs

以下代码示例演示了如何使用 `list-runs`。

Amazon CLI

示例 1：列出管道的运行情况

以下 `list-runs` 示例列出指定管道的运行情况。

```
aws datapipeline list-runs --pipeline-id df-00627471SOVYZEXAMPLE
```

输出：

```
    Name                        Scheduled Start        Status                      ID
                                             Started                Ended

 ---------------------------------------------------------------------------------------
1.  EC2ResourceObj              2015-04-12T17:33:02    CREATING
 @EC2ResourceObj_2015-04-12T17:33:02             2015-04-12T17:33:10
2.  S3InputLocation             2015-04-12T17:33:02    FINISHED
      @S3InputLocation_2015-04-12T17:33:02            2015-04-12T17:33:09
 2015-04-12T17:33:09
```

```
3.  S3OutputLocation           2015-04-12T17:33:02    WAITING_ON_DEPENDENCIES
 @S3OutputLocation_2015-04-12T17:33:02           2015-04-12T17:33:09
4.  ShellCommandActivityObj    2015-04-12T17:33:02    WAITING_FOR_RUNNER
 @ShellCommandActivityObj_2015-04-12T17:33:02    2015-04-12T17:33:09
```

示例 2：列出在指定日期之间的管道运行情况

以下 `list-runs` 示例使用 `--start-interval` 来指定要包含在输出中的日期。

```
aws datapipeline list-runs --pipeline-id df-01434553B58A2SHZUKO5 --start-
interval 2017-10-07T00:00:00,2017-10-08T00:00:00
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRuns。

put-pipeline-definition

以下代码示例演示了如何使用 `put-pipeline-definition`。

Amazon CLI

上传管道定义

此示例将指定的管道定义上传到指定的管道：

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://my-pipeline-definition.json
```

下面是示例输出：

```
{
  "validationErrors": [],
  "errored": false,
  "validationWarnings": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPipelineDefinition。

remove-tags

以下代码示例演示了如何使用 `remove-tags`。

Amazon CLI

从管道中移除标签

此示例从指定管道中移除指定标签：

```
aws datapipeline remove-tags --pipeline-id df-00627471SOVYZEXAMPLE --tag-
keys environment
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTags。

使用 Amazon CLI 的 DataSync 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 DataSync 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

update-location-azure-blob

以下代码示例演示了如何使用 update-location-azure-blob。

Amazon CLI

使用新代理更新您的传输位置

以下 update-location-object-storage 示例使用新代理更新 Microsoft Azure Blob Storage 的 DataSync 位置。

```
aws datasync update-location-azure-blob \
    --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-
abcdef01234567890 \
```

```
    --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/
agent-1234567890abcdef0 \
    --sas-configuration '{ \
        "Token": "sas-token-for-azure-blob-storage-access" \
    }'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DataSync 用户指南》中的更换代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLocationAzureBlob。

update-location-hdfs

以下代码示例演示了如何使用 update-location-hdfs。

Amazon CLI

使用新代理更新您的传输位置

以下 update-location-hdfs 示例使用新代理更新您的 DataSync HDFS 位置。只有当您的 HDFS 集群使用 Kerberos 身份验证时，才需要 --kerberos-keytab 和 --kerberos-krb5-conf 选项。

```
aws datasync update-location-hdfs \
    --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-
abcdef01234567890 \
    --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/
agent-1234567890abcdef0 \
    --kerberos-keytab file://hdfs.keytab
    --kerberos-krb5-conf file://krb5.conf
```

的内容hdfs.keytab：

```
N/A. The content of this file is encrypted and not human readable.
```

的内容krb5.conf：

```
[libdefaults]
    default_realm = EXAMPLE.COM
    dns_lookup_realm = false
    dns_lookup_kdc = false
```

```
    rdns = true
    ticket_lifetime = 24h
    forwardable = true
    udp_preference_limit = 1000000
    default_tkt_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-
sha1
    default_tgs_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-
sha1
    permitted_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-
sha1

[realms]
    EXAMPLE.COM = {
      kdc = kdc1.example.com
      admin_server = krbadmin.example.com
      default_domain = example.com
    }

[domain_realm]
    .example.com = EXAMPLE.COM
    example.com = EXAMPLE.COM

[logging]
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kerberos/kadmin.log
    default = FILE:/var/log/krb5libs.log
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DataSync 用户指南》中的更换代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLocationHdfs。

update-location-nfs

以下代码示例演示了如何使用 update-location-nfs。

Amazon CLI

使用新代理更新您的传输位置

以下 update-location-nfs 示例使用新代理更新您的 DataSync NFS 位置。

```
aws datasync update-location-nfs \
```

```
    --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-
abcdef01234567890 \
    --on-prem-config AgentArns=arn:aws:datasync:us-west-2:123456789012:agent/
agent-1234567890abcdef0
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DataSync 用户指南》中的更换代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLocationNfs。

update-location-object-storage

以下代码示例演示了如何使用 update-location-object-storage。

Amazon CLI

使用新代理更新您的传输位置

以下 update-location-object-storage 示例使用新代理更新您 DataSync 对象的存储位置。

```
aws datasync update-location-object-storage \
    --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-
abcdef01234567890 \
    --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/
agent-1234567890abcdef0 \
    --secret-key secret-key-for-object-storage
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DataSync 用户指南》中的更换代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLocationObjectStorage。

update-location-smb

以下代码示例演示了如何使用 update-location-smb。

Amazon CLI

使用新代理更新您的传输位置

以下 update-location-smb 示例使用新代理更新您的 DataSync SMB 位置。

```
aws datasync update-location-smb \
    --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-
abcdef01234567890 \
    --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/
agent-1234567890abcdef0 \
    --password smb-file-server-password
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DataSync 用户指南》中的更换代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLocationSmb。

使用 Amazon CLI 的 DAX 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 DAX 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-cluster`

以下代码示例演示了如何使用 `create-cluster`。

Amazon CLI

创建 DAX 集群

以下 `create-cluster` 示例创建具有指定设置的 DAX 集群。

```
aws dax create-cluster \
```

```
    --cluster-name daxcluster \
    --node-type dax.r4.large \
    --replication-factor 3 \
    --iam-role-arn roleARN  \
    --sse-specification Enabled=true
```

输出：

```
{
    "Cluster": {
        "ClusterName": "daxcluster",
        "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
        "TotalNodes": 3,
        "ActiveNodes": 0,
        "NodeType": "dax.r4.large",
        "Status": "creating",
        "ClusterDiscoveryEndpoint": {
            "Port": 8111
        },
        "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
        "SubnetGroup": "default",
        "SecurityGroups": [
            {
                "SecurityGroupIdentifier": "sg-1af6e36e",
                "Status": "active"
            }
        ],
        "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
        "ParameterGroup": {
            "ParameterGroupName": "default.dax1.0",
            "ParameterApplyStatus": "in-sync",
            "NodeIdsToReboot": []
        },
        "SSEDescription": {
            "Status": "ENABLED"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的步骤 3：创建 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-parameter-group

以下代码示例演示了如何使用 create-parameter-group。

Amazon CLI

创建参数组

以下“create-parameter-group”示例创建一个具有指定设置的参数组。

```
aws dax create-parameter-group \
    --parameter-group-name daxparametergroup \
    --description "A new parameter group"
```

输出：

```
{
    "ParameterGroup": {
        "ParameterGroupName": "daxparametergroup",
        "Description": "A new parameter group"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateParameterGroup。

create-subnet-group

以下代码示例演示了如何使用 create-subnet-group。

Amazon CLI

创建 DAX 子网组

以下 create-subnet-group 示例创建具有指定设置的子网组。

```
aws dax create-subnet-group \
    --subnet-group-name daxSubnetGroup \
    --subnet-ids subnet-11111111 subnet-22222222
```

输出：

```
{
    "SubnetGroup": {
        "SubnetGroupName": "daxSubnetGroup",
        "VpcId": "vpc-05a1fa8e00c325226",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-11111111",
                "SubnetAvailabilityZone": "us-west-2b"
            },
            {
                "SubnetIdentifier": "subnet-22222222",
                "SubnetAvailabilityZone": "us-west-2c"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的步骤 2：创建子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubnetGroup。

decrease-replication-factor

以下代码示例演示了如何使用 decrease-replication-factor。

Amazon CLI

从集群移除一个或多个节点

以下 decrease-replication-factor 示例将指定 DAX 集群中节点的数量减少到一个。

```
aws dax decrease-replication-factor \
    --cluster-name daxcluster \
    --new-replication-factor 1
```

输出：

```
{
    "Cluster": {
        "ClusterName": "daxcluster",
        "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
        "TotalNodes": 3,
        "ActiveNodes": 3,
```

```
        "NodeType": "dax.r4.large",
        "Status": "modifying",
        "ClusterDiscoveryEndpoint": {
            "Address": "daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",
            "Port": 8111
        },
        "Nodes": [
            {
                "NodeId": "daxcluster-a",
                "Endpoint": {
                    "Address": "daxcluster-
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
                    "Port": 8111
                },
                "NodeCreateTime": 1576625059.509,
                "AvailabilityZone": "us-west-2c",
                "NodeStatus": "available",
                "ParameterGroupStatus": "in-sync"
            },
            {
                "NodeId": "daxcluster-b",
                "Endpoint": {
                    "Address": "daxcluster-
b.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
                    "Port": 8111
                },
                "NodeCreateTime": 1576625059.509,
                "AvailabilityZone": "us-west-2a",
                "NodeStatus": "available",
                "ParameterGroupStatus": "in-sync"
            },
            {
                "NodeId": "daxcluster-c",
                "Endpoint": {
                    "Address": "daxcluster-
c.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
                    "Port": 8111
                },
                "NodeCreateTime": 1576625059.509,
                "AvailabilityZone": "us-west-2b",
                "NodeStatus": "available",
                "ParameterGroupStatus": "in-sync"
            }
        ],
```

```
        "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
        "SubnetGroup": "default",
        "SecurityGroups": [
            {
                "SecurityGroupIdentifier": "sg-1af6e36e",
                "Status": "active"
            }
        ],
        "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
        "ParameterGroup": {
            "ParameterGroupName": "default.dax1.0",
            "ParameterApplyStatus": "in-sync",
            "NodeIdsToReboot": []
        },
        "SSEDescription": {
            "Status": "ENABLED"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecreaseReplicationFactor。

delete-cluster

以下代码示例演示了如何使用 delete-cluster。

Amazon CLI

删除 DAX 集群

以下 delete-cluster 示例删除指定的 DAX 集群。

```
aws dax delete-cluster \
    --cluster-name daxcluster
```

输出：

```
{
    "Cluster": {
        "ClusterName": "daxcluster",
```

```
        "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
        "TotalNodes": 3,
        "ActiveNodes": 0,
        "NodeType": "dax.r4.large",
        "Status": "deleting",
        "ClusterDiscoveryEndpoint": {
            "Address": "dd.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",
            "Port": 8111
        },
        "PreferredMaintenanceWindow": "fri:06:00-fri:07:00",
        "SubnetGroup": "default",
        "SecurityGroups": [
            {
                "SecurityGroupIdentifier": "sg-1af6e36e",
                "Status": "active"
            }
        ],
        "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
        "ParameterGroup": {
            "ParameterGroupName": "default.dax1.0",
            "ParameterApplyStatus": "in-sync",
            "NodeIdsToReboot": []
        },
        "SSEDescription": {
            "Status": "ENABLED"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCluster。

delete-parameter-group

以下代码示例演示了如何使用 delete-parameter-group。

Amazon CLI

删除参数组

以下 delete-parameter-group 示例删除指定的 DAX 参数组。

```
aws dax  delete-parameter-group \
    --parameter-group-name daxparametergroup
```

输出：

```
{
    "DeletionMessage": "Parameter group daxparametergroup has been deleted."
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteParameterGroup。

delete-subnet-group

以下代码示例演示了如何使用 delete-subnet-group。

Amazon CLI

删除子网组

以下 delete-subnet-group 示例删除指定的 DAX 子网组。

```
aws dax delete-subnet-group \
    --subnet-group-name daxSubnetGroup
```

输出：

```
{
    "DeletionMessage": "Subnet group daxSubnetGroup has been deleted."
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubnetGroup。

describe-clusters

以下代码示例演示了如何使用 describe-clusters。

Amazon CLI

返回有关所有预置 DAX 集群的信息

以下 describe-clusters 示例显示有关所有预置 DAX 集群的详细信息。

```
aws dax describe-clusters
```

输出：

```
{
    "Clusters": [
        {
            "ClusterName": "daxcluster",
            "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
            "TotalNodes": 1,
            "ActiveNodes": 1,
            "NodeType": "dax.r4.large",
            "Status": "available",
            "ClusterDiscoveryEndpoint": {
                "Address":
 "daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",
                "Port": 8111
            },
            "Nodes": [
                {
                    "NodeId": "daxcluster-a",
                    "Endpoint": {
                        "Address": "daxcluster-
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
                        "Port": 8111
                    },
                    "NodeCreateTime": 1576625059.509,
                    "AvailabilityZone": "us-west-2c",
                    "NodeStatus": "available",
                    "ParameterGroupStatus": "in-sync"
                }
            ],
            "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
            "SubnetGroup": "default",
            "SecurityGroups": [
                {
                    "SecurityGroupIdentifier": "sg-1af6e36e",
```

```
                    "Status": "active"
                }
            ],
            "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
            "ParameterGroup": {
                "ParameterGroupName": "default.dax1.0",
                "ParameterApplyStatus": "in-sync",
                "NodeIdsToReboot": []
            },
            "SSEDescription": {
                "Status": "ENABLED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusters。

describe-default-parameters

以下代码示例演示了如何使用 describe-default-parameters。

Amazon CLI

返回 DAX 的默认系统参数信息

以下 describe-default-parameters 示例显示 DAX 的默认系统参数信息。

```
aws dax describe-default-parameters
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "query-ttl-millis",
            "ParameterType": "DEFAULT",
            "ParameterValue": "300000",
            "NodeTypeSpecificValues": [],
            "Description": "Duration in milliseconds for queries to remain cached",
```

```
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": "TRUE",
            "ChangeType": "IMMEDIATE"
        },
        {
            "ParameterName": "record-ttl-millis",
            "ParameterType": "DEFAULT",
            "ParameterValue": "300000",
            "NodeTypeSpecificValues": [],
            "Description": "Duration in milliseconds for records to remain valid in
 cache (Default: 0 = infinite)",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": "TRUE",
            "ChangeType": "IMMEDIATE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDefaultParameters。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

返回与 DAX 集群和参数组相关的所有事件

以下 describe-events 示例显示与 DAX 集群和参数组相关的事件的详细信息。

```
aws dax describe-events
```

输出：

```
{
    "Events": [
        {
```

```
            "SourceName": "daxcluster",
            "SourceType": "CLUSTER",
            "Message": "Cluster deleted.",
            "Date": 1576702736.706
        },
        {
            "SourceName": "daxcluster",
            "SourceType": "CLUSTER",
            "Message": "Removed node daxcluster-b.",
            "Date": 1576702691.738
        },
        {
            "SourceName": "daxcluster",
            "SourceType": "CLUSTER",
            "Message": "Removed node daxcluster-a.",
            "Date": 1576702633.498
        },
        {
            "SourceName": "daxcluster",
            "SourceType": "CLUSTER",
            "Message": "Removed node daxcluster-c.",
            "Date": 1576702631.329
        },
        {
            "SourceName": "daxcluster",
            "SourceType": "CLUSTER",
            "Message": "Cluster created.",
            "Date": 1576626560.057
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

`describe-parameter-groups`

以下代码示例演示了如何使用 `describe-parameter-groups`。

Amazon CLI

描述 DAX 中定义的参数组

以下 describe-parameter-groups 示例检索 DAX 中定义的参数组的详细信息。

```
aws dax describe-parameter-groups
```

输出：

```
{
    "ParameterGroups": [
        {
            "ParameterGroupName": "default.dax1.0",
            "Description": "Default parameter group for dax1.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeParameterGroups。

describe-parameters

以下代码示例演示了如何使用 describe-parameters。

Amazon CLI

描述 DAX 参数组中定义的参数

以下 describe-parameters 示例检索指定 DAX 参数组中定义的参数的详细信息。

```
aws dax describe-parameters \
    --parameter-group-name default.dax1.0
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "query-ttl-millis",
            "ParameterType": "DEFAULT",
            "ParameterValue": "300000",
            "NodeTypeSpecificValues": [],
```

```
            "Description": "Duration in milliseconds for queries to remain cached",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": "TRUE",
            "ChangeType": "IMMEDIATE"
        },
        {
            "ParameterName": "record-ttl-millis",
            "ParameterType": "DEFAULT",
            "ParameterValue": "300000",
            "NodeTypeSpecificValues": [],
            "Description": "Duration in milliseconds for records to remain valid in
 cache (Default: 0 = infinite)",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": "TRUE",
            "ChangeType": "IMMEDIATE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeParameters。

describe-subnet-groups

以下代码示例演示了如何使用 describe-subnet-groups。

Amazon CLI

描述 DAX 中定义的子网组

以下 describe-subnet-groups 示例检索 DAX 中定义的子网组的详细信息。

```
aws dax describe-subnet-groups
```

输出：

```
{
    "SubnetGroups": [
```

```
        {
            "SubnetGroupName": "default",
            "Description": "Default CacheSubnetGroup",
            "VpcId": "vpc-ee70a196",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-874953af",
                    "SubnetAvailabilityZone": "us-west-2d"
                },
                {
                    "SubnetIdentifier": "subnet-bd3d1fc4",
                    "SubnetAvailabilityZone": "us-west-2a"
                },
                {
                    "SubnetIdentifier": "subnet-72c2ff28",
                    "SubnetAvailabilityZone": "us-west-2c"
                },
                {
                    "SubnetIdentifier": "subnet-09e6aa42",
                    "SubnetAvailabilityZone": "us-west-2b"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSubnetGroups。

increase-replication-factor

以下代码示例演示了如何使用 increase-replication-factor。

Amazon CLI

提高 DAX 集群的复制系数

以下 increase-replication-factor 示例将指定 DAX 集群的复制系数增加到 3。

```
aws dax increase-replication-factor \
    --cluster-name daxcluster \
    --new-replication-factor 3
```

输出：

```
{
    "Cluster": {
        "ClusterName": "daxcluster",
        "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
        "TotalNodes": 3,
        "ActiveNodes": 1,
        "NodeType": "dax.r4.large",
        "Status": "modifying",
        "ClusterDiscoveryEndpoint": {
            "Address": "daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",
            "Port": 8111
        },
        "Nodes": [
            {
                "NodeId": "daxcluster-a",
                "Endpoint": {
                    "Address": "daxcluster-
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
                    "Port": 8111
                },
                "NodeCreateTime": 1576625059.509,
                "AvailabilityZone": "us-west-2c",
                "NodeStatus": "available",
                "ParameterGroupStatus": "in-sync"
            },
            {
                "NodeId": "daxcluster-b",
                "NodeStatus": "creating"
            },
            {
                "NodeId": "daxcluster-c",
                "NodeStatus": "creating"
            }
        ],
        "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
        "SubnetGroup": "default",
        "SecurityGroups": [
            {
                "SecurityGroupIdentifier": "sg-1af6e36e",
                "Status": "active"
            }
        ],
```

```
        "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
        "ParameterGroup": {
            "ParameterGroupName": "default.dax1.0",
            "ParameterApplyStatus": "in-sync",
            "NodeIdsToReboot": []
        },
        "SSEDescription": {
            "Status": "ENABLED"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IncreaseReplicationFactor。

`list-tags`

以下代码示例演示了如何使用 `list-tags`。

Amazon CLI

列出 DAX 资源上的标签

以下 `list-tags` 示例列出附加到指定 DAX 集群的标签键和值。

```
aws dax list-tags \
    --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster
```

输出：

```
{
    "Tags": [
        {
            "Key": "ClusterUsage",
            "Value": "prod"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTags。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

为 DAX 资源添加标签

以下 `tag-resource` 示例将指定的标签键名和关联值附加到指定的 DAX 集群，以描述集群的使用情况。

```
aws dax tag-resource \
    --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster \
    --tags="Key=ClusterUsage,Value=prod"
```

输出：

```
{
    "Tags": [
        {
            "Key": "ClusterUsage",
            "Value": "prod"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从 DAX 资源中移除标签

以下 `untag-resource` 示例从 DAX 集群中移除具有指定键名的标签。

```
aws dax untag-resource  \
    --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster \
    --tag-keys="ClusterUsage"
```

输出：

```
{
    "Tags": []
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的管理 DAX 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Detective 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Detective 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-invitation`

以下代码示例演示了如何使用 `accept-invitation`。

Amazon CLI

接受成为行为图中成员账户的邀请

以下 `accept-invitation` 示例接受成为行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中成员账户的邀请。

```
aws detective accept-invitation \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Detective 管理指南》中的响应行为图邀请。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptInvitation。

create-graph

以下代码示例演示了如何使用 create-graph。

Amazon CLI

启用 Amazon Detective 并创建新的行为图

以下 create-graph 示例为在运行该命令的区域中运行该命令的 Amazon 帐户启用 Detective。创建一个新的行为图，将该账户作为管理员账户。该命令还将值 Finance 分配给 Department 标签。

```
aws detective create-graph \
    --tags '{"Department": "Finance"}'
```

输出：

```
{
    "GraphArn": "arn:aws:detective:us-
east-1:111122223333:graph:027c7c4610ea4aacaf0b883093cab899"
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的启用 Amazon Detective。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGraph。

create-members

以下代码示例演示了如何使用 create-members。

Amazon CLI

邀请成员账户加入行为图

以下 `create-members` 示例邀请两个 Amazon 账户成为行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中的成员账户。对于每个账户，请求会提供 Amazon 账户 ID 和账户根用户电子邮件地址。该请求包括要插入到邀请电子邮件中的自定义消息。

```
aws detective create-members \
    --
accounts AccountId=444455556666,EmailAddress=mmajor@example.com AccountId=123456789012,Email
 \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \
    --message "This is Paul Santos. I need to add your account to the data we use
 for security investigation in Amazon Detective. If you have any questions, contact
 me at psantos@example.com."
```

输出：

```
{
    "Members": [
    {
        "AccountId": "444455556666",
        "AdministratorId": "111122223333",
        "EmailAddress": "mmajor@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "INVITED",
        "UpdatedTime": 1579826107000
   },
   {
        "AccountId": "123456789012",
        "AdministratorId": "111122223333",
        "EmailAddress": "jstiles@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "VERIFICATION_IN_PROGRESS",
        "UpdatedTime": 1579826107000
     }
    ],
    "UnprocessedAccounts": [ ]
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“邀请成员账户加入行为图<https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-add-member-accounts.html>”。

在无需发送邀请电子邮件的情况下邀请成员账户

以下 `create-members` 示例邀请两个 Amazon 账户成为行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中的成员账户。对于每个账户，请求会提供 Amazon 账户 ID 和账户根用户电子邮件地址。成员账户不会收到邀请电子邮件。

```
aws detective create-members \
    --
accounts AccountId=444455556666,EmailAddress=mmajor@example.com AccountId=123456789012,Email
 \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \
    --disable-email-notification
```

输出：

```
{
    "Members": [
    {
        "AccountId": "444455556666",
        "AdministratorId": "111122223333",
        "EmailAddress": "mmajor@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "INVITED",
        "UpdatedTime": 1579826107000
   },
   {
        "AccountId": "123456789012",
        "AdministratorId": "111122223333",
        "EmailAddress": "jstiles@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "VERIFICATION_IN_PROGRESS",
        "UpdatedTime": 1579826107000
     }
    ],
    "UnprocessedAccounts": [ ]
```

```
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“邀请成员账户加入行为图<https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-add-member-accounts.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMembers。

`delete-graph`

以下代码示例演示了如何使用 `delete-graph`。

Amazon CLI

禁用 Detective 并删除行为图

以下 `delete-graph` 示例禁用 Detective 并删除指定的行为图。

```
aws detective delete-graph \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Detective 管理指南》中的禁用 Amazon Detective。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGraph。

`delete-members`

以下代码示例演示了如何使用 `delete-members`。

Amazon CLI

从行为图中移除成员账户

以下 `delete-members` 示例从行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中移除两个成员账户。为了识别账户，该请求提供了 Amazon 账户 ID。

```
aws detective delete-members \
    --account-ids 444455556666 123456789012 \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

输出：

```
{
    "AccountIds": [ "444455556666", "123456789012" ],
    "UnprocessedAccounts": [ ]
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“从行为图中移除成员账户<https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-remove-member-accounts.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMembers。

`disassociate-membership`

以下代码示例演示了如何使用 `disassociate-membership`。

Amazon CLI

放弃行为图成员资格

以下 disassociate-membership 示例从行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中移除运行命令的 Amazon 账户。

```
aws detective disassociate-membership \
     --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“从行为图中移除账户<https://docs.aws.amazon.com/detective/latest/adminguide/member-remove-self-from-graph.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateMembership。

`get-members`

以下代码示例演示了如何使用 `get-members`。

Amazon CLI

检索所选行为图成员账户的相关信息

以下 `get-members` 示例检索行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中两个成员账户的信息。对于这两个账户，该请求提供了 Amazon 账户 ID。

```
aws detective get-members \
    --account-ids 444455556666 123456789012 \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

输出：

```
{
    "MemberDetails": [
    {
        "AccountId": "444455556666",
        "AdministratorId": "111122223333",
        "EmailAddress": "mmajor@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "INVITED",
        "UpdatedTime": 1579826107000
    }
    {
        "AccountId": "123456789012",
        "AdministratorId": "111122223333",
        "EmailAddress": "jstiles@example.com",
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "INVITED",
        "UpdatedTime": 1579826107000
    }
],
    "UnprocessedAccounts": [ ]
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“查看行为图中账户的列表<https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-view-accounts.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMembers。

list-graphs

以下代码示例演示了如何使用 list-graphs。

Amazon CLI

查看您的账户作为管理员的行为图列表

以下 `list-graphs` 示例检索当前区域内调用账户作为管理员的行为图。

```
aws detective list-graphs
```

输出：

```
{
    "GraphList": [
        {
            "Arn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
            "CreatedTime": 1579736111000
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGraphs。

list-invitations

以下代码示例演示了如何使用 `list-invitations`。

Amazon CLI

查看账户加入或受邀加入的行为图列表

以下 `list-invitations` 示例检索调用账户已被邀请的行为图。结果仅包括公开和已接受的邀请。其中不包括被拒绝的邀请或已移除的成员资格。

```
aws detective list-invitations
```

输出：

```
{
    "Invitations": [
    {
        "AccountId": "444455556666",
        "AdministratorId": "111122223333",
        "EmailAddress": "mmajor@example.com",
```

```
        "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
        "InvitedTime": 1579826107000,
        "MasterId": "111122223333",
        "Status": "INVITED",
        "UpdatedTime": 1579826107000
    }
]
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的“查看行为图邀请的列表<https://docs.aws.amazon.com/detective/latest/adminguide/member-view-graph-invitations.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInvitations。

`list-members`

以下代码示例演示了如何使用 `list-members`。

Amazon CLI

列出行为图中的成员账户

以下 `list-members` 示例检索行为图 `arn:aws:detective:us-east-1:111122223333:graph:123412341234` 的已邀请和已启用的成员账户。结果不包括已移除的成员账户。

```
aws detective list-members \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

输出：

```
{
    "MemberDetails": [
        {
            "AccountId": "444455556666",
            "AdministratorId": "111122223333",
            "EmailAddress": "mmajor@example.com",
            "GraphArn": "arn:aws:detective:us-
east-1:111122223333:graph:123412341234",
            "InvitedTime": 1579826107000,
            "MasterId": "111122223333",
            "Status": "INVITED",
            "UpdatedTime": 1579826107000
```

```
        },
        {
            "AccountId": "123456789012",
            "AdministratorId": "111122223333",
            "EmailAddress": "jstiles@example.com",
            "GraphArn": "arn:aws:detective:us-
east-1:111122223333:graph:123412341234",
            "InvitedTime": 1579826107000,
            "MasterId": "111122223333",
            "PercentOfGraphUtilization": 2,
            "PercentOfGraphUtilizationUpdatedTime": 1586287843,
            "Status": "ENABLED",
            "UpdatedTime": 1579973711000,
            "VolumeUsageInBytes": 200,
            "VolumeUsageUpdatedTime": 1586287843
        }
    ]
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的查看行为图中账户的列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMembers。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

检索分配给行为图的标签

以下 list-tags-for-resource 示例返回分配给指定行为图的标签。

```
aws detective list-tags-for-resource \
    --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

输出：

```
{
    "Tags": {
        "Department" : "Finance"
    }
}
```

有关更多信息，请参阅《Amazon Detective 管理指南》中的管理行为图的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`reject-invitation`

以下代码示例演示了如何使用 `reject-invitation`。

Amazon CLI

拒绝成为行为图中成员账户的邀请

以下 `reject-invitation` 示例拒绝成为行为图 arn:aws:detective:us-east-1:111122223333:graph:123412341234 中成员账户的邀请。

```
aws detective reject-invitation \
    --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Detective 管理指南》中的“响应行为图邀请<https://docs.aws.amazon.com/detective/latest/adminguide/member-invitation-response.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectInvitation。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

将标签添加到资源

以下 `tag-resource` 示例为指定行为图的 Department 标签赋值。

```
aws detective tag-resource \
    --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \
    --tags '{"Department":"Finance"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Detective 管理指南》中的管理行为图的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源中删除标签值

以下 `untag-resource` 示例从指定的行为图中移除 Department 标签。

```
aws detective untag-resource \
    --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \
    --tag-keys "Department"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Detective 管理指南》中的管理行为图的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Device Farm 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Device Farm 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-device-pool`

以下代码示例演示了如何使用 `create-device-pool`。

Amazon CLI

创建设备池

以下命令为项目创建 Android 设备池：

```
aws devicefarm create-device-pool --name pool1 --rules file://
device-pool-rules.json --project-arn "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506"
```

您可以从 `create-project` 或 `list-projects` 的输出中获取项目 ARN。文件 `device-pool-rules.json` 是当前文件夹中指定设备平台的 JSON 文档：

```
[
    {
        "attribute": "PLATFORM",
        "operator": "EQUALS",
        "value": "\"ANDROID\""
    }
]
```

输出：

```
{
    "devicePool": {
        "rules": [
            {
                "operator": "EQUALS",
                "attribute": "PLATFORM",
                "value": "\"ANDROID\""
            }
        ],
        "type": "PRIVATE",
        "name": "pool1",
        "arn": "arn:aws:devicefarm:us-
west-2:123456789012:devicepool:070fc3ca-7ec1-4741-9c1f-
d3e044efc506/2aa8d2a9-5e73-47ca-b929-659cb34b7dcd"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDevicePool。

create-project

以下代码示例演示了如何使用 create-project。

Amazon CLI

创建项目

以下命令创建名为 my-project 的新项目：

```
aws devicefarm create-project --name my-project
```

输出：

```
{
    "project": {
        "name": "myproject",
        "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506",
        "created": 1503612890.057
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProject。

create-upload

以下代码示例演示了如何使用 create-upload。

Amazon CLI

创建上传

以下命令为 Android 应用创建上传：

```
aws devicefarm create-upload --project-arn "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506" --name app.apk --
type ANDROID_APP
```

您可以从 create-project 或 list-projects 的输出中获取项目 ARN。

输出：

```
{
    "upload": {
        "status": "INITIALIZED",
        "name": "app.apk",
        "created": 1503614408.769,
        "url": "https://prod-us-west-2-uploads.s3-us-west-2.amazonaws.com/
arn%3Aaws%3Adevicefarm%3Aus-west-2%3A123456789012%3Aproject%3A070fc3ca-
c7e1-4471-91cf-d3e4efc50604/uploads/arn%3Aaws%3Adevicefarm%3Aus-
west-2%3A123456789012%3Aupload%3A070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514/app.apk?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170824T224008Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-Amz-
Credential=AKIAEXAMPLEPBUMBC3GA%2F20170824%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-
Signature=05050370c38894ef5bd09f5d009f36fc8f96fa4bb04e1bba9aca71b8dbe49a0f",
        "type": "ANDROID_APP",
        "arn": "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514"
    }
}
```

使用输出中的已签名 URL 将文件上传到 Device Farm：

```
curl -T app.apk "https://prod-us-west-2-uploads.s3-us-west-2.amazonaws.com/
arn%3Aaws%3Adevicefarm%3Aus-west-2%3A123456789012%3Aproject%3A070fc3ca-
c7e1-4471-91cf-d3e4efc50604/uploads/arn%3Aaws%3Adevicefarm%3Aus-
west-2%3A123456789012%3Aupload%3A070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514/app.apk?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170824T224008Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-Amz-
Credential=AKIAEXAMPLEPBUMBC3GA%2F20170824%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-
Signature=05050370c38894ef5bd09f5d009f36fc8f96fa4bb04e1bba9aca71b8dbe49a0f"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUpload。

get-upload

以下代码示例演示了如何使用 get-upload。

Amazon CLI

查看上传

以下命令检索有关上传的信息：

```
aws devicefarm get-upload --arn "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514"
```

您可以从 create-upload 的输出中获取上传 ARN。

输出：

```
{
    "upload": {
        "status": "SUCCEEDED",
        "name": "app.apk",
        "created": 1505262773.186,
        "type": "ANDROID_APP",
        "arn": "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514",
        "metadata": "{\"device_admin\":false,\"activity_name\":
\"ccom.example.client.LauncherActivity\",\"version_name\":\"1.0.2.94\",\"screens
\":[\"small\",\"normal\",\"large\",\"xlarge\"],\"error_type\":null,\"sdk_version
\":\"16\",\"package_name\":\"com.example.client\",\"version_code\":\"20994\",
\"native_code\":[\"armeabi-v7a\"],\"target_sdk_version\":\"25\"}"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUpload。

list-projects

以下代码示例演示了如何使用 list-projects。

Amazon CLI

列出项目

以下内容检索项目列表：

```
aws devicefarm list-projects
```

输出：

```
{
```

```
    "projects": [
        {
            "name": "myproject",
            "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506",
            "created": 1503612890.057
        },
        {
            "name": "otherproject",
            "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:a5f5b752-8098-49d1-86bf-5f7682c1c77e",
            "created": 1505257519.337
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProjects。

Amazon Direct Connect使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Direct Connect 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-direct-connect-gateway-association-proposal`

以下代码示例演示了如何使用 `accept-direct-connect-gateway-association-proposal`。

Amazon CLI

接受网关关联提议

以下 accept-direct-connect-gateway-association-proposal 接受指定的提议。

```
aws directconnect  accept-direct-connect-gateway-association-proposal \
    --direct-connect-gateway-id 11460968-4ac1-4fd3-bdb2-00599EXAMPLE \
    --proposal-id cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE \
    --associated-gateway-owner-account 111122223333

{
    "directConnectGatewayAssociation": {
        "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
        "directConnectGatewayOwnerAccount": "111122223333",
        "associationState": "associating",
        "associatedGateway": {
            "id": "tgw-02f776b1a7EXAMPLE",
            "type": "transitGateway",
            "ownerAccount": "111122223333",
            "region": "us-east-1"
        },
        "associationId": "6441f8bf-5917-4279-ade1-9708bEXAMPLE",
        "allowedPrefixesToDirectConnectGateway": [
            {
                "cidr": "192.168.1.0/30"
            }
        ]

  }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的接受或拒绝 Transit Gateway 关联提议。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptDirectConnectGatewayAssociationProposal。

allocate-connection-on-interconnect

以下代码示例演示了如何使用 allocate-connection-on-interconnect。

Amazon CLI

在互连上创建托管连接

以下 allocate-connection-on-interconnect 命令在互连上创建托管连接：

```
aws directconnect allocate-connection-on-interconnect --bandwidth 500Mbps --
connection-name mydcinterconnect --owner-account 123456789012 --interconnect-
id dxcon-fgktov66 --vlan 101
```

输出：

```
{
    "partnerName": "TIVIT",
    "vlan": 101,
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffzc51m1",
    "connectionState": "ordering",
    "bandwidth": "500Mbps",
    "location": "TIVIT",
    "connectionName": "mydcinterconnect",
    "region": "sa-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateConnectionOnInterconnect。

allocate-hosted-connection

以下代码示例演示了如何使用 allocate-hosted-connection。

Amazon CLI

在互连上创建托管连接

以下 allocate-hosted-connection 示例在指定互连上创建托管连接。

```
aws directconnect allocate-hosted-connection \
    --bandwidth 500Mbps \
    --connection-name mydcinterconnect \
    --owner-account 123456789012
    -connection-id dxcon-fgktov66
    -vlan 101
```

输出：

```
{
```

```
    "partnerName": "TIVIT",
    "vlan": 101,
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffzc51m1",
    "connectionState": "ordering",
    "bandwidth": "500Mbps",
    "location": "TIVIT",
    "connectionName": "mydcinterconnect",
    "region": "sa-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateHostedConnection。

allocate-private-virtual-interface

以下代码示例演示了如何使用 allocate-private-virtual-interface。

Amazon CLI

预置私有虚拟接口

以下 allocate-private-virtual-interface 命令预置由不同客户拥有的私有虚拟接口。

```
aws directconnect allocate-private-virtual-interface --connection-id dxcon-
ffjrkx17 --owner-account 123456789012 --new-private-virtual-interface-
allocation virtualInterfaceName=PrivateVirtualInterface,vlan=1000,asn=65000,authKey=asdf34ex
```

输出：

```
{
    "virtualInterfaceState": "confirming",
    "asn": 65000,
    "vlan": 1000,
    "customerAddress": "192.168.1.2/30",
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffjrkx17",
    "virtualInterfaceId": "dxvif-fgy8orxu",
    "authKey": "asdf34example",
    "routeFilterPrefixes": [],
    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n  <logical_connection id=\"dxvif-fgy8orxu\">\n  <vlan>1000</
vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n
```

```
 <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>private</connection_type>\n</logical_connection>\n",
    "amazonAddress": "192.168.1.1/30",
    "virtualInterfaceType": "private",
    "virtualInterfaceName": "PrivateVirtualInterface"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocatePrivateVirtualInterface。

allocate-public-virtual-interface

以下代码示例演示了如何使用 allocate-public-virtual-interface。

Amazon CLI

预置公有虚拟接口

以下 allocate-public-virtual-interface 命令预置由不同客户拥有的公有虚拟接口。

```
aws directconnect allocate-public-virtual-interface --connection-id dxcon-
ffjrkx17 --owner-account 123456789012 --new-public-virtual-interface-
allocation virtualInterfaceName=PublicVirtualInterface,vlan=2000,asn=65000,authKey=asdf34exa
{cidr=203.0.113.4/30}]
```

输出：

```
{
    "virtualInterfaceState": "confirming",
    "asn": 65000,
    "vlan": 2000,
    "customerAddress": "203.0.113.2/30",
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffjrkx17",
    "virtualInterfaceId": "dxvif-fg9xo9vp",
    "authKey": "asdf34example",
    "routeFilterPrefixes": [
        {
            "cidr": "203.0.113.0/30"
        },
        {
            "cidr": "203.0.113.4/30"
        }
```

```
    ],
    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fg9xo9vp\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
 <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
    "amazonAddress": "203.0.113.1/30",
    "virtualInterfaceType": "public",
    "virtualInterfaceName": "PublicVirtualInterface"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocatePublicVirtualInterface。

allocate-transit-virtual-interface

以下代码示例演示了如何使用 allocate-transit-virtual-interface。

Amazon CLI

预置由指定 Amazon 账户拥有的中转虚拟接口

以下 allocate-transit-virtual-interface 示例为指定账户预置中转虚拟接口。

```
aws directconnect allocate-transit-virtual-interface \
    --connection-id dxlag-fEXAMPLE \
    --owner-account 123456789012 \
    --new-transit-virtual-interface-allocation "virtualInterfaceName=Example Transit
 Virtual
 Interface,vlan=126,asn=65110,mtu=1500,authKey=0xzxgA9YoW9h58u8SEXAMPLE,amazonAddress=192.16
```

输出：

```
{
    "virtualInterface": {
        "ownerAccount": "123456789012",
        "virtualInterfaceId": "dxvif-fEXAMPLE",
        "location": "loc1",
        "connectionId": "dxlag-fEXAMPLE",
        "virtualInterfaceType": "transit",
        "virtualInterfaceName": "Example Transit Virtual Interface",
        "vlan": 126,
```

```
        "asn": 65110,
        "amazonSideAsn": 7224,
        "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
        "amazonAddress": "192.168.1.1/30",
        "customerAddress": "192.168.1.2/30",
        "addressFamily": "ipv4",
        "virtualInterfaceState": "confirming",
        "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fEXAMPLE\">\n  <vlan>126</
vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n
 <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65110</bgp_asn>\n
 <bgp_auth_key>0xzxgA9YoW9h58u8SEXAMPLE</bgp_auth_key>\n  <amazon_bgp_asn>7224</
amazon_bgp_asn>\n  <connection_type>transit</connection_type>\n</logical_connection>
\n",
        "mtu": 1500,
        "jumboFrameCapable": true,
        "virtualGatewayId": "",
        "directConnectGatewayId": "",
        "routeFilterPrefixes": [],
        "bgpPeers": [
            {
                "bgpPeerId": "dxpeer-fEXAMPLE",
                "asn": 65110,
                "authKey": "0xzxgA9YoW9h58u8EXAMPLE",
                "addressFamily": "ipv4",
                "amazonAddress": "192.168.1.1/30",
                "customerAddress": "192.168.1.2/30",
                "bgpPeerState": "pending",
                "bgpStatus": "down",
                "awsDeviceV2": "loc1-26wz6vEXAMPLE"
            }
        ],
        "region": "sa-east-1",
        "awsDeviceV2": "loc1-26wz6vEXAMPLE",
        "tags": [
            {
                "key": "Tag",
                "value": "Example"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的创建托管中转虚拟接口。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateTransitVirtualInterface。

associate-connection-with-lag

以下代码示例演示了如何使用 associate-connection-with-lag。

Amazon CLI

将连接与 LAG 关联

以下示例将指定连接与指定 LAG 关联。

命令:

```
aws directconnect associate-connection-with-lag --lag-id dxlag-fhccu14t --
connection-id dxcon-fg9607vm
```

输出：

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg9607vm",
  "lagId": "dxlag-fhccu14t",
  "connectionState": "requested",
  "bandwidth": "1Gbps",
  "location": "EqDC2",
  "connectionName": "Con2ForLag",
  "region": "us-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateConnectionWithLag。

associate-hosted-connection

以下代码示例演示了如何使用 associate-hosted-connection。

Amazon CLI

将托管连接与 LAG 关联

以下示例将指定托管连接与指定 LAG 关联。

命令:

```
aws directconnect associate-hosted-connection --parent-connection-id dxlag-fhccu14t
 --connection-id dxcon-fg9607vm
```

输出：

```
{
  "partnerName": "TIVIT",
  "vlan": 101,
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg9607vm",
  "lagId": "dxlag-fhccu14t",
  "connectionState": "ordering",
  "bandwidth": "500Mbps",
  "location": "TIVIT",
  "connectionName": "mydcinterconnect",
  "region": "sa-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateHostedConnection。

associate-virtual-interface

以下代码示例演示了如何使用 associate-virtual-interface。

Amazon CLI

将虚拟接口与连接关联

以下示例将指定虚拟接口与指定 LAG 关联。或者，要将虚拟接口与连接关联，请指定 --connection-id 的 Amazon Direct Connect 连接 ID，例如，dxcon-ffnikghc。

命令:

```
aws directconnect associate-virtual-interface --connection-id dxlag-ffjhj9lx --
virtual-interface-id dxvif-fgputw0j
```

输出：

```
{
  "virtualInterfaceState": "pending",
```

```
  "asn": 65000,
  "vlan": 123,
  "customerAddress": "169.254.255.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxlag-ffjhj9lx",
  "addressFamily": "ipv4",
  "virtualGatewayId": "vgw-38e90b51",
  "virtualInterfaceId": "dxvif-fgputw0j",
  "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
  "routeFilterPrefixes": [],
  "location": "CSVA1",
  "bgpPeers": [
      {
          "bgpStatus": "down",
          "customerAddress": "169.254.255.2/30",
          "addressFamily": "ipv4",
          "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
          "bgpPeerState": "deleting",
          "amazonAddress": "169.254.255.1/30",
          "asn": 65000
      },
      {
          "bgpStatus": "down",
          "customerAddress": "169.254.255.2/30",
          "addressFamily": "ipv4",
          "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
          "bgpPeerState": "pending",
          "amazonAddress": "169.254.255.1/30",
          "asn": 65000
      }
  ],
  "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fgputw0j\">\n  <vlan>123</vlan>
\n  <customer_address>169.254.255.2/30</customer_address>\n
 <amazon_address>169.254.255.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
  <bgp_auth_key>0x123pK5_VBqv.UQ3kJ4123_</bgp_auth_key>\n  <amazon_bgp_asn>7224</
amazon_bgp_asn>\n  <connection_type>private</connection_type>\n</logical_connection>
\n",
  "amazonAddress": "169.254.255.1/30",
  "virtualInterfaceType": "private",
  "virtualInterfaceName": "VIF1A"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateVirtualInterface。

confirm-connection

以下代码示例演示了如何使用 confirm-connection。

Amazon CLI

确认在互连上创建托管连接

以下 confirm-connection 命令确认在互连上创建托管连接：

```
aws directconnect confirm-connection --connection-id dxcon-fg2wi7hy
```

输出：

```
{
    "connectionState": "pending"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmConnection。

confirm-private-virtual-interface

以下代码示例演示了如何使用 confirm-private-virtual-interface。

Amazon CLI

接受私有虚拟接口的所有权

以下 confirm-private-virtual-interface 命令接受其他客户创建的私有虚拟接口的所有权。

```
aws directconnect confirm-private-virtual-interface --virtual-interface-id dxvif-fgy8orxu --virtual-gateway-id vgw-e4a47df9
```

输出：

```
{
    "virtualInterfaceState": "pending"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmPrivateVirtualInterface。

confirm-public-virtual-interface

以下代码示例演示了如何使用 confirm-public-virtual-interface。

Amazon CLI

接受公有虚拟接口的所有权

以下 confirm-public-virtual-interface 命令接受其他客户创建的公有虚拟接口的所有权。

```
aws directconnect confirm-public-virtual-interface --virtual-interface-id dxvif-fg9xo9vp
```

输出：

```
{
    "virtualInterfaceState": "verifying"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmPublicVirtualInterface。

confirm-transit-virtual-interface

以下代码示例演示了如何使用 confirm-transit-virtual-interface。

Amazon CLI

接受中转虚拟接口的所有权

以下 confirm-transit-virtual-interface 接受其他客户创建的中转虚拟接口的所有权。

```
aws directconnect confirm-transit-virtual-interface \
    --virtual-interface-id dxvif-fEXAMPLE \
    --direct-connect-gateway-id 4112ccf9-25e9-4111-8237-b6c5dEXAMPLE
```

输出：

```
{
    "virtualInterfaceState": "pending"
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的接受托管的虚拟接口。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmTransitVirtualInterface。

create-bgp-peer

以下代码示例演示了如何使用 create-bgp-peer。

Amazon CLI

创建 IPv6 BGP 对等会话

以下示例在私有虚拟接口 dxvif-fg1vuj3d 上创建 IPv6 BGP 对等会话。对等 IPv6 地址由 Amazon 自动分配。

命令:

```
aws directconnect create-bgp-peer --virtual-interface-id dxvif-fg1vuj3d --new-bgp-peer asn=64600,addressFamily=ipv6
```

输出：

```
{
  "virtualInterface": {
      "virtualInterfaceState": "available",
      "asn": 65000,
      "vlan": 125,
      "customerAddress": "169.254.255.2/30",
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-fguhmqlc",
      "addressFamily": "ipv4",
      "virtualGatewayId": "vgw-f9eb0c90",
      "virtualInterfaceId": "dxvif-fg1vuj3d",
      "authKey": "0xC_ukbCerl6EYA0example",
      "routeFilterPrefixes": [],
      "location": "EqDC2",
      "bgpPeers": [
          {
              "bgpStatus": "down",
              "customerAddress": "169.254.255.2/30",
              "addressFamily": "ipv4",
              "authKey": "0xC_ukbCerl6EYA0uexample",
              "bgpPeerState": "available",
```

```
                "amazonAddress": "169.254.255.1/30",
                "asn": 65000
            },
            {
                "bgpStatus": "down",
                "customerAddress": "2001:db8:1100:2f0:0:1:9cb4:4216/125",
                "addressFamily": "ipv6",
                "authKey": "0xS27kAIU_VHPjjAexample",
                "bgpPeerState": "pending",
                "amazonAddress": "2001:db8:1100:2f0:0:1:9cb4:4211/125",
                "asn": 64600
            }
        ],
        "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fg1vuj3d\">\n  <vlan>125</
vlan>\n  <customer_address>169.254.255.2/30</customer_address>\n
 <amazon_address>169.254.255.1/30</amazon_address>\n  <bgp_asn>65000</
bgp_asn>\n  <bgp_auth_key>0xC_ukbCerl6EYA0uexample</bgp_auth_key>\n
 <ipv6_customer_address>2001:db8:1100:2f0:0:1:9cb4:4216/125</ipv6_customer_address>
\n  <ipv6_amazon_address>2001:db8:1100:2f0:0:1:9cb4:4211/125</ipv6_amazon_address>\n
  <ipv6_bgp_asn>64600</ipv6_bgp_asn>\n  <ipv6_bgp_auth_key>0xS27kAIU_VHPjjAexample</
ipv6_bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>\n
 <connection_type>private</connection_type>\n</logical_connection>\n",
        "amazonAddress": "169.254.255.1/30",
        "virtualInterfaceType": "private",
        "virtualInterfaceName": "Test"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBgpPeer。

`create-connection`

以下代码示例演示了如何使用 `create-connection`。

Amazon CLI

创建从您的网络到 Amazon Direct Connect 位置的连接

以下 `create-connection` 命令创建从您的网络到 Amazon Direct Connect 位置的连接：

```
aws directconnect create-connection --location TIVIT --bandwidth 1Gbps --connection-
name "Connection to AWS"
```

输出：

```
{
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-fg31dyv6",
    "connectionState": "requested",
    "bandwidth": "1Gbps",
    "location": "TIVIT",
    "connectionName": "Connection to AWS",
    "region": "sa-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnection。

create-direct-connect-gateway-association-proposal

以下代码示例演示了如何使用 create-direct-connect-gateway-association-proposal。

Amazon CLI

创建提议以将指定的中转网关与指定的 Direct Connect 网关关联

以下 create-direct-connect-gateway-association-proposal 示例创建提议以将指定的中转网关与指定的 Direct Connect 网关关联。

```
aws directconnect create-direct-connect-gateway-association-proposal \
    --direct-connect-gateway-id 11460968-4ac1-4fd3-bdb2-00599EXAMPLE \
    --direct-connect-gateway-owner-account 111122223333 \
    --gateway-id tgw-02f776b1a7EXAMPLE \
    --add-allowed-prefixes-to-direct-connect-gateway cidr=192.168.1.0/30
```

输出：

```
{
    "directConnectGatewayAssociationProposal": {
        "proposalId": "cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE",
        "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
        "directConnectGatewayOwnerAccount": "111122223333",
        "proposalState": "requested",
        "associatedGateway": {
            "id": "tgw-02f776b1a7EXAMPLE",
            "type": "transitGateway",
```

```
            "ownerAccount": "111122223333",
            "region": "us-east-1"
        },
        "requestedAllowedPrefixesToDirectConnectGateway": [
            {
                "cidr": "192.168.1.0/30"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的创建 Transit Gateway 关联提议。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDirectConnectGatewayAssociationProposal。

`create-direct-connect-gateway-association`

以下代码示例演示了如何使用 `create-direct-connect-gateway-association`。

Amazon CLI

将虚拟专用网关与 Direct Connect 网关关联

以下示例将虚拟专用网关 `vgw-6efe725e` 与 Direct Connect 网关 `5f294f92-bafb-4011-916d-9b0bexample` 关联。您必须在虚拟专用网关所在的区域中运行该命令。

命令:

```
aws directconnect create-direct-connect-gateway-association --direct-connect-gateway-id 5f294f92-bafb-4011-916d-9b0bexample --virtual-gateway-id vgw-6efe725e
```

输出：

```
{
  "directConnectGatewayAssociation": {
      "associationState": "associating",
      "virtualGatewayOwnerAccount": "123456789012",
      "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
      "virtualGatewayId": "vgw-6efe725e",
      "virtualGatewayRegion": "us-east-2"
```

```
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDirectConnectGatewayAssociation。

create-direct-connect-gateway

以下代码示例演示了如何使用 create-direct-connect-gateway。

Amazon CLI

创建 Direct Connect 网关

以下示例创建名为 DxGateway1 的 Direct Connect 网关。

命令:

```
aws directconnect create-direct-connect-gateway --direct-connect-gateway-
name "DxGateway1"
```

输出：

```
{
  "directConnectGateway": {
      "amazonSideAsn": 64512,
      "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bdexample",
      "ownerAccount": "123456789012",
      "directConnectGatewayName": "DxGateway1",
      "directConnectGatewayState": "available"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDirectConnectGateway。

create-interconnect

以下代码示例演示了如何使用 create-interconnect。

Amazon CLI

在合作伙伴的网络和 Amazon 之间创建互连

以下 `create-interconnect` 命令在 Amazon Direct Connect 合作伙伴网络与指定 Amazon Direct Connect 位置之间创建新互连。

```
aws directconnect create-interconnect --interconnect-name "1G Interconnect to AWS"
 --bandwidth 1Gbps --location TIVIT
```

输出：

```
{
    "region": "sa-east-1",
    "bandwidth": "1Gbps",
    "location": "TIVIT",
    "interconnectName": "1G Interconnect to AWS",
    "interconnectId": "dxcon-fgktov66",
    "interconnectState": "requested"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInterconnect。

create-lag

以下代码示例演示了如何使用 `create-lag`。

Amazon CLI

创建带有新连接的 LAG

以下示例创建一个 LAG，并为 LAG 请求两个带宽为 1 Gbps 的新 Amazon Direct Connect 连接。

命令:

```
aws directconnect create-lag --location CSVA1 --number-of-connections 2 --
connections-bandwidth 1Gbps --lag-name 1GBLag
```

输出：

```
{
  "awsDevice": "CSVA1-23u8tlpaz8iks",
  "numberOfConnections": 2,
  "lagState": "pending",
  "ownerAccount": "123456789012",
```

```
  "lagName": "1GBLag",
  "connections": [
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-ffqr6x5q",
          "lagId": "dxlag-ffjhj9lx",
          "connectionState": "requested",
          "bandwidth": "1Gbps",
          "location": "CSVA1",
          "connectionName": "Requested Connection 1 for Lag dxlag-ffjhj9lx",
          "region": "us-east-1"
      },
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-fflqyj95",
          "lagId": "dxlag-ffjhj9lx",
          "connectionState": "requested",
          "bandwidth": "1Gbps",
          "location": "CSVA1",
          "connectionName": "Requested Connection 2 for Lag dxlag-ffjhj9lx",
          "region": "us-east-1"
      }
  ],
  "lagId": "dxlag-ffjhj9lx",
  "minimumLinks": 0,
  "connectionsBandwidth": "1Gbps",
  "region": "us-east-1",
  "location": "CSVA1"
}
```

使用现有连接创建 LAG

以下示例使用您账户中的现有连接创建 LAG，并请求使用与现有连接相同的带宽和位置为 LAG 创建第二个新连接。

命令:

```
aws directconnect create-lag --location EqDC2 --number-of-connections 2 --
connections-bandwidth 1Gbps --lag-name 2ConnLAG --connection-id dxcon-fgk145dr
```

输出：

```
{
```

```
  "awsDevice": "EqDC2-4h6ce2r1bes6",
  "numberOfConnections": 2,
  "lagState": "pending",
  "ownerAccount": "123456789012",
  "lagName": "2ConnLAG",
  "connections": [
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-fh6ljcvo",
          "lagId": "dxlag-fhccu14t",
          "connectionState": "requested",
          "bandwidth": "1Gbps",
          "location": "EqDC2",
          "connectionName": "Requested Connection 1 for Lag dxlag-fhccu14t",
          "region": "us-east-1"
      },
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-fgk145dr",
          "lagId": "dxlag-fhccu14t",
          "connectionState": "down",
          "bandwidth": "1Gbps",
          "location": "EqDC2",
          "connectionName": "VAConn1",
          "region": "us-east-1"
      }
  ],
  "lagId": "dxlag-fhccu14t",
  "minimumLinks": 0,
  "connectionsBandwidth": "1Gbps",
  "region": "us-east-1",
  "location": "EqDC2"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLag。

create-private-virtual-interface

以下代码示例演示了如何使用 create-private-virtual-interface。

Amazon CLI

创建私有虚拟接口

以下 create-private-virtual-interface 命令创建私有虚拟接口：

```
aws directconnect create-private-virtual-interface --connection-id dxcon-ffjrkx17 --
new-private-virtual-
interface virtualInterfaceName=PrivateVirtualInterface,vlan=101,asn=65000,authKey=asdf34exam
aba37db6
```

输出：

```
{
    "virtualInterfaceState": "pending",
    "asn": 65000,
    "vlan": 101,
    "customerAddress": "192.168.1.2/30",
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffjrkx17",
    "virtualGatewayId": "vgw-aba37db6",
    "virtualInterfaceId": "dxvif-ffhhk74f",
    "authKey": "asdf34example",
    "routeFilterPrefixes": [],
    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-ffhhk74f\">\n  <vlan>101</
vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n
 <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>private</connection_type>\n</logical_connection>\n",
    "amazonAddress": "192.168.1.1/30",
    "virtualInterfaceType": "private",
    "virtualInterfaceName": "PrivateVirtualInterface"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePrivateVirtualInterface。

create-public-virtual-interface

以下代码示例演示了如何使用 create-public-virtual-interface。

Amazon CLI

创建公有虚拟接口

以下 create-public-virtual-interface 命令创建公有虚拟接口：

```
aws directconnect create-public-virtual-interface --connection-id dxcon-ffjrkx17 --
new-public-virtual-
interface virtualInterfaceName=PublicVirtualInterface,vlan=2000,asn=65000,authKey=asdf34exam
{cidr=203.0.113.4/30}]
```

输出：

```
{
    "virtualInterfaceState": "verifying",
    "asn": 65000,
    "vlan": 2000,
    "customerAddress": "203.0.113.2/30",
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-ffjrkx17",
    "virtualInterfaceId": "dxvif-fgh0hcrk",
    "authKey": "asdf34example",
    "routeFilterPrefixes": [
        {
            "cidr": "203.0.113.0/30"
        },
        {
            "cidr": "203.0.113.4/30"
        }
    ],
    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fgh0hcrk\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
 <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
    "amazonAddress": "203.0.113.1/30",
    "virtualInterfaceType": "public",
    "virtualInterfaceName": "PublicVirtualInterface"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePublicVirtualInterface。

create-transit-virtual-interface

以下代码示例演示了如何使用 create-transit-virtual-interface。

Amazon CLI

创建中转虚拟接口

以下 create-transit-virtual-interface 示例为指定连接创建中转虚拟接口。

```
aws directconnect create-transit-virtual-interface \
    --connection-id dxlag-fEXAMPLE \
    --new-transit-virtual-interface "virtualInterfaceName=Example Transit Virtual
 Interface,vlan=126,asn=65110,mtu=1500,authKey=0xzxgA9YoW9h58u8SvEXAMPLE,amazonAddress=192.1
aada-5a1baEXAMPLE,tags=[{key=Tag,value=Example}]"
```

输出：

```
{
    "virtualInterface": {
        "ownerAccount": "1111222233333",
        "virtualInterfaceId": "dxvif-fEXAMPLE",
        "location": "loc1",
        "connectionId": "dxlag-fEXAMPLE",
        "virtualInterfaceType": "transit",
        "virtualInterfaceName": "Example Transit Virtual Interface",
        "vlan": 126,
        "asn": 65110,
        "amazonSideAsn": 4200000000,
        "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
        "amazonAddress": "192.168.1.1/30",
        "customerAddress": "192.168.1.2/30",
        "addressFamily": "ipv4",
        "virtualInterfaceState": "pending",
        "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fEXAMPLE\">\n  <vlan>126</
vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n
 <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65110</
bgp_asn>\n  <bgp_auth_key>0xzxgA9YoW9h58u8SvOmXRTw</bgp_auth_key>\n
 <amazon_bgp_asn>4200000000</amazon_bgp_asn>\n  <connection_type>transit</
connection_type>\n</logical_connection>\n",
        "mtu": 1500,
        "jumboFrameCapable": true,
```

```
        "virtualGatewayId": "",
        "directConnectGatewayId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
        "routeFilterPrefixes": [],
        "bgpPeers": [
            {
                "bgpPeerId": "dxpeer-EXAMPLE",
                "asn": 65110,
                "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
                "addressFamily": "ipv4",
                "amazonAddress": "192.168.1.1/30",
                "customerAddress": "192.168.1.2/30",
                "bgpPeerState": "pending",
                "bgpStatus": "down",
                "awsDeviceV2": "loc1-26wz6vEXAMPLE"
            }
        ],
        "region": "sa-east-1",
        "awsDeviceV2": "loc1-26wz6vEXAMPLE",
        "tags": [
            {
                "key": "Tag",
                "value": "Example"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的创建 Direct Connect 网关的中转虚拟接口。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitVirtualInterface。

`delete-bgp-peer`

以下代码示例演示了如何使用 `delete-bgp-peer`。

Amazon CLI

从虚拟接口删除 BGP 对等体

以下示例从虚拟接口 `dxvif-fg1vuj3d` 删除 IPv6 BGP 对等体。

命令:

```
aws directconnect delete-bgp-peer --virtual-interface-id dxvif-fg1vuj3d --asn 64600
 --customer-address 2001:db8:1100:2f0:0:1:9cb4:4216/125
```

输出：

```
{
  "virtualInterface": {
      "virtualInterfaceState": "available",
      "asn": 65000,
      "vlan": 125,
      "customerAddress": "169.254.255.2/30",
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-fguhmqlc",
      "addressFamily": "ipv4",
      "virtualGatewayId": "vgw-f9eb0c90",
      "virtualInterfaceId": "dxvif-fg1vuj3d",
      "authKey": "0xC_ukbCerl6EYA0example",
      "routeFilterPrefixes": [],
      "location": "EqDC2",
      "bgpPeers": [
          {
              "bgpStatus": "down",
              "customerAddress": "169.254.255.2/30",
              "addressFamily": "ipv4",
              "authKey": "0xC_ukbCerl6EYA0uexample",
              "bgpPeerState": "available",
              "amazonAddress": "169.254.255.1/30",
              "asn": 65000
          },
          {
              "bgpStatus": "down",
              "customerAddress": "2001:db8:1100:2f0:0:1:9cb4:4216/125",
              "addressFamily": "ipv6",
              "authKey": "0xS27kAIU_VHPjjAexample",
              "bgpPeerState": "deleting",
              "amazonAddress": "2001:db8:1100:2f0:0:1:9cb4:4211/125",
              "asn": 64600
          }
      ],
      "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fg1vuj3d\">\n  <vlan>125</
vlan>\n  <customer_address>169.254.255.2/30</customer_address>\n
 <amazon_address>169.254.255.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
```

```
  <bgp_auth_key>0xC_ukbCerl6EYA0example</bgp_auth_key>\n  <amazon_bgp_asn>7224</
amazon_bgp_asn>\n  <connection_type>private</connection_type>\n</logical_connection>
\n",
      "amazonAddress": "169.254.255.1/30",
      "virtualInterfaceType": "private",
      "virtualInterfaceName": "Test"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBgpPeer。

delete-connection

以下代码示例演示了如何使用 delete-connection。

Amazon CLI

删除连接

以下 delete-connection 命令删除指定的连接。

```
aws directconnect delete-connection --connection-id dxcon-fg31dyv6
```

输出：

```
{
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-fg31dyv6",
    "connectionState": "deleted",
    "bandwidth": "1Gbps",
    "location": "TIVIT",
    "connectionName": "Connection to AWS",
    "region": "sa-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnection。

delete-direct-connect-gateway-association

以下代码示例演示了如何使用 delete-direct-connect-gateway-association。

Amazon CLI

删除 Direct Connect 网关关联

以下 `delete-direct-connect-gateway-association` 示例删除与具有指定关联 ID 的中转网关的 Direct Connect 网关关联。

```
aws directconnect delete-direct-connect-gateway-association --association-id
  be85116d-46eb-4b43-a27a-da0c2ad648de
```

输出：

```
{
    "directConnectGatewayAssociation": {
        "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPlE",
        "directConnectGatewayOwnerAccount": "123456789012",
        "associationState": "disassociating",
        "associatedGateway": {
            "id": "tgw-095b3b0b54EXAMPLE",
            "type": "transitGateway",
            "ownerAccount": "123456789012",
            "region": "us-east-1"
       },
        "associationId": " be85116d-46eb-4b43-a27a-da0c2ad648deEXAMPLE ",
        "allowedPrefixesToDirectConnectGateway": [
            {
                "cidr": "192.0.1.0/28"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的关联和接触关联 Transit Gateway。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDirectConnectGatewayAssociation。

`delete-direct-connect-gateway`

以下代码示例演示了如何使用 `delete-direct-connect-gateway`。

Amazon CLI

删除 Direct Connect 网关

以下示例删除 Direct Connect 网关 5f294f92-bafb-4011-916d-9b0bexample。

命令:

```
aws directconnect delete-direct-connect-gateway --direct-connect-gateway-
id 5f294f92-bafb-4011-916d-9b0bexample
```

输出：

```
{
  "directConnectGateway": {
      "amazonSideAsn": 64512,
      "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
      "ownerAccount": "123456789012",
      "directConnectGatewayName": "DxGateway1",
      "directConnectGatewayState": "deleting"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDirectConnectGateway。

delete-interconnect

以下代码示例演示了如何使用 delete-interconnect。

Amazon CLI

删除互连

以下 delete-interconnect 命令删除指定的互连。

```
aws directconnect delete-interconnect --interconnect-id dxcon-fgktov66
```

输出：

```
{
```

```
    "interconnectState": "deleted"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInterconnect。

delete-lag

以下代码示例演示了如何使用 delete-lag。

Amazon CLI

删除 LAG

以下示例删除指定 LAG。

命令:

```
aws directconnect delete-lag --lag-id dxlag-ffrhowd9
```

输出：

```
{
  "awsDevice": "EqDC2-4h6ce2r1bes6",
  "numberOfConnections": 0,
  "lagState": "deleted",
  "ownerAccount": "123456789012",
  "lagName": "TestLAG",
  "connections": [],
  "lagId": "dxlag-ffrhowd9",
  "minimumLinks": 0,
  "connectionsBandwidth": "1Gbps",
  "region": "us-east-1",
  "location": "EqDC2"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLag。

delete-virtual-interface

以下代码示例演示了如何使用 delete-virtual-interface。

Amazon CLI

删除虚拟接口

以下 delete-virtual-interface 命令删除指定的虚拟接口：

```
aws directconnect delete-virtual-interface --virtual-interface-id dxvif-ffhhk74f
```

输出：

```
{
    "virtualInterfaceState": "deleting"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVirtualInterface。

describe-connection-loa

以下代码示例演示了如何使用 describe-connection-loa。

Amazon CLI

描述使用 Linux 或 Mac OS X 连接的 LOA-CFA

以下示例描述用于连接 dxcon-fh6ayh1d 的 LOA-CFA。LOA-CFA 的内容经过 base64 编码。此命令使用 --output 和 --query 参数来控制输出并提取 loaContent 结构的内容。命令的最后一部分使用 base64 实用工具解码内容并将输出发送到 PDF 文件。

```
aws directconnect describe-connection-loa --connection-id dxcon-fh6ayh1d --
output text --query loa.loaContent|base64 --decode > myLoaCfa.pdf
```

描述使用 Windows 连接的 LOA-CFA

前面的示例需要使用 base64 实用程序解码输出。在 Windows 电脑上，您可以改用 certutil。在以下示例中，第一个命令描述了用于连接 dxcon-fh6ayh1d 的 LOA-CFA，并使用 --output 和 --query 参数控制输出，将 loaContent 结构的内容提取到名为 myLoaCfa.base64 的文件中。第二个命令使用 certutil 实用工具解码文件并将输出发送到 PDF 文件。

```
aws directconnect describe-connection-loa --connection-id dxcon-fh6ayh1d --
output text --query loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

有关控制 Amazon CLI 输出的更多信息，请参阅《Amazon Command Line Interface 用户指南》中的从 Amazon Command Line Interface 控制命令输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConnectionLoa。

`describe-connections-on-interconnect`

以下代码示例演示了如何使用 `describe-connections-on-interconnect`。

Amazon CLI

列出互连上的连接

以下 `describe-connections-on-interconnect` 命令列出给定互连中已预置的连接：

```
aws directconnect describe-connections-on-interconnect --interconnect-id dxcon-fgktov66
```

输出：

```
{
    "connections": [
        {
            "partnerName": "TIVIT",
            "vlan": 101,
            "ownerAccount": "123456789012",
            "connectionId": "dxcon-ffzc51m1",
            "connectionState": "ordering",
            "bandwidth": "500Mbps",
            "location": "TIVIT",
            "connectionName": "mydcinterconnect",
            "region": "sa-east-1"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConnectionsOnInterconnect。

describe-connections

以下代码示例演示了如何使用 describe-connections。

Amazon CLI

列出当前区域的所有连接

以下 describe-connections 命令列出当前区域中的所有连接：

```
aws directconnect describe-connections
```

输出：

```
{
  "connections": [
      {
          "awsDevice": "EqDC2-123h49s71dabc",
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-fguhmqlc",
          "lagId": "dxlag-ffrz71kw",
          "connectionState": "down",
          "bandwidth": "1Gbps",
          "location": "EqDC2",
          "connectionName": "My_Connection",
          "loaIssueTime": 1491568964.0,
          "region": "us-east-1"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConnections。

describe-direct-connect-gateway-association-proposals

以下代码示例演示了如何使用 describe-direct-connect-gateway-association-proposals。

Amazon CLI

描述您的 Direct Connect 网关关联提议

以下 describe-direct-connect-gateway-association-proposals 示例显示有关您 Direct Connect 网关关联提议的详细信息。

```
aws directconnect describe-direct-connect-gateway-association-proposals
```

输出：

```
{
    "directConnectGatewayAssociationProposals": [
        {
            "proposalId": "c2ede9b4-bbc6-4d33-923c-bc4feEXAMPLE",
            "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
            "directConnectGatewayOwnerAccount": "111122223333",
            "proposalState": "requested",
            "associatedGateway": {
                "id": "tgw-02f776b1a7EXAMPLE",
                "type": "transitGateway",
                "ownerAccount": "111122223333",
                "region": "us-east-1"
            },
            "existingAllowedPrefixesToDirectConnectGateway": [
                {
                    "cidr": "192.168.2.0/30"
                },
                {
                    "cidr": "192.168.1.0/30"
                }
            ],
            "requestedAllowedPrefixesToDirectConnectGateway": [
                {
                    "cidr": "192.168.1.0/30"
                }
            ]
        },
        {
            "proposalId": "cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE",
            "directConnectGatewayId": "11560968-4ac1-4fd3-bcb2-00599EXAMPLE",
            "directConnectGatewayOwnerAccount": "111122223333",
            "proposalState": "accepted",
            "associatedGateway": {
                "id": "tgw-045776b1a7EXAMPLE",
                "type": "transitGateway",
                "ownerAccount": "111122223333",
```

```
                "region": "us-east-1"
            },
            "existingAllowedPrefixesToDirectConnectGateway": [
                {
                    "cidr": "192.168.4.0/30"
                },
                {
                    "cidr": "192.168.5.0/30"
                }
            ],
            "requestedAllowedPrefixesToDirectConnectGateway": [
                {
                    "cidr": "192.168.5.0/30"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的关联和接触关联 Transit Gateway。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDirectConnectGatewayAssociationProposals。

`describe-direct-connect-gateway-associations`

以下代码示例演示了如何使用 `describe-direct-connect-gateway-associations`。

Amazon CLI

描述 Direct Connect 网关关联

以下示例描述 Direct Connect 网关 `5f294f92-bafb-4011-916d-9b0bexample` 的所有关联。

命令:

```
aws directconnect describe-direct-connect-gateway-associations --direct-connect-gateway-id 5f294f92-bafb-4011-916d-9b0bexample
```

输出：

```
{
```

```
  "nextToken":
 "eyJ2IjoxLCJzIjoxLCJpIjoiOU83OTFodzdycnZCbkN4MExHeHVwQT09IiwiYyI6InIxTEN0UEVHV0I1UFlkaWFnNl
  "directConnectGatewayAssociations": [
      {
          "associationState": "associating",
          "virtualGatewayOwnerAccount": "123456789012",
          "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
          "virtualGatewayId": "vgw-6efe725e",
          "virtualGatewayRegion": "us-east-2"
      },
      {
          "associationState": "disassociating",
          "virtualGatewayOwnerAccount": "123456789012",
          "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
          "virtualGatewayId": "vgw-ebaa27db",
          "virtualGatewayRegion": "us-east-2"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDirectConnectGatewayAssociations。

describe-direct-connect-gateway-attachments

以下代码示例演示了如何使用 describe-direct-connect-gateway-attachments。

Amazon CLI

描述 Direct Connect 网关附件

以下示例描述附加到 Direct Connect 网关 5f294f92-bafb-4011-916d-9b0bexample 的虚拟接口。

命令:

```
aws directconnect describe-direct-connect-gateway-attachments --direct-connect-
gateway-id 5f294f92-bafb-4011-916d-9b0bexample
```

输出：

```
{
```

```
  "directConnectGatewayAttachments": [
      {
          "virtualInterfaceOwnerAccount": "123456789012",
          "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
          "virtualInterfaceRegion": "us-east-2",
          "attachmentState": "attaching",
          "virtualInterfaceId": "dxvif-fg9zyabc"
      }
  ],
  "nextToken":
 "eyJ2IjoxLCJzIjoxLCJpIjoibEhXdlNpUXF5RzhoL1JyUW52SlV2QT09IiwiYyI6Im5wQjFHQ0RyQUdRS3puNnNXcU
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDirectConnectGatewayAttachments。

describe-direct-connect-gateways

以下代码示例演示了如何使用 describe-direct-connect-gateways。

Amazon CLI

描述您的 Direct Connect 网关

以下示例描述您的所有 Direct Connect 网关。

命令:

```
aws directconnect describe-direct-connect-gateways
```

输出：

```
{
  "directConnectGateways": [
      {
          "amazonSideAsn": 64512,
          "directConnectGatewayId": "cf68415c-f4ae-48f2-87a7-3b52cexample",
          "ownerAccount": "123456789012",
          "directConnectGatewayName": "DxGateway2",
          "directConnectGatewayState": "available"
      },
```

```
        {
            "amazonSideAsn": 64512,
            "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bdexample",
            "ownerAccount": "123456789012",
            "directConnectGatewayName": "DxGateway1",
            "directConnectGatewayState": "available"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDirectConnectGateways。

describe-hosted-connections

以下代码示例演示了如何使用 describe-hosted-connections。

Amazon CLI

列出互连上的连接

以下示例列出已在给定互连上预置的连接列表。

命令:

```
aws directconnect describe-hosted-connections --connection-id dxcon-fgktov66
```

输出：

```
{
    "connections": [
        {
            "partnerName": "TIVIT",
            "vlan": 101,
            "ownerAccount": "123456789012",
            "connectionId": "dxcon-ffzc51m1",
            "connectionState": "ordering",
            "bandwidth": "500Mbps",
            "location": "TIVIT",
            "connectionName": "mydcinterconnect",
            "region": "sa-east-1"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHostedConnections。

describe-interconnect-loa

以下代码示例演示了如何使用 describe-interconnect-loa。

Amazon CLI

描述使用 Linux 或 Mac OS X 进行互连的 LOA-CFA

以下示例描述用于互连 dxcon-fh6ayh1d 的 LOA-CFA。LOA-CFA 的内容经过 base64 编码。此命令使用 --output 和 --query 参数来控制输出并提取 loaContent 结构的内容。命令的最后一部分使用 base64 实用工具解码内容并将输出发送到 PDF 文件。

```
aws directconnect describe-interconnect-loa --interconnect-id dxcon-fh6ayh1d --
output text --query loa.loaContent|base64 --decode > myLoaCfa.pdf
```

描述使用 Windows 进行互连的 LOA-CFA

前面的示例需要使用 base64 实用程序解码输出。在 Windows 电脑上，您可以改用 certutil。在以下示例中，第一个命令描述了用于互连 dxcon-fh6ayh1d 的 LOA-CFA，并使用 --output 和 --query 参数控制输出，将 loaContent 结构的内容提取到名为 myLoaCfa.base64 的文件中。第二个命令使用 certutil 实用工具解码文件并将输出发送到 PDF 文件。

```
aws directconnect describe-interconnect-loa --interconnect-id dxcon-fh6ayh1d --
output text --query loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

有关控制 Amazon CLI 输出的更多信息，请参阅《Amazon Command Line Interface 用户指南》中的从 Amazon Command Line Interface 控制命令输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInterconnectLoa。

describe-interconnects

以下代码示例演示了如何使用 describe-interconnects。

Amazon CLI

列出互连

以下 describe-interconnects 命令列出您 Amazon 账户拥有的互连：

```
aws directconnect describe-interconnects
```

输出：

```
{
    "interconnects": [
        {
            "region": "sa-east-1",
            "bandwidth": "1Gbps",
            "location": "TIVIT",
            "interconnectName": "1G Interconnect to AWS",
            "interconnectId": "dxcon-fgktov66",
            "interconnectState": "down"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInterconnects。

describe-lags

以下代码示例演示了如何使用 describe-lags。

Amazon CLI

描述您的 LAG

以下命令描述当前区域的所有 LAG。

命令:

```
aws directconnect describe-lags
```

输出：

```
{
```

```
  "lags": [
      {
          "awsDevice": "EqDC2-19y7z3m17xpuz",
          "numberOfConnections": 2,
          "lagState": "down",
          "ownerAccount": "123456789012",
          "lagName": "DA-LAG",
          "connections": [
              {
                  "ownerAccount": "123456789012",
                  "connectionId": "dxcon-ffnikghc",
                  "lagId": "dxlag-fgsu9erb",
                  "connectionState": "requested",
                  "bandwidth": "10Gbps",
                  "location": "EqDC2",
                  "connectionName": "Requested Connection 1 for Lag dxlag-fgsu9erb",
                  "region": "us-east-1"
              },
              {
                  "ownerAccount": "123456789012",
                  "connectionId": "dxcon-fglgbdea",
                  "lagId": "dxlag-fgsu9erb",
                  "connectionState": "requested",
                  "bandwidth": "10Gbps",
                  "location": "EqDC2",
                  "connectionName": "Requested Connection 2 for Lag dxlag-fgsu9erb",
                  "region": "us-east-1"
              }
          ],
          "lagId": "dxlag-fgsu9erb",
          "minimumLinks": 0,
          "connectionsBandwidth": "10Gbps",
          "region": "us-east-1",
          "location": "EqDC2"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLags。

describe-loa

以下代码示例演示了如何使用 describe-loa。

Amazon CLI

描述使用 Linux 或 Mac OS X 连接的 LOA-CFA

以下示例描述用于连接 dxcon-fh6ayh1d 的 LOA-CFA。LOA-CFA 的内容经过 base64 编码。此命令使用 --output 和 --query 参数来控制输出并提取 loaContent 结构的内容。命令的最后一部分使用 base64 实用工具解码内容并将输出发送到 PDF 文件。

```
aws directconnect describe-loa --connection-id dxcon-fh6ayh1d --output text --
query loa.loaContent|base64 --decode > myLoaCfa.pdf
```

描述使用 Windows 连接的 LOA-CFA

前面的示例需要使用 base64 实用程序解码输出。在 Windows 电脑上，您可以改用 certutil。在以下示例中，第一个命令描述了用于连接 dxcon-fh6ayh1d 的 LOA-CFA，并使用 --output 和 --query 参数控制输出，将 loaContent 结构的内容提取到名为 myLoaCfa.base64 的文件中。第二个命令使用 certutil 实用工具解码文件并将输出发送到 PDF 文件。

```
aws directconnect describe-loa --connection-id dxcon-fh6ayh1d --output text --
query loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

有关控制 Amazon CLI 输出的更多信息，请参阅《Amazon Command Line Interface 用户指南》中的从 Amazon Command Line Interface 控制命令输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoa。

describe-locations

以下代码示例演示了如何使用 describe-locations。

Amazon CLI

列出 Amazon Direct Connect 合作伙伴和位置

以下 describe-locations 命令列出当前区域的 Amazon Direct Connect 合作伙伴和位置：

```
aws directconnect describe-locations
```

输出：

```
{
    "locations": [
        {
            "locationName": "NAP do Brasil, Barueri, Sao Paulo",
            "locationCode": "TNDB"
        },
        {
            "locationName": "Tivit - Site Transamerica (Sao Paulo)",
            "locationCode": "TIVIT"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocations。

describe-tags

以下代码示例演示了如何使用 describe-tags。

Amazon CLI

描述您 Amazon Direct Connect 资源的标签

以下命令描述连接 dxcon-abcabc12 的标签。

命令:

```
aws directconnect describe-tags --resource-arns arn:aws:directconnect:us-
east-1:123456789012:dxcon/dxcon-abcabc12
```

输出：

```
{
  "resourceTags": [
      {
          "resourceArn": "arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-
abcabc12",
          "tags": [
              {
                  "value": "VAConnection",
```

```
                "key": "Name"
            }
        ]
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

describe-virtual-gateways

以下代码示例演示了如何使用 describe-virtual-gateways。

Amazon CLI

列出虚拟专用网关

以下 describe-virtual-gateways 命令列出您 Amazon 账户拥有的虚拟专用网关：

```
aws directconnect describe-virtual-gateways
```

输出：

```
{
    "virtualGateways": [
        {
            "virtualGatewayId": "vgw-aba37db6",
            "virtualGatewayState": "available"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVirtualGateways。

describe-virtual-interfaces

以下代码示例演示了如何使用 describe-virtual-interfaces。

Amazon CLI

列出所有虚拟接口

以下 describe-virtual-interfaces 命令列出与您 Amazon 账户关联的所有虚拟接口的相关信息：

```
aws directconnect describe-virtual-interfaces --connection-id dxcon-ffjrkx17
```

输出：

```
{
    "virtualInterfaces": [
        {
            "virtualInterfaceState": "down",
            "asn": 65000,
            "vlan": 101,
            "customerAddress": "192.168.1.2/30",
            "ownerAccount": "123456789012",
            "connectionId": "dxcon-ffjrkx17",
            "virtualGatewayId": "vgw-aba37db6",
            "virtualInterfaceId": "dxvif-ffhhk74f",
            "authKey": "asdf34example",
            "routeFilterPrefixes": [],
            "location": "TIVIT",
            "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-ffhhk74f\">\n  <vlan>101</
vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n
 <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>private</connection_type>\n</logical_connection>\n",
            "amazonAddress": "192.168.1.1/30",
            "virtualInterfaceType": "private",
            "virtualInterfaceName": "PrivateVirtualInterface"
        },
        {
            "virtualInterfaceState": "verifying",
            "asn": 65000,
            "vlan": 2000,
            "customerAddress": "203.0.113.2/30",
            "ownerAccount": "123456789012",
            "connectionId": "dxcon-ffjrkx17",
            "virtualGatewayId": "",
            "virtualInterfaceId": "dxvif-fgh0hcrk",
            "authKey": "asdf34example",
            "routeFilterPrefixes": [
                {
```

```
                    "cidr": "203.0.113.4/30"
                },
                {
                    "cidr": "203.0.113.0/30"
                }
            ],
            "location": "TIVIT",
            "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fgh0hcrk\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
 <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
 <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
            "amazonAddress": "203.0.113.1/30",
            "virtualInterfaceType": "public",
            "virtualInterfaceName": "PublicVirtualInterface"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVirtualInterfaces。

`disassociate-connection-from-lag`

以下代码示例演示了如何使用 `disassociate-connection-from-lag`。

Amazon CLI

解除连接与 LAG 的关联

以下示例解除指定连接与指定 LAG 的关联。

命令:

```
aws directconnect disassociate-connection-from-lag --lag-id dxlag-fhccu14t --
connection-id  dxcon-fg9607vm
```

输出：

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg9607vm",
  "connectionState": "requested",
```

```
  "bandwidth": "1Gbps",
  "location": "EqDC2",
  "connectionName": "Con2ForLag",
  "region": "us-east-1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateConnectionFromLag。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向 Amazon Direct Connect 资源添加标签

以下命令向连接 dxcon-abcabc12 中添加一个键为 Name、值为 VAConnection 的标签。如果命令成功，则不返回任何输出。

命令:

```
aws directconnect tag-resource --resource-arn arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12 --tags "key=Name,value=VAConnection"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从 Amazon Direct Connect 资源中移除标签

以下命令从连接 dxcon-abcabc12 中移除具有键 Name 的标签。如果命令成功，则不返回任何输出。

命令:

```
aws directconnect untag-resource --resource-arn arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12 --tag-keys Name
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-direct-connect-gateway-association

以下代码示例演示了如何使用 update-direct-connect-gateway-association。

Amazon CLI

更新 Direct Connect 网关关联的指定属性

以下 update-direct-connect-gateway-association 示例将指定的 CIDR 块添加到 Direct Connect 网关关联。

```
aws directconnect update-direct-connect-gateway-association \
    --association-id 820a6e4f-5374-4004-8317-3f64bEXAMPLE \
    --add-allowed-prefixes-to-direct-connect-gateway cidr=192.168.2.0/30
```

输出：

```
{
    "directConnectGatewayAssociation": {
        "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
        "directConnectGatewayOwnerAccount": "111122223333",
        "associationState": "updating",
        "associatedGateway": {
            "id": "tgw-02f776b1a7EXAMPLE",
            "type": "transitGateway",
            "ownerAccount": "111122223333",
            "region": "us-east-1"
        },
        "associationId": "820a6e4f-5374-4004-8317-3f64bEXAMPLE",
        "allowedPrefixesToDirectConnectGateway": [
            {
                "cidr": "192.168.2.0/30"
            },
            {
                "cidr": "192.168.1.0/30"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的使用 Direct Connect 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDirectConnectGatewayAssociation。

update-lag

以下代码示例演示了如何使用 update-lag。

Amazon CLI

更新 LAG

以下示例更改指定 LAG 的名称。

命令:

```
aws directconnect update-lag --lag-id dxlag-ffjhj9lx --lag-name 2ConnLag
```

输出：

```
{
  "awsDevice": "CSVA1-23u8tlpaz8iks",
  "numberOfConnections": 2,
  "lagState": "down",
  "ownerAccount": "123456789012",
  "lagName": "2ConnLag",
  "connections": [
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-fflqyj95",
          "lagId": "dxlag-ffjhj9lx",
          "connectionState": "requested",
          "bandwidth": "1Gbps",
          "location": "CSVA1",
          "connectionName": "Requested Connection 2 for Lag dxlag-ffjhj9lx",
          "region": "us-east-1"
      },
      {
          "ownerAccount": "123456789012",
          "connectionId": "dxcon-ffqr6x5q",
          "lagId": "dxlag-ffjhj9lx",
          "connectionState": "requested",
```

```
            "bandwidth": "1Gbps",
            "location": "CSVA1",
            "connectionName": "Requested Connection 1 for Lag dxlag-ffjhj9lx",
            "region": "us-east-1"
        }
    ],
    "lagId": "dxlag-ffjhj9lx",
    "minimumLinks": 0,
    "connectionsBandwidth": "1Gbps",
    "region": "us-east-1",
    "location": "CSVA1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLag。

update-virtual-interface-attributes

以下代码示例演示了如何使用 update-virtual-interface-attributes。

Amazon CLI

更新虚拟接口的 MTU

以下 update-virtual-interface-attributes 示例更新指定虚拟接口的 MTU。

```
aws directconnect update-virtual-interface-attributes \
    --virtual-interface-id dxvif-fEXAMPLE \
    --mtu 1500
```

输出：

```
{
    "ownerAccount": "1111222233333",
    "virtualInterfaceId": "dxvif-fEXAMPLE",
    "location": "loc1",
    "connectionId": "dxlag-fEXAMPLE",
    "virtualInterfaceType": "transit",
    "virtualInterfaceName": "example transit virtual interface",
    "vlan": 125,
    "asn": 650001,
    "amazonSideAsn": 64512,
    "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
```

```
    "amazonAddress": "169.254.248.1/30",
    "customerAddress": "169.254.248.2/30",
    "addressFamily": "ipv4",
    "virtualInterfaceState": "down",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fEXAMPLE\">\n  <vlan>125</vlan>
\n  <customer_address>169.254.248.2/30</customer_address>\n
 <amazon_address>169.254.248.1/30</amazon_address>\n  <bgp_asn>650001</bgp_asn>\n
  <bgp_auth_key>0xzxgA9YoW9h58u8SEXAMPLE</bgp_auth_key>\n  <amazon_bgp_asn>64512</
amazon_bgp_asn>\n  <connection_type>transit</connection_type>\n</logical_connection>
\n",
    "mtu": 1500,
    "jumboFrameCapable": true,
    "virtualGatewayId": "",
    "directConnectGatewayId": "879b76a1-403d-4700-8b53-4a56ed85436e",
    "routeFilterPrefixes": [],
    "bgpPeers": [
        {
            "bgpPeerId": "dxpeer-fEXAMPLE",
            "asn": 650001,
            "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
            "addressFamily": "ipv4",
            "amazonAddress": "169.254.248.1/30",
            "customerAddress": "169.254.248.2/30",
            "bgpPeerState": "available",
            "bgpStatus": "down",
            "awsDeviceV2": "loc1-26wz6vEXAMPLE"
        }
    ],
    "region": "sa-east-1",
    "awsDeviceV2": "loc1-26wz6vEXAMPLE",
    "tags": []
}
```

有关更多信息，请参阅《Amazon Direct Connect 用户指南》中的为私有虚拟接口或中转虚拟接口设置网络 MTU。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVirtualInterfaceAttributes。

Amazon Directory Service使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Directory Service 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-directories`

以下代码示例演示了如何使用 `describe-directories`。

Amazon CLI

获取有关目录的详细信息

以下 `describe-directories` 示例显示指定目录的详细信息。

```
aws ds describe-directories \
   --directory-id d-a1b2c3d4e5
```

输出：

```
{
    "DirectoryDescriptions": [
        {
            "DirectoryId": "d-a1b2c3d4e5",
            "Name": "mydirectory.example.com",
            "ShortName": "mydirectory",
            "Size": "Small",
            "Edition": "Standard",
            "Alias": "d-a1b2c3d4e5",
            "AccessUrl": "d-a1b2c3d4e5.awsapps.com",
            "Stage": "Active",
            "ShareStatus": "Shared",
            "ShareMethod": "HANDSHAKE",
            "ShareNotes": "These are my share notes",
```

```
            "LaunchTime": "2019-07-08T15:33:46.327000-07:00",
            "StageLastUpdatedDateTime": "2019-07-08T15:59:12.307000-07:00",
            "Type": "SharedMicrosoftAD",
            "SsoEnabled": false,
            "DesiredNumberOfDomainControllers": 0,
            "OwnerDirectoryDescription": {
                "DirectoryId": "d-b2c3d4e5f6",
                "AccountId": "123456789111",
                "DnsIpAddrs": [
                    "203.113.0.248",
                    "203.113.0.253"
                ],
                "VpcSettings": {
                    "VpcId": "vpc-a1b2c3d4",
                    "SubnetIds": [
                        "subnet-a1b2c3d4",
                        "subnet-d4c3b2a1"
                    ],
                    "AvailabilityZones": [
                        "us-west-2a",
                        "us-west-2c"
                    ]
                }
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDirectories。

describe-trusts

以下代码示例演示了如何使用 describe-trusts。

Amazon CLI

获取有关您信任关系的详细信息

以下 describe-trusts 示例显示指定目录的信任关系的详细信息。

```
aws ds describe-trusts \
   --directory-id d-a1b2c3d4e5
```

输出：

```
{
    "Trusts": [
        {
            "DirectoryId": "d-a1b2c3d4e5",
            "TrustId": "t-9a8b7c6d5e",
            "RemoteDomainName": "other.example.com",
            "TrustType": "Forest",
            "TrustDirection": "Two-Way",
            "TrustState": "Verified",
            "CreatedDateTime": "2017-06-20T18:08:45.614000-07:00",
            "LastUpdatedDateTime": "2019-06-04T10:52:12.410000-07:00",
            "StateLastUpdatedDateTime": "2019-06-04T10:52:12.410000-07:00",
            "SelectiveAuth": "Disabled"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrusts。

使用 Amazon CLI 的 Amazon Directory Service 数据示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 Amazon Directory Service 数据结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-group-member`

以下代码示例演示了如何使用 `add-group-member`。

Amazon CLI

向目录中添加组成员

以下 add-group-member 示例向指定目录中的指定组添加指定用户。

```
aws ds-data add-group-member \
    --directory-id d-1234567890 \
    --group-name 'sales' \
    --member-name 'john.doe'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Adding or removing Amazon Managed Microsoft AD members to groups and groups to groups。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AddGroupMember。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

列出可用的小部件

以下 create-group 示例在指定的目录中创建组。

```
aws ds-data create-group \
    --directory-id d-1234567890 \
    --sam-account-name "sales"
```

输出：

```
{
    "DirectoryId": "d-1234567890",
    "SAMAccountName": "sales",
    "SID": "S-1-2-34-5567891234-5678912345-67891234567-8912"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Creating an Amazon Managed Microsoft AD group。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

创建用户

以下 create-user 示例在指定目录中创建用户。

```
aws ds-data create-user \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe'
```

输出：

```
{
    "DirectoryId": "d-1234567890",
    "SAMAccountName": "john.doe",
    "SID": "S-1-2-34-5567891234-5678912345-67891234567-8912"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Creating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除组

以下 delete-group 示例从指定目录中删除指定组。

```
aws ds-data delete-group \
    --directory-id d-1234567890 \
    --sam-account-name 'sales'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Deleting an Amazon Managed Microsoft AD group。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

以下 delete-user 示例从指定目录中删除指定用户。

```
aws ds-data delete-user \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Deleting an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-group

以下代码示例演示了如何使用 describe-group。

Amazon CLI

列出组的详细信息

以下 describe-group 示例获取指定目录中的指定组的信息。

```
aws ds-data describe-group \
    --directory-id d-1234567890 \
    --sam-account-name 'sales'
```

输出：

```
{
    "DirectoryId": "d-1234567890",
    "DistinguishedName": "CN=sales,OU=Users,OU=CORP,DC=corp,DC=example,DC=com",
    "GroupScope": "Global",
    "GroupType": "Security",
    "Realm": "corp.example.com",
    "SAMAccountName": "sales",
    "SID": "S-1-2-34-5567891234-5678912345-67891234567-8912"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD group's details。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGroup。

describe-user

以下代码示例演示了如何使用 describe-user。

Amazon CLI

列出用户的信息

以下 describe-user 示例获取指定目录中的指定用户的信息。

```
aws ds-data describe-user command-name \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe'
```

输出：

```
{
    "DirectoryId": "d-1234567890",
    "DistinguishedName": "CN=john.doe,OU=Users,OU=CORP,DC=corp,DC=example,DC=com",
    "Enabled": false,
    "Realm": "corp.example.com",
    "SAMAccountName": "john.doe",
    "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567",
    "UserPrincipalName": "john.doe@CORP.EXAMPLE.COM"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUser。

disable-directory-data-access

以下代码示例演示了如何使用 disable-directory-data-access。

Amazon CLI

禁用目录的 Directory Service Data API

以下 disable-directory-data-access 示例禁用指定目录的 Directory Service Data API。

```
aws ds disable-directory-data-access \
    --directory-id d-1234567890
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Enabling or disabling user and group management or Amazon Directory Service Data。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableDirectoryDataAccess。

disable-user

以下代码示例演示了如何使用 disable-user。

Amazon CLI

禁用用户

以下 disable-user 示例禁用指定目录中的指定用户。

```
aws ds-data disable-user \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Disabling an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableUser。

`enable-directory-data-access`

以下代码示例演示了如何使用 `enable-directory-data-access`。

Amazon CLI

启用目录的 Directory Service Data API

以下 `enable-directory-data-access` 示例启用指定目录的 Directory Service Data API。

```
aws ds enable-directory-data-access \
    --directory-id d-1234567890
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Enabling or disabling user and group management or Amazon Directory Service Data。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 EnableDirectoryDataAccess。

`list-group-members`

以下代码示例演示了如何使用 `list-group-members`。

Amazon CLI

列出目录的组成员

以下 `list-group-members` 示例列出指定目录中的指定组的组成员。

```
aws ds-data list-group-members \
    --directory-id d-1234567890 \
    --sam-account-name 'sales'
```

输出：

```
{
```

```
    "Members": [
        {
            "MemberType": "USER",
            "SAMAccountName": "Jane Doe",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4568"
        },
        {
            "MemberType": "USER",
            "SAMAccountName": "John Doe",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4569"
        }
    ],
    "DirectoryId": "d-1234567890",
    "MemberRealm": "corp.example.com",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD group's details。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupMembers。

`list-groups-for-member`

以下代码示例演示了如何使用 `list-groups-for-member`。

Amazon CLI

列出目录的组成员资格

以下 `list-groups-for-member` 示例列出指定目录中的指定用户的组成员资格。

```
aws ds-data list-groups-for-member \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe'
```

输出：

```
{
    "Groups": [
        {
            "GroupScope": "Global",
            "GroupType": "Security",
```

```
            "SAMAccountName": "Domain Users",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        }
    ],
    "DirectoryId": "d-1234567890",
    "MemberRealm": "corp.example.com",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListGroupsForMember。

`list-groups`

以下代码示例演示了如何使用 `list-groups`。

Amazon CLI

列出目录的组

以下 `list-groups` 示例列出指定目录中的组。

```
aws ds-data list-groups \
    --directory-id d-1234567890
```

输出：

```
{
    "Groups": [
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Administrators",
            "SID": "S-1-2-33-441"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Users",
            "SID": "S-1-2-33-442"
```

```
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Guests",
            "SID": "S-1-2-33-443"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Print Operators",
            "SID": "S-1-2-33-444"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Backup Operators",
            "SID": "S-1-2-33-445"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Replicator",
            "SID": "S-1-2-33-446"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Remote Desktop Users",
            "SID": "S-1-2-33-447"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Network Configuration Operators",
            "SID": "S-1-2-33-448"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Performance Monitor Users",
            "SID": "S-1-2-33-449"
        },
        {
```

```
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Performance Log Users",
            "SID": "S-1-2-33-450"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Distributed COM Users",
            "SID": "S-1-2-33-451"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "IIS_IUSRS",
            "SID": "S-1-2-33-452"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Cryptographic Operators",
            "SID": "S-1-2-33-453"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Event Log Readers",
            "SID": "S-1-2-33-454"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Certificate Service DCOM Access",
            "SID": "S-1-2-33-456"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "RDS Remote Access Servers",
            "SID": "S-1-2-33-457"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
```

```
            "SAMAccountName": "RDS Endpoint Servers",
            "SID": "S-1-2-33-458"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "RDS Management Servers",
            "SID": "S-1-2-33-459"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Hyper-V Administrators",
            "SID": "S-1-2-33-460"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Access Control Assistance Operators",
            "SID": "S-1-2-33-461"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Remote Management Users",
            "SID": "S-1-2-33-462"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Storage Replica Administrators",
            "SID": "S-1-2-33-463"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Domain Computers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-789"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Domain Controllers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-790"
```

```
        },
        {
            "GroupScope": "Universal",
            "GroupType": "Security",
            "SAMAccountName": "Schema Admins",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-791"
        },
        {
            "GroupScope": "Universal",
            "GroupType": "Security",
            "SAMAccountName": "Enterprise Admins",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-792"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "Cert Publishers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-793"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Domain Admins",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-794"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Domain Users",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-795"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Domain Guests",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-796"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Group Policy Creator Owners",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-797"
        },
        {
```

```
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "RAS and IAS Servers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-798"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Server Operators",
            "SID": "S-1-2-33-464"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Account Operators",
            "SID": "S-1-2-33-465"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Pre-Windows 2000 Compatible Access",
            "SID": "S-1-2-33-466"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Incoming Forest Trust Builders",
            "SID": "S-1-2-33-467"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Windows Authorization Access Group",
            "SID": "S-1-2-33-468"
        },
        {
            "GroupScope": "BuiltinLocal",
            "GroupType": "Security",
            "SAMAccountName": "Terminal Server License Servers",
            "SID": "S-1-2-33-469"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
```

```
            "SAMAccountName": "Allowed RODC Password Replication Group",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-798"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "Denied RODC Password Replication Group",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-799"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Read-only Domain Controllers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-800"
        },
        {
            "GroupScope": "Universal",
            "GroupType": "Security",
            "SAMAccountName": "Enterprise Read-only Domain Controllers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-801"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Cloneable Domain Controllers",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-802"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Protected Users",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-803"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Key Admins",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-804"
        },
        {
            "GroupScope": "Universal",
            "GroupType": "Security",
            "SAMAccountName": "Enterprise Key Admins",
            "SID": "S-1-2-34-56789123456-7891012345-6789123486-805"
```

```
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "DnsAdmins",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "DnsUpdateProxy",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4568"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "Admins",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4569"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWSAdministrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4570"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Object Management Service Accounts",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4571"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Private CA Connector for AD Delegated Group",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4572"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Application and Service Delegated Group",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4573"
        },
        {
```

```
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4574"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated FSx Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4575"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Account Operators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4576"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Active Directory Based Activation
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4577"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Allowed to Authenticate Objects",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4578"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Allowed to Authenticate to Domain
Controllers",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4579"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Deleted Object Lifetime
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4580"
        },
```

```
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Distributed File System
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4581"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Dynamic Host Configuration Protocol
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4582"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Enterprise Certificate Authority
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4583"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Fine Grained Password Policy
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4584"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Group Policy Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4585"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Managed Service Account
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4586"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
```

```
            "SAMAccountName": "AWS Delegated Read Foreign Security Principals",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4587"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Remote Access Service Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4588"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Replicate Directory Changes
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4588"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Sites and Services Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4589"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated System Management Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4590"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Terminal Server Licensing
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4591"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated User Principal Name Suffix
Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4592"
        },
        {
            "GroupScope": "DomainLocal",
```

```
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Add Workstations To Domain Users",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4593"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Domain Name System Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4594"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Kerberos Delegation Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4595"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated Server Administrators",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4596"
        },
        {
            "GroupScope": "DomainLocal",
            "GroupType": "Security",
            "SAMAccountName": "AWS Delegated MS-NPRC Non-Compliant Devices",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4597"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Remote Access",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4598"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Security",
            "SAMAccountName": "Accounting",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4599"
        },
        {
            "GroupScope": "Global",
            "GroupType": "Distribution",
            "SAMAccountName": "sales",
```

```
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        }
    ],
    "DirectoryId": "d-1234567890",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD group's details。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

`list-users`

以下代码示例演示了如何使用 `list-users`。

Amazon CLI

列出目录的用户

以下 `list-users` 示例列出指定目录中的用户。

```
aws ds-data list-users \
    --directory-id d-1234567890
```

输出：

```
{
    "Users": [
        {
            "Enabled": true,
            "SAMAccountName": "Administrator",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-345"
        },
        {
            "Enabled": false,
            "SAMAccountName": "Guest",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-345"
        },
        {
            "Enabled": false,
            "SAMAccountName": "krbtgt",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-346"
```

```
        },
        {
            "Enabled": true,
            "SAMAccountName": "Admin",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-347"
        },
        {
            "Enabled": true,
            "SAMAccountName": "Richard Roe",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-348"
        },
        {
            "Enabled": true,
            "SAMAccountName": "Jane Doe",
            "SID": "S-1-2-34-5678910123-4567895012-3456789012-349"
        },
        {
            "Enabled": true,
            "SAMAccountName": "AWS_WGnzYlN6YyY",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        },
        {
            "Enabled": true,
            "SAMAccountName": "john.doe",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4568"
        }
    ],
    "DirectoryId": "d-1234567890",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

`remove-group-member`

以下代码示例演示了如何使用 `remove-group-member`。

Amazon CLI

从目录中移除组成员

以下 remove-group-member 示例从指定目录中的指定组中移除指定的组成员。

```
aws ds-data remove-group-member \
    --directory-id d-1234567890 \
    --group-name 'sales' \
    --member-name 'john.doe'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Adding and removing Amazon Managed Microsoft AD members to groups and groups to groups。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RemoveGroupMember。

reset-user-password

以下代码示例演示了如何使用 reset-user-password。

Amazon CLI

重置目录中的用户密码

以下 reset-user-password 示例重置和启用指定目录中的指定用户的密码。

```
aws ds reset-user-password \
    --directory-id d-1234567890 \
    --user-name 'john.doe' \
    --new-password 'password'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Resetting and enabling an Amazon Managed Microsoft AD user's password。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ResetUserPassword。

search-groups

以下代码示例演示了如何使用 search-groups。

Amazon CLI

在目录中搜索组

以下 search-groups 示例在指定目录中搜索指定组。

```
aws ds-data search-groups \
    --directory-id d-1234567890 \
    --search-attributes 'SamAccountName' \
    --search-string 'sales'
```

输出：

```
{
    "Groups": [
        {
            "GroupScope": "Global",
            "GroupType": "Distribution",
            "SAMAccountName": "sales",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        }
    ],
    "DirectoryId": "d-1234567890",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD group's details。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SearchGroups。

search-users

以下代码示例演示了如何使用 search-users。

Amazon CLI

在目录中搜索用户

以下 search-users 示例在指定目录中搜索指定用户。

```
aws ds-data search-users \
    --directory-id d-1234567890 \
```

```
    --search-attributes 'SamAccountName' \
    --Search-string 'john.doe'
```

输出：

```
{
    "Users": [
        {
            "Enabled": true,
            "SAMAccountName": "john.doe",
            "SID": "S-1-2-34-5678901234-5678901234-5678910123-4567"
        }
    ],
    "DirectoryId": "d-1234567890",
    "Realm": "corp.example.com"
}
```

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SearchUsers。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

更新目录中的组的属性

以下 update-group 示例更新指定目录中的指定组的指定属性。

```
aws ds-data update-group \
    --directory-id d-1234567890 \
    --sam-account-name 'sales' \
    --update-type 'REPLACE' \
    --group-type 'Distribution'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD group's details。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

update-user

以下代码示例演示了如何使用 update-user。

Amazon CLI

更新目录中的用户的属性

以下 update-user 示例更新指定目录中的指定用户的指定属性。

```
aws ds-data update-user \
    --directory-id d-1234567890 \
    --sam-account-name 'john.doe' \
    --update-type 'ADD' \
    --email-address 'example.corp.com'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Directory Service Administration Guide》中的 Viewing and updating an Amazon Managed Microsoft AD user。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUser。

Amazon DMS使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon DMS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-tags-to-resource

以下代码示例演示了如何使用 add-tags-to-resource。

Amazon CLI

为资源添加标签

以下 add-tags-to-resource 示例为复制实例添加标签。

```
aws dms add-tags-to-resource \
    --resource-arn arn:aws:dms:us-east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
 \
    --tags Key=Environment,Value=PROD Key=Project,Value=dbMigration
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToResource。

create-endpoint

以下代码示例演示了如何使用 create-endpoint。

Amazon CLI

创建端点

以下 create-endpoint 示例为 Amazon S3 源创建端点。

```
aws dms create-endpoint \
    --endpoint-type source \
    --engine-name s3 \
    --endpoint-identifier src-endpoint \
    --s3-settings file://s3-settings.json
```

的内容s3-settings.json：

```
{
```

```
    "BucketName":"my-corp-data",
    "BucketFolder":"sourcedata",
    "ServiceAccessRoleArn":"arn:aws:iam::123456789012:role/my-s3-access-role"
}
```

输出：

```
{
    "Endpoint": {
        "EndpointIdentifier": "src-endpoint",
        "EndpointType": "SOURCE",
        "EngineName": "s3",
        "EngineDisplayName": "Amazon S3",
        "ExtraConnectionAttributes": "bucketFolder=sourcedata;bucketName=my-corp-
data;compressionType=NONE;csvDelimiter=,;csvRowDelimiter=\\n;",
        "Status": "active",
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:GUVAFG34EECUOJ6QVZ56DAHT3U",
        "SslMode": "none",
        "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",
        "S3Settings": {
            "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-
role",
            "CsvRowDelimiter": "\\n",
            "CsvDelimiter": ",",
            "BucketFolder": "sourcedata",
            "BucketName": "my-corp-data",
            "CompressionType": "NONE",
            "EnableStatistics": true
        }
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEndpoint。

create-event-subscription

以下代码示例演示了如何使用 create-event-subscription。

Amazon CLI

列出事件订阅

以下 create-event-subscription 示例创建对 Amazon SNS 主题（my-sns-topic）的事件订阅。

```
aws dms create-event-subscription \
    --subscription-name my-dms-events \
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:my-sns-topic
```

输出：

```
{
    "EventSubscription": {
        "CustomerAwsId": "123456789012",
        "CustSubscriptionId": "my-dms-events",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
        "Status": "creating",
        "SubscriptionCreationTime": "2020-05-21 21:58:38.598",
        "Enabled": true
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEventSubscription。

create-replication-instance

以下代码示例演示了如何使用 create-replication-instance。

Amazon CLI

创建复制实例

以下 create-replication-instance 示例创建复制实例。

```
aws dms create-replication-instance \
    --replication-instance-identifier my-repl-instance \
    --replication-instance-class dms.t2.micro \
    --allocated-storage 5
```

输出：

```
{
    "ReplicationInstance": {
        "ReplicationInstanceIdentifier": "my-repl-instance",
        "ReplicationInstanceClass": "dms.t2.micro",
        "ReplicationInstanceStatus": "creating",
        "AllocatedStorage": 5,
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-f839b688",
                "Status": "active"
            }
        ],
        "ReplicationSubnetGroup": {
            "ReplicationSubnetGroupIdentifier": "default",
            "ReplicationSubnetGroupDescription": "default",
            "VpcId": "vpc-136a4c6a",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-da327bf6",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-42599426",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1d"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-bac383e0",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-6746046b",
                    "SubnetAvailabilityZone": {
```

```
                    "Name": "us-east-1f"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-d7c825e8",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1e"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-cbfff283",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1b"
                },
                "SubnetStatus": "Active"
            }
        ]
    },
    "PreferredMaintenanceWindow": "sat:12:35-sat:13:05",
    "PendingModifiedValues": {},
    "MultiAZ": false,
    "EngineVersion": "3.3.2",
    "AutoMinorVersionUpgrade": true,
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:ZK2VQBUWFDBAWHIXHAYG5G2PKY",
    "PubliclyAccessible": true
}
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplicationInstance。

create-replication-subnet-group

以下代码示例演示了如何使用 create-replication-subnet-group。

Amazon CLI

创建子网组

以下 create-replication-subnet-group 示例创建包含 3 个子网的组。

```
aws dms create-replication-subnet-group \
    --replication-subnet-group-identifier my-subnet-group \
    --replication-subnet-group-description "my subnet group" \
    --subnet-ids subnet-da327bf6 subnet-bac383e0 subnet-d7c825e8
```

输出：

```
{
    "ReplicationSubnetGroup": {
        "ReplicationSubnetGroupIdentifier": "my-subnet-group",
        "ReplicationSubnetGroupDescription": "my subnet group",
        "VpcId": "vpc-136a4c6a",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-da327bf6",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1a"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-bac383e0",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1c"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-d7c825e8",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1e"
                },
                "SubnetStatus": "Active"
            }
        ]
    }
```

```
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的为复制实例设置网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplicationSubnetGroup。

create-replication-task

以下代码示例演示了如何使用 create-replication-task。

Amazon CLI

创建复制任务

以下 create-replication-task 示例创建复制任务。

```
aws dms create-replication-task \
    --replication-task-identifier movedata \
    --source-endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA \
    --target-endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U \
    --replication-instance-arn $RI_ARN \
    --migration-type full-load \
    --table-mappings file://table-mappings.json
```

的内容table-mappings.json：

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "prodrep",
                "table-name": "%"
            },
            "rule-action": "include",
            "filters": []
        }
```

```
    ]
}
```

输出：

```
{
    "ReplicationTask": {
        "ReplicationTaskIdentifier": "moveit2",
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted... ,
        "ReplicationTaskSettings": ...output omitted... ,
        "Status": "creating",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplicationTask。

delete-connection

以下代码示例演示了如何使用 delete-connection。

Amazon CLI

删除连接

以下 delete-connection 示例解除端点与复制实例的关联。

```
aws dms delete-connection \
    --endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA \
```

```
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
```

输出：

```
{
    "Connection": {
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "Status": "deleting",
        "EndpointIdentifier": "src-database-1",
        "ReplicationInstanceIdentifier": "my-repl-instance"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的 https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.Creating.html。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnection。

delete-endpoint

以下代码示例演示了如何使用 delete-endpoint。

Amazon CLI

删除端点

以下 delete-endpoint 示例删除端点。

```
aws dms delete-endpoint \
    --endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:OUJJVXO4XZ4CYTSEG5XGMN2R3Y
```

输出：

```
{
    "Endpoint": {
        "EndpointIdentifier": "src-endpoint",
        "EndpointType": "SOURCE",
```

```
        "EngineName": "s3",
        "EngineDisplayName": "Amazon S3",
        "ExtraConnectionAttributes": "bucketFolder=sourcedata;bucketName=my-corp-
data;compressionType=NONE;csvDelimiter=,;csvRowDelimiter=\\n;",
        "Status": "deleting",
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:OUJJVXO4XZ4CYTSEG5XGMN2R3Y",
        "SslMode": "none",
        "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",
        "S3Settings": {
            "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-
role",
            "CsvRowDelimiter": "\\n",
            "CsvDelimiter": ",",
            "BucketFolder": "sourcedata",
            "BucketName": "my-corp-data",
            "CompressionType": "NONE",
            "EnableStatistics": true
        }
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEndpoint。

delete-event-subscription

以下代码示例演示了如何使用 delete-event-subscription。

Amazon CLI

删除事件订阅

以下 delete-event-subscription 示例删除 Amazon SNS 主题的订阅。

```
aws dms delete-event-subscription \
    --subscription-name "my-dms-events"
```

输出：

```
{
```

```
    "EventSubscription": {
        "CustomerAwsId": "123456789012",
        "CustSubscriptionId": "my-dms-events",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
        "Status": "deleting",
        "SubscriptionCreationTime": "2020-05-21 21:58:38.598",
        "Enabled": true
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEventSubscription。

delete-replication-instance

以下代码示例演示了如何使用 delete-replication-instance。

Amazon CLI

删除复制实例

以下 delete-replication-instance 示例删除复制实例。

```
aws dms delete-replication-instance \
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
```

输出：

```
{
    "ReplicationInstance": {
        "ReplicationInstanceIdentifier": "my-repl-instance",
        "ReplicationInstanceClass": "dms.t2.micro",
        "ReplicationInstanceStatus": "deleting",
        "AllocatedStorage": 5,
        "InstanceCreateTime": 1590011235.952,
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-f839b688",
                "Status": "active"
            }
        ],
```

```
        "AvailabilityZone": "us-east-1e",
        "ReplicationSubnetGroup": {
            "ReplicationSubnetGroupIdentifier": "default",
            "ReplicationSubnetGroupDescription": "default",
            "VpcId": "vpc-136a4c6a",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-da327bf6",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-42599426",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1d"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-bac383e0",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-6746046b",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1f"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-d7c825e8",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1e"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-cbfff283",
```

```
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1b"
                    },
                    "SubnetStatus": "Active"
                }
            ]
        },
        "PreferredMaintenanceWindow": "wed:11:42-wed:12:12",
        "PendingModifiedValues": {},
        "MultiAZ": true,
        "EngineVersion": "3.3.2",
        "AutoMinorVersionUpgrade": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "ReplicationInstancePublicIpAddress": "54.225.120.92",
        "ReplicationInstancePrivateIpAddress": "172.31.30.121",
        "ReplicationInstancePublicIpAddresses": [
            "54.225.120.92",
            "3.230.18.248"
        ],
        "ReplicationInstancePrivateIpAddresses": [
            "172.31.30.121",
            "172.31.75.90"
        ],
        "PubliclyAccessible": true,
        "SecondaryAvailabilityZone": "us-east-1b"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReplicationInstance。

`delete-replication-subnet-group`

以下代码示例演示了如何使用 `delete-replication-subnet-group`。

Amazon CLI

删除子网组

以下 delete-replication-subnet-group 示例删除子网组。

```
aws dms delete-replication-subnet-group \
--replication-subnet-group-identifier my-subnet-group
```

输出：

```
(none)
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的为复制实例设置网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReplicationSubnetGroup。

delete-replication-task

以下代码示例演示了如何使用 delete-replication-task。

Amazon CLI

删除复制任务

以下 delete-replication-task 示例删除复制任务。

```
aws dms delete-replication-task \
    --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII
```

输出：

```
{
    "ReplicationTask": {
        "ReplicationTaskIdentifier": "moveit2",
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted...,
        "ReplicationTaskSettings": ...output omitted...,
```

```
        "Status": "deleting",
        "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590789988.677,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReplicationTask。

describe-account-attributes

以下代码示例演示了如何使用 describe-account-attributes。

Amazon CLI

描述账户属性

以下 describe-account-attributes 示例列出您 Amazon 账户的属性。

```
aws dms describe-account-attributes
```

输出：

```
{
    "AccountQuotas": [
        {
            "AccountQuotaName": "ReplicationInstances",
            "Used": 1,
            "Max": 20
        },
        {
            "AccountQuotaName": "AllocatedStorage",
            "Used": 5,
            "Max": 10000
        },

        ...remaining output omitted...
```

```
    ],
    "UniqueAccountIdentifier": "cqahfbfy5xee"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAttributes。

describe-certificates

以下代码示例演示了如何使用 describe-certificates。

Amazon CLI

列出可用证书

以下 describe-certificates 示例列出您 Amazon 账户中的可用证书。

```
aws dms describe-certificates
```

输出：

```
{
    "Certificates": [
        {
            "CertificateIdentifier": "my-cert",
            "CertificateCreationDate": 1543259542.506,
            "CertificatePem": "-----BEGIN CERTIFICATE-----
\nMIID9DCCAtygAwIBAgIBQjANBgkqhkiG9w0BAQ ...U"

            ... remaining output omittted ...

        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 SSL。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificates。

describe-connections

以下代码示例演示了如何使用 describe-connections。

Amazon CLI

描述连接

以下 describe-connections 示例列出您在复制实例和端点之间测试过的连接。

```
aws dms describe-connections
```

输出：

```
{
    "Connections": [
        {
            "Status": "successful",
            "ReplicationInstanceIdentifier": "test",
            "EndpointArn": "arn:aws:dms:us-east-arn:aws:dms:us-
east-1:123456789012:endpoint:ZW5UAN6P4E77EC7YWHK4RZZ3BE",
            "EndpointIdentifier": "testsrc1",
            "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:6UTDJGBOUS3VI3SUWA66XFJCJQ"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的创建源端点和目标端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConnections。

describe-endpoint-types

以下代码示例演示了如何使用 describe-endpoint-types。

Amazon CLI

列出可用的端点类型

以下 describe-endpoint-types 示例列出可用的 MySQL 端点类型。

```
aws dms describe-endpoint-types \
    --filters "Name=engine-name,Values=mysql"
```

输出：

```
{
    "SupportedEndpointTypes": [
        {
            "EngineName": "mysql",
            "SupportsCDC": true,
            "EndpointType": "source",
            "EngineDisplayName": "MySQL"
        },
        {
            "EngineName": "mysql",
            "SupportsCDC": true,
            "EndpointType": "target",
            "EngineDisplayName": "MySQL"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的“使用 Amazon DMS 端点 <https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpointTypes。

`describe-endpoints`

以下代码示例演示了如何使用 `describe-endpoints`。

Amazon CLI

描述端点

以下 `describe-endpoints` 示例列出您 Amazon 账户中的端点。

```
aws dms describe-endpoints
```

输出：

```
{
    "Endpoints": [
        {
            "Username": "dms",
```

```
            "Status": "active",
            "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:SF2WOFLWYWKVEOHID2EKLP3SJI",
            "ServerName": "ec2-52-32-48-61.us-west-2.compute.amazonaws.com",
            "EndpointType": "SOURCE",
            "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/94d5c4e7-4e4c-44be-
b58a-c8da7adf57cd",
            "DatabaseName": "test",
            "EngineName": "mysql",
            "EndpointIdentifier": "pri100",
            "Port": 8193
        },
        {
            "Username": "admin",
            "Status": "active",
            "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:TJJZCIH3CJ24TJRU4VC32WEWFR",
            "ServerName": "test.example.com",
            "EndpointType": "SOURCE",
            "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/2431021b-1cf2-
a2d4-77b2-59a9e4bce323",
            "DatabaseName": "EMPL",
            "EngineName": "oracle",
            "EndpointIdentifier": "test",
            "Port": 1521
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpoints。

`describe-event-categories`

以下代码示例演示了如何使用 `describe-event-categories`。

Amazon CLI

描述事件类别

以下 `describe-event-categories` 示例列出可用的事件类别。

```
aws dms describe-event-categories
```

输出：

```
{
    "EventCategoryGroupList": [
        {
            "SourceType": "replication-instance",
            "EventCategories": [
                "low storage",
                "configuration change",
                "maintenance",
                "deletion",
                "creation",
                "failover",
                "failure"
            ]
        },
        {
            "SourceType": "replication-task",
            "EventCategories": [
                "configuration change",
                "state change",
                "deletion",
                "creation",
                "failure"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventCategories。

`describe-event-subscriptions`

以下代码示例演示了如何使用 `describe-event-subscriptions`。

Amazon CLI

描述事件订阅

以下 describe-event-subscriptions 示例列出对 Amazon SNS 主题的事件订阅。

```
aws dms describe-event-subscriptions
```

输出：

```
{
    "EventSubscriptionsList": [
        {
            "CustomerAwsId": "123456789012",
            "CustSubscriptionId": "my-dms-events",
            "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
            "Status": "deleting",
            "SubscriptionCreationTime": "2020-05-21 22:28:51.924",
            "Enabled": true
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventSubscriptions。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

列出 DMS 事件

以下 describe-events 示例列出源自复制实例的事件。

```
aws dms describe-events \
    --source-type "replication-instance"
```

输出：

```
{
    "Events": [
        {
            "SourceIdentifier": "my-repl-instance",
```

```
            "SourceType": "replication-instance",
            "Message": "Replication application shutdown",
            "EventCategories": [],
            "Date": 1590771645.776
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-orderable-replication-instances

以下代码示例演示了如何使用 describe-orderable-replication-instances。

Amazon CLI

描述可排序的复制实例

以下 describe-orderable-replication-instances 示例列出您可以排序的复制实例类型。

```
aws dms describe-orderable-replication-instances
```

输出：

```
{
    "OrderableReplicationInstances": [
        {
            "EngineVersion": "3.3.2",
            "ReplicationInstanceClass": "dms.c4.2xlarge",
            "StorageType": "gp2",
            "MinAllocatedStorage": 5,
            "MaxAllocatedStorage": 6144,
            "DefaultAllocatedStorage": 100,
            "IncludedAllocatedStorage": 100,
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
```

```
                "us-east-1e",
                "us-east-1f"
            ]
        },
        {
            "EngineVersion": "3.3.2",
            "ReplicationInstanceClass": "dms.c4.4xlarge",
            "StorageType": "gp2",
            "MinAllocatedStorage": 5,
            "MaxAllocatedStorage": 6144,
            "DefaultAllocatedStorage": 100,
            "IncludedAllocatedStorage": 100,
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ]
        },

        ...remaining output omitted...

    }
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrderableReplicationInstances。

`describe-refresh-schemas-status`

以下代码示例演示了如何使用 `describe-refresh-schemas-status`。

Amazon CLI

列出端点的刷新状态

以下 `describe-refresh-schemas-status` 示例返回先前刷新请求的状态。

```
aws dms describe-refresh-schemas-status \
```

```
    --endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA
```

输出：

```
{
    "RefreshSchemasStatus": {
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "Status": "successful",
        "LastRefreshDate": 1590786544.605
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRefreshSchemasStatus。

describe-replication-instances

以下代码示例演示了如何使用 describe-replication-instances。

Amazon CLI

描述复制实例

以下 describe-replication-instances 示例列出您 Amazon 账户中的复制实例。

```
aws dms describe-replication-instances
```

输出：

```
{
    "ReplicationInstances": [
        {
            "ReplicationInstanceIdentifier": "my-repl-instance",
            "ReplicationInstanceClass": "dms.t2.micro",
            "ReplicationInstanceStatus": "available",
            "AllocatedStorage": 5,
            "InstanceCreateTime": 1590011235.952,
            "VpcSecurityGroups": [
                {
```

```
                "VpcSecurityGroupId": "sg-f839b688",
                "Status": "active"
            }
        ],
        "AvailabilityZone": "us-east-1e",
        "ReplicationSubnetGroup": {
            "ReplicationSubnetGroupIdentifier": "default",
            "ReplicationSubnetGroupDescription": "default",
            "VpcId": "vpc-136a4c6a",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-da327bf6",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-42599426",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1d"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-bac383e0",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-6746046b",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1f"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-d7c825e8",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1e"
                    },
```

```
                        "SubnetStatus": "Active"
                    },
                    {
                        "SubnetIdentifier": "subnet-cbfff283",
                        "SubnetAvailabilityZone": {
                            "Name": "us-east-1b"
                        },
                        "SubnetStatus": "Active"
                    }
                ]
            },
            "PreferredMaintenanceWindow": "wed:11:42-wed:12:12",
            "PendingModifiedValues": {
                "MultiAZ": true
            },
            "MultiAZ": false,
            "EngineVersion": "3.3.2",
            "AutoMinorVersionUpgrade": true,
            "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
f7bc0f8e-1a3a-4ace-9faa-e8494fa3921a",
            "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
            "ReplicationInstancePublicIpAddress": "3.230.18.248",
            "ReplicationInstancePrivateIpAddress": "172.31.75.90",
            "ReplicationInstancePublicIpAddresses": [
                "3.230.18.248"
            ],
            "ReplicationInstancePrivateIpAddresses": [
                "172.31.75.90"
            ],
            "PubliclyAccessible": true,
            "FreeUntil": 1590194829.267
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplicationInstances。

describe-replication-subnet-groups

以下代码示例演示了如何使用 describe-replication-subnet-groups。

Amazon CLI

显示可用的子网组

以下 describe-replication-subnet-groups 示例列出可用的子网组。

```
aws dms describe-replication-subnet-groups \
    --filter "Name=replication-subnet-group-id,Values=my-subnet-group"
```

输出：

```
{
    "ReplicationSubnetGroups": [
        {
            "ReplicationSubnetGroupIdentifier": "my-subnet-group",
            "ReplicationSubnetGroupDescription": "my subnet group",
            "VpcId": "vpc-136a4c6a",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-da327bf6",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-bac383e0",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-d7c825e8",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1e"
                    },
                    "SubnetStatus": "Active"
```

```
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的为复制实例设置网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplicationSubnetGroups。

describe-replication-task-assessment-results

以下代码示例演示了如何使用 describe-replication-task-assessment-results。

Amazon CLI

列出复制任务评估的结果

以下 describe-replication-task-assessment-results 示例列出先前任务评估的结果。

```
aws dms describe-replication-task-assessment-results
```

输出：

```
{
    "ReplicationTaskAssessmentResults": [
        {
            "ReplicationTaskIdentifier": "moveit2",
            "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
            "ReplicationTaskLastAssessmentDate": 1590790230.0,
            "AssessmentStatus": "No issues found",
            "AssessmentResultsFile": "moveit2/2020-05-29-22-10"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的创建任务评估报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplicationTaskAssessmentResults。

describe-replication-tasks

以下代码示例演示了如何使用 describe-replication-tasks。

Amazon CLI

描述复制任务

以下 describe-replication-tasks 示例描述当前复制任务。

```
aws dms describe-replication-tasks
```

输出：

```
{
    "ReplicationTasks": [
        {
            "ReplicationTaskIdentifier": "moveit2",
            "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
            "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
            "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
            "MigrationType": "full-load",
            "TableMappings": ...output omitted... ,
            "ReplicationTaskSettings": ...output omitted... ,
            "Status": "stopped",
            "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
            "ReplicationTaskCreationDate": 1590524772.505,
            "ReplicationTaskStartDate": 1590619805.212,
            "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
            "ReplicationTaskStats": {
                "FullLoadProgressPercent": 100,
                "ElapsedTimeMillis": 0,
                "TablesLoaded": 0,
                "TablesLoading": 0,
                "TablesQueued": 0,
                "TablesErrored": 0,
                "FreshStartDate": 1590619811.528,
                "StartDate": 1590619811.528,
                "StopDate": 1590619842.068
```

```
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplicationTasks。

`describe-schemas`

以下代码示例演示了如何使用 `describe-schemas`。

Amazon CLI

描述数据库架构

以下 `describe-schemas` 示例列出端点上可用的表。

```
aws dms describe-schemas \
    --endpoint-arn "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA"
```

输出：

```
{
    "Schemas": [
        "prodrep"
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的这是主题标题。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSchemas。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出复制实例的标签。

```
aws dms list-tags-for-resource \
    --resource-arn arn:aws:dms:us-east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
```

输出：

```
{
    "TagList": [
        {
            "Key": "Project",
            "Value": "dbMigration"
        },
        {
            "Key": "Environment",
            "Value": "PROD"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

modify-endpoint

以下代码示例演示了如何使用 modify-endpoint。

Amazon CLI

修改端点

以下 modify-endpoint 示例向端点添加一个额外的连接属性。

```
aws dms modify-endpoint \
    --endpoint-arn "arn:aws:dms:us-
east-1:123456789012:endpoint:GUVAFG34EECUOJ6QVZ56DAHT3U" \
    --extra-connection-attributes "compressionType=GZIP"
```

输出：

```
{
```

```
    "Endpoint": {
        "EndpointIdentifier": "src-endpoint",
        "EndpointType": "SOURCE",
        "EngineName": "s3",
        "EngineDisplayName": "Amazon S3",
        "ExtraConnectionAttributes":
 "compressionType=GZIP;csvDelimiter=,;csvRowDelimiter=\\n;",
        "Status": "active",
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:GUVAFG34EECUOJ6QVZ56DAHT3U",
        "SslMode": "none",
        "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",
        "S3Settings": {
            "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-
role",
            "CsvRowDelimiter": "\\n",
            "CsvDelimiter": ",",
            "BucketFolder": "",
            "BucketName": "",
            "CompressionType": "GZIP",
            "EnableStatistics": true
        }
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的“使用 Amazon DMS 端点 <https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyEndpoint。

`modify-event-subscription`

以下代码示例演示了如何使用 `modify-event-subscription`。

Amazon CLI

修改事件订阅

以下 `modify-event-subscription` 实例更改事件订阅的源类型。

```
aws dms modify-event-subscription \
    --subscription-name "my-dms-events" \
    --source-type replication-task
```

输出：

```
{
    "EventSubscription": {
        "CustomerAwsId": "123456789012",
        "CustSubscriptionId": "my-dms-events",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
        "Status": "modifying",
        "SubscriptionCreationTime": "2020-05-29 17:04:40.262",
        "SourceType": "replication-task",
        "Enabled": true
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用事件和通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyEventSubscription。

`modify-replication-instance`

以下代码示例演示了如何使用 `modify-replication-instance`。

Amazon CLI

修改复制实例

以下 `modify-replication-instance` 示例修改复制实例，使其使用多可用区部署。

```
aws dms modify-replication-instance \
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE \
    --multi-az
```

输出：

```
{
    "ReplicationInstance": {
        "ReplicationInstanceIdentifier": "my-repl-instance",
        "ReplicationInstanceClass": "dms.t2.micro",
        "ReplicationInstanceStatus": "available",
        "AllocatedStorage": 5,
        "InstanceCreateTime": 1590011235.952,
```

```
        ...output omitted...

        "PendingModifiedValues": {
            "MultiAZ": true
        },
        "MultiAZ": false,
        "EngineVersion": "3.3.2",
        "AutoMinorVersionUpgrade": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",

        ...output omitted...

    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReplicationInstance。

`modify-replication-subnet-group`

以下代码示例演示了如何使用 `modify-replication-subnet-group`。

Amazon CLI

修改子网组

以下 `modify-replication-subnet-group` 示例更改与子网组关联的子网列表。

```
aws dms modify-replication-subnet-group \
    --replication-subnet-group-identifier my-subnet-group \
    --subnet-id subnet-da327bf6 subnet-bac383e0
```

输出：

```
{
    "ReplicationSubnetGroup": {
        "ReplicationSubnetGroupIdentifier": "my-subnet-group",
        "ReplicationSubnetGroupDescription": "my subnet group",
```

```
        "VpcId": "vpc-136a4c6a",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-da327bf6",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1a"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-bac383e0",
                "SubnetAvailabilityZone": {
                    "Name": "us-east-1c"
                },
                "SubnetStatus": "Active"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的为复制实例设置网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReplicationSubnetGroup。

modify-replication-task

以下代码示例演示了如何使用 modify-replication-task。

Amazon CLI

修改复制任务

以下 modify-replication-task 示例更改任务的表映射。

```
aws dms modify-replication-task \
    --replication-task-arn "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII" \
    --table-mappings file://table-mappings.json
```

的内容table-mappings.json：

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "prodrep",
                "table-name": "ACCT_%"
            },
            "rule-action": "include",
            "filters": []
        }
    ]
}
```

输出：

```
{
    "ReplicationTask": {
        "ReplicationTaskIdentifier": "moveit2",
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted...,
        "ReplicationTaskSettings": ...output omitted...,
        "Status": "modifying",
        "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590789424.653,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReplicationTask。

reboot-replication-instance

以下代码示例演示了如何使用 reboot-replication-instance。

Amazon CLI

重新引导复制实例

以下 reboot-replication-instance 示例重新引导复制实例。

```
aws dms reboot-replication-instance \
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
```

输出：

```
{
    "ReplicationInstance": {
        "ReplicationInstanceIdentifier": "my-repl-instance",
        "ReplicationInstanceClass": "dms.t2.micro",
        "ReplicationInstanceStatus": "rebooting",
        "AllocatedStorage": 5,
        "InstanceCreateTime": 1590011235.952,
    ... output omitted ...
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 复制实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootReplicationInstance。

refresh-schemas

以下代码示例演示了如何使用 refresh-schemas。

Amazon CLI

刷新数据库架构

以下 refresh-schemas 示例请求 Amazon DMS 刷新端点上的架构列表。

```
aws dms refresh-schemas \
```

```
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE \
    --endpoint-arn "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA"
```

输出：

```
{
    "RefreshSchemasStatus": {
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "Status": "refreshing",
        "LastRefreshDate": 1590019949.103
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RefreshSchemas。

reload-tables

以下代码示例演示了如何使用 reload-tables。

Amazon CLI

刷新端点上可用表的列表

以下 reload-tables 示例重新加载端点上可用表的列表。

```
aws dms reload-tables \
    --replication-task-arn "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII" \
    --tables-to-reload "SchemaName=prodrep,TableName=ACCT_BAL"
```

输出：

```
{
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReloadTables。

remove-tags-from-resource

以下代码示例演示了如何使用 remove-tags-from-resource。

Amazon CLI

从复制实例中移除标签

以下 remove-tags-from-resource 示例从复制实例中移除标签。

```
aws dms remove-tags-from-resource \
    --resource-arn arn:aws:dms:us-east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE
 \
    --tag-keys Environment Project
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromResource。

start-replication-task-assessment

以下代码示例演示了如何使用 start-replication-task-assessment。

Amazon CLI

启动作业评估

以下 start-replication-task-assessment 示例启动复制任务评估。

```
aws dms start-replication-task-assessment \
    --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII
```

输出：

```
{
    "ReplicationTask": {
```

```
        "ReplicationTaskIdentifier": "moveit2",
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted...,
        "ReplicationTaskSettings": ...output omitted...,
        "Status": "testing",
        "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590789988.677,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的创建任务评估报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReplicationTaskAssessment。

start-replication-task

以下代码示例演示了如何使用 start-replication-task。

Amazon CLI

启动复制任务

以下 command-name 示例列出您的 Amazon 账户中可用的小组件。

```
aws dms start-replication-task \
    --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII \
    --start-replication-task-type reload-target
```

输出：

```
{
    "ReplicationTask": {
```

```
        "ReplicationTaskIdentifier": "moveit2",
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted... ,
        "ReplicationTaskSettings": ...output omitted... ,
        "Status": "starting",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590619805.212,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReplicationTask。

stop-replication-task

以下代码示例演示了如何使用 stop-replication-task。

Amazon CLI

停止任务

以下 stop-replication-task 示例停止任务。

```
aws dms stop-replication-task \
    --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII
```

输出：

```
{
    "ReplicationTask": {
        "ReplicationTaskIdentifier": "moveit2",
```

```
        "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted...,
        "ReplicationTaskSettings": ...output omitted...,
        "Status": "stopping",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590789424.653,
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的使用 Amazon DMS 任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopReplicationTask。

`test-connection`

以下代码示例演示了如何使用 `test-connection`。

Amazon CLI

测试与端点的连接

以下 `test-connection` 示例测试是否可以从复制实例访问端点。

```
aws dms test-connection \
    --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE \
    --endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA
```

输出：

```
{
    "Connection": {
```

```
        "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
        "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
        "Status": "testing",
        "EndpointIdentifier": "src-database-1",
        "ReplicationInstanceIdentifier": "my-repl-instance"
    }
}
```

有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的创建源端点和目标端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestConnection。

使用 Amazon CLI 的 Amazon DocumentDB 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon DocumentDB 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-resource`

以下代码示例演示了如何使用 `add-tags-to-resource`。

Amazon CLI

为指定资源添加一个或多个标签

以下 `add-tags-to-resource` 示例向 `sample-cluster` 添加三个标签。一个标签（`CropB`）有键名但没有值。

```
aws docdb add-tags-to-resource \
    --resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \
    --tags Key="CropA",Value="Apple" Key="CropB" Key="CropC",Value="Corn"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的标记 Amazon DocumentDB 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToResource。

apply-pending-maintenance-action

以下代码示例演示了如何使用 apply-pending-maintenance-action。

Amazon CLI

在下一个维护时段内执行待处理的维护操作

以下 apply-pending-maintenance-action 示例在下一个计划维护时段内执行所有 system-update 操作。

```
aws docdb apply-pending-maintenance-action \
--resource-identifier arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \
--apply-action system-update \
--opt-in-type next-maintenance
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的应用 Amazon DocumentDB 更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ApplyPendingMaintenanceAction。

copy-db-cluster-parameter-group

以下代码示例演示了如何使用 copy-db-cluster-parameter-group。

Amazon CLI

复制现有的数据库集群参数组

以下 copy-db-cluster-parameter-group 示例复制名为 custom-docdb3-6-copy 的参数组 custom-docdb3-6。创建副本时，它会将标签添加到新的参数组中。

```
aws docdb copy-db-cluster-parameter-group \
    --source-db-cluster-parameter-group-identifier custom-docdb3-6 \
    --target-db-cluster-parameter-group-identifier custom-docdb3-6-copy \
    --target-db-cluster-parameter-group-description "Copy of custom-docdb3-6" \
    --tags Key="CopyNumber",Value="1" Key="Modifiable",Value="Yes"
```

输出：

```
{
    "DBClusterParameterGroup": {
        "DBParameterGroupFamily": "docdb3.6",
        "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:12345678901:cluster-
pg:custom-docdb3-6-copy",
        "DBClusterParameterGroupName": "custom-docdb3-6-copy",
        "Description": "Copy of custom-docdb3-6"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的复制 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbClusterParameterGroup。

copy-db-cluster-snapshot

以下代码示例演示了如何使用 copy-db-cluster-snapshot。

Amazon CLI

复制快照

以下 copy-db-cluster-snapshot 示例创建名为 sample-cluster-snapshot-copy 的 sample-cluster-snapshot 的副本。副本包含原始标签的所有标签以及带有键名 CopyNumber 的新标签。

```
aws docdb copy-db-cluster-snapshot \
    --source-db-cluster-snapshot-identifier sample-cluster-snapshot \
    --target-db-cluster-snapshot-identifier sample-cluster-snapshot-copy \
```

```
    --copy-tags \
    --tags Key="CopyNumber",Value="1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的复制集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbClusterSnapshot。

`create-db-cluster-parameter-group`

以下代码示例演示了如何使用 `create-db-cluster-parameter-group`。

Amazon CLI

创建 Amazon DocumentDB 集群参数组

以下 `create-db-cluster-parameter-group` 示例使用 `docdb3.6` 系列创建数据库集群参数组 `sample-parameter-group`。

```
aws docdb create-db-cluster-parameter-group \
    --db-cluster-parameter-group-name sample-parameter-group \
    --db-parameter-group-family docdb3.6 \
    --description "Sample parameter group based on docdb3.6"
```

输出：

```
{
    "DBClusterParameterGroup": {
        "Description": "Sample parameter group based on docdb3.6",
        "DBParameterGroupFamily": "docdb3.6",
        "DBClusterParameterGroupArn": "arn:aws:rds:us-west-2:123456789012:cluster-
pg:sample-parameter-group",
        "DBClusterParameterGroupName": "sample-parameter-group"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的创建 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbClusterParameterGroup。

create-db-cluster-snapshot

以下代码示例演示了如何使用 create-db-cluster-snapshot。

Amazon CLI

创建手动 Amazon DocumentDB 集群快照

以下 create-db-cluster-snapshot 示例创建一个名为 sample-cluster-snapshot 的 Amazon 数据库集群快照。

```
aws docdb create-db-cluster-snapshot \
   --db-cluster-identifier sample-cluster \
   --db-cluster-snapshot-identifier sample-cluster-snapshot
```

输出：

```
{
    "DBClusterSnapshot": {
        "MasterUsername": "master-user",
        "SnapshotCreateTime": "2019-03-18T18:27:14.794Z",
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c",
            "us-west-2d",
            "us-west-2e",
            "us-west-2f"
        ],
        "SnapshotType": "manual",
        "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-
snapshot:sample-cluster-snapshot",
        "EngineVersion": "3.6.0",
        "PercentProgress": 0,
        "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",
        "Engine": "docdb",
        "DBClusterIdentifier": "sample-cluster",
        "Status": "creating",
        "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
        "Port": 0,
        "StorageEncrypted": false,
        "VpcId": "vpc-91280df6"
    }
```

```
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的创建手动集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbClusterSnapshot。

`create-db-cluster`

以下代码示例演示了如何使用 `create-db-cluster`。

Amazon CLI

创建 Amazon DocumentDB 集群

以下 `create-db-cluster` 示例创建一个名为 `sample-cluster` 的 Amazon DocumentDB 集群，其首选维护时段为星期日 20:30 到 11:00。

```
aws docdb create-db-cluster \
    --db-cluster-identifier sample-cluster \
    --engine docdb \
    --master-username master-user \
    --master-user-password password \
    --preferred-maintenance-window Sun:20:30-Sun:21:00
```

输出：

```
{
    "DBCluster": {
        "DBClusterParameterGroup": "default.docdb3.6",
        "AssociatedRoles": [],
        "DBSubnetGroup": "default",
        "ClusterCreateTime": "2019-03-18T18:06:34.616Z",
        "Status": "creating",
        "Port": 27017,
        "PreferredMaintenanceWindow": "sun:20:30-sun:21:00",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "DBClusterMembers": [],
        "Engine": "docdb",
        "DBClusterIdentifier": "sample-cluster",
        "PreferredBackupWindow": "10:12-10:42",
        "AvailabilityZones": [
            "us-west-2d",
            "us-west-2f",
```

```
            "us-west-2e"
        ],
        "MasterUsername": "master-user",
        "BackupRetentionPeriod": 1,
        "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-77186e0d",
                "Status": "active"
            }
        ],
        "StorageEncrypted": false,
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
        "DbClusterResourceId": "cluster-L3R4YRSBUYDP4GLMTJ2WF5GH5Q",
        "MultiAZ": false,
        "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "EngineVersion": "3.6.0"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的创建 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbCluster。

`create-db-instance`

以下代码示例演示了如何使用 `create-db-instance`。

Amazon CLI

创建 Amazon DocumentDB 集群实例

以下 `create-db-instance` 示例代码在 Amazon DocumentDB 集群 `sample-cluster` 中创建实例 `sample-cluster-instance-2`。

```
aws docdb create-db-instance \
    --db-cluster-identifier sample-cluster \
    --db-instance-class db.r4.xlarge \
    --db-instance-identifier sample-cluster-instance-2 \
    --engine docdb
```

输出：

```
{
    "DBInstance": {
        "DBInstanceStatus": "creating",
        "PendingModifiedValues": {
            "PendingCloudwatchLogsExports": {
                "LogTypesToEnable": [
                    "audit"
                ]
            }
        },
        "PubliclyAccessible": false,
        "PreferredBackupWindow": "00:00-00:30",
        "PromotionTier": 1,
        "EngineVersion": "3.6.0",
        "BackupRetentionPeriod": 3,
        "DBInstanceIdentifier": "sample-cluster-instance-2",
        "PreferredMaintenanceWindow": "tue:10:28-tue:10:58",
        "StorageEncrypted": false,
        "Engine": "docdb",
        "DBClusterIdentifier": "sample-cluster",
        "DBSubnetGroup": {
            "Subnets": [
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-4e26d263"
                },
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-afc329f4"
                },
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-53ab3636"
```

```
                },
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-991cb8d0"
                }
            ],
            "DBSubnetGroupDescription": "default",
            "SubnetGroupStatus": "Complete",
            "VpcId": "vpc-91280df6",
            "DBSubnetGroupName": "default"
        },
        "DBInstanceClass": "db.r4.xlarge",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ],
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance-2",
        "DbiResourceId": "db-XEKJLEMGRV5ZKCARUVA4HO3ITE"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的向集群添加 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbInstance。

`create-db-subnet-group`

以下代码示例演示了如何使用 `create-db-subnet-group`。

Amazon CLI

创建 Amazon DocumentDB 子网组

以下 `create-db-subnet-group` 示例创建一个名为 `sample-subnet-group` 的 Amazon DocumentDB 子网组。

```
aws docdb create-db-subnet-group \
    --db-subnet-group-description "a sample subnet group" \
    --db-subnet-group-name sample-subnet-group \
    --subnet-ids "subnet-29ab1025" "subnet-991cb8d0" "subnet-53ab3636"
```

输出：

```
{
    "DBSubnetGroup": {
        "SubnetGroupStatus": "Complete",
        "DBSubnetGroupName": "sample-subnet-group",
        "DBSubnetGroupDescription": "a sample subnet group",
        "VpcId": "vpc-91280df6",
        "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:sample-
subnet-group",
        "Subnets": [
            {
                "SubnetStatus": "Active",
                "SubnetIdentifier": "subnet-53ab3636",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2d"
                }
            },
            {
                "SubnetStatus": "Active",
                "SubnetIdentifier": "subnet-991cb8d0",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                }
            },
            {
                "SubnetStatus": "Active",
                "SubnetIdentifier": "subnet-29ab1025",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2c"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的创建 Amazon DocumentDB 子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbSubnetGroup。

delete-db-cluster-parameter-group

以下代码示例演示了如何使用 delete-db-cluster-parameter-group。

Amazon CLI

删除 Amazon DocumentDB 集群参数组

以下 delete-db-cluster-parameter-group 示例删除 Amazon DocumentDB 参数组 sample-parameter-group。

```
aws docdb delete-db-cluster-parameter-group \
    --db-cluster-parameter-group-name sample-parameter-group
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的删除 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbClusterParameterGroup。

delete-db-cluster-snapshot

以下代码示例演示了如何使用 delete-db-cluster-snapshot。

Amazon CLI

删除 Amazon DocumentDB 集群快照

以下 delete-db-cluster-snapshot 示例删除 Amazon DocumentDB 集群快照 sample-cluster-snapshot。

```
aws docdb delete-db-cluster-snapshot \
    --db-cluster-snapshot-identifier sample-cluster-snapshot
```

输出：

```
{
    "DBClusterSnapshot": {
        "DBClusterIdentifier": "sample-cluster",
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c",
            "us-west-2d"
        ],
        "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",
        "VpcId": "vpc-91280df6",
        "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-
snapshot:sample-cluster-snapshot",
        "EngineVersion": "3.6.0",
        "Engine": "docdb",
        "SnapshotCreateTime": "2019-03-18T18:27:14.794Z",
        "Status": "available",
        "MasterUsername": "master-user",
        "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
        "PercentProgress": 100,
        "StorageEncrypted": false,
        "SnapshotType": "manual",
        "Port": 0
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的删除集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbClusterSnapshot。

delete-db-cluster

以下代码示例演示了如何使用 delete-db-cluster。

Amazon CLI

删除 Amazon DocumentDB 集群

以下 delete-db-cluster 示例删除 Amazon DocumentDB 集群。sample-cluster在删除集群之前，不会对其进行备份。请注意：删除前，您必须删除所有与集群关联的实例。

```
aws docdb delete-db-cluster \
    --db-cluster-identifier sample-cluster \
```

```
    --skip-final-snapshot
```

输出：

```
{
    "DBCluster": {
        "DBClusterIdentifier": "sample-cluster",
        "DBSubnetGroup": "default",
        "EngineVersion": "3.6.0",
        "Engine": "docdb",
        "LatestRestorableTime": "2019-03-18T18:07:24.610Z",
        "PreferredMaintenanceWindow": "sun:20:30-sun:21:00",
        "StorageEncrypted": false,
        "EarliestRestorableTime": "2019-03-18T18:07:24.610Z",
        "Port": 27017,
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ],
        "MultiAZ": false,
        "MasterUsername": "master-user",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
        "Status": "available",
        "PreferredBackupWindow": "10:12-10:42",
        "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "AvailabilityZones": [
            "us-west-2c",
            "us-west-2b",
            "us-west-2a"
        ],
        "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "DbClusterResourceId": "cluster-L3R4YRSBUYDP4GLMTJ2WF5GH5Q",
        "ClusterCreateTime": "2019-03-18T18:06:34.616Z",
        "AssociatedRoles": [],
        "DBClusterParameterGroup": "default.docdb3.6",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "BackupRetentionPeriod": 1,
        "DBClusterMembers": []
    }
```

```
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的删除 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbCluster。

delete-db-instance

以下代码示例演示了如何使用 delete-db-instance。

Amazon CLI

删除 Amazon DocumentDB 实例

以下 delete-db-instance 示例删除 Amazon DocumentDB 实例 sample-cluster-instance-2。

```
aws docdb delete-db-instance \
    --db-instance-identifier sample-cluster-instance-2
```

输出：

```
{
    "DBInstance": {
        "DBSubnetGroup": {
            "Subnets": [
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-4e26d263"
                },
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-afc329f4"
                },
                {
                    "SubnetAvailabilityZone": {
```

```
                        "Name": "us-west-2d"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-53ab3636"
                },
                {
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetStatus": "Active",
                    "SubnetIdentifier": "subnet-991cb8d0"
                }
            ],
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "VpcId": "vpc-91280df6",
            "SubnetGroupStatus": "Complete"
        },
        "PreferredBackupWindow": "00:00-00:30",
        "InstanceCreateTime": "2019-03-18T18:37:33.709Z",
        "DBInstanceClass": "db.r4.xlarge",
        "DbiResourceId": "db-XEKJLEMGRV5ZKCARUVA4HO3ITE",
        "BackupRetentionPeriod": 3,
        "Engine": "docdb",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ],
        "AutoMinorVersionUpgrade": true,
        "PromotionTier": 1,
        "EngineVersion": "3.6.0",
        "Endpoint": {
            "Address": "sample-cluster-instance-2.corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
            "HostedZoneId": "ZNKXH85TT8WVW",
            "Port": 27017
        },
        "DBInstanceIdentifier": "sample-cluster-instance-2",
        "PreferredMaintenanceWindow": "tue:10:28-tue:10:58",
        "EnabledCloudwatchLogsExports": [
            "audit"
        ],
```

```
        "PendingModifiedValues": {},
        "DBInstanceStatus": "deleting",
        "PubliclyAccessible": false,
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance-2",
        "DBClusterIdentifier": "sample-cluster",
        "AvailabilityZone": "us-west-2c",
        "StorageEncrypted": false
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的删除 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbInstance。

`delete-db-subnet-group`

以下代码示例演示了如何使用 `delete-db-subnet-group`。

Amazon CLI

删除 Amazon DocumentDB 子网组

以下 `delete-db-subnet-group` 示例删除 Amazon DocumentDB 子网组 `sample-subnet-group`。

```
aws docdb delete-db-subnet-group \
    --db-subnet-group-name sample-subnet-group
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的删除 Amazon DocumentDB 子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbSubnetGroup。

`describe-db-cluster-parameter-groups`

以下代码示例演示了如何使用 `describe-db-cluster-parameter-groups`。

Amazon CLI

查看一个或多个 Amazon DocumentDB 集群参数组的详细信息

以下 `describe-db-cluster-parameter-groups` 示例显示 Amazon DocumentDB 集群参数组 `custom3-6-param-grp` 的详细信息。

```
aws docdb describe-db-cluster-parameter-groups \
    --db-cluster-parameter-group-name custom3-6-param-grp
```

输出：

```
{
    "DBClusterParameterGroups": [
        {
            "DBParameterGroupFamily": "docdb3.6",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:custom3-6-param-grp",
            "Description": "Custom docdb3.6 parameter group",
            "DBClusterParameterGroupName": "custom3-6-param-grp"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的查看 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterParameterGroups。

`describe-db-cluster-parameters`

以下代码示例演示了如何使用 `describe-db-cluster-parameters`。

Amazon CLI

查看 Amazon DocumentDB 集群参数组的详细参数列表。

以下 `describe-db-cluster-parameters` 示例列出 Amazon DocumentDB 参数组 custom3-6-param-grp 的参数。

```
aws docdb describe-db-cluster-parameters \
     --db-cluster-parameter-group-name custom3-6-param-grp
```

输出：

```
{
    "Parameters": [
        {
            "DataType": "string",
            "ParameterName": "audit_logs",
            "IsModifiable": true,
            "ApplyMethod": "pending-reboot",
            "Source": "system",
            "ApplyType": "dynamic",
            "AllowedValues": "enabled,disabled",
            "Description": "Enables auditing on cluster.",
            "ParameterValue": "disabled"
        },
        {
            "DataType": "string",
            "ParameterName": "tls",
            "IsModifiable": true,
            "ApplyMethod": "pending-reboot",
            "Source": "system",
            "ApplyType": "static",
            "AllowedValues": "disabled,enabled",
            "Description": "Config to enable/disable TLS",
            "ParameterValue": "enabled"
        },
        {
            "DataType": "string",
            "ParameterName": "ttl_monitor",
            "IsModifiable": true,
            "ApplyMethod": "pending-reboot",
            "Source": "user",
            "ApplyType": "dynamic",
            "AllowedValues": "disabled,enabled",
            "Description": "Enables TTL Monitoring",
            "ParameterValue": "enabled"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的查看 Amazon DocumentDB 集群参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterParameters。

describe-db-cluster-snapshot-attributes

以下代码示例演示了如何使用 describe-db-cluster-snapshot-attributes。

Amazon CLI

列出 Amazon DocumentDB 快照属性名称和值

以下 describe-db-cluster-snapshot-attributes 示例列出 Amazon DocumentDB 快照 sample-cluster-snapshot 的属性名称和值。

```
aws docdb describe-db-cluster-snapshot-attributes \
    --db-cluster-snapshot-identifier sample-cluster-snapshot
```

输出：

```
{
    "DBClusterSnapshotAttributesResult": {
        "DBClusterSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": []
            }
        ],
        "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 DescribeDBClusterSnapshotAttributes。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterSnapshotAttributes。

describe-db-cluster-snapshots

以下代码示例演示了如何使用 describe-db-cluster-snapshots。

Amazon CLI

描述 Amazon DocumentDB 快照

以下 `describe-db-cluster-snapshots` 示例显示 Amazon DocumentDB 快照 `sample-cluster-snapshot` 的详细信息。

```
aws docdb describe-db-cluster-snapshots \
    --db-cluster-snapshot-identifier sample-cluster-snapshot
```

输出：

```
{
    "DBClusterSnapshots": [
        {
            "AvailabilityZones": [
                "us-west-2a",
                "us-west-2b",
                "us-west-2c",
                "us-west-2d"
            ],
            "Status": "available",
            "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-
snapshot:sample-cluster-snapshot",
            "SnapshotCreateTime": "2019-03-15T20:41:26.515Z",
            "SnapshotType": "manual",
            "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",
            "DBClusterIdentifier": "sample-cluster",
            "MasterUsername": "master-user",
            "StorageEncrypted": false,
            "VpcId": "vpc-91280df6",
            "EngineVersion": "3.6.0",
            "PercentProgress": 100,
            "Port": 0,
            "Engine": "docdb",
            "ClusterCreateTime": "2019-03-15T20:29:58.836Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 DescribeDBClusterSnapshots。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterSnapshots。

describe-db-clusters

以下代码示例演示了如何使用 describe-db-clusters。

Amazon CLI

获取有关一个或多个 Amazon DocumentDB 集群的详细信息。

以下 describe-db-clusters 示例显示 Amazon DocumentDB 集群 sample-cluster 的详细信息。通过省略 --db-cluster-identifier 参数，您最多可以获得 100 个集群的信息。

```
aws docdb describe-db-clusters
    --db-cluster-identifier sample-cluster
```

输出：

```
{
    "DBClusters": [
        {
            "DBClusterParameterGroup": "default.docdb3.6",
            "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
            "PreferredBackupWindow": "00:00-00:30",
            "DBClusterIdentifier": "sample-cluster",
            "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
            "LatestRestorableTime": "2019-03-18T20:28:03.239Z",
            "MasterUsername": "master-user",
            "DBClusterMembers": [
                {
                    "PromotionTier": 1,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "IsClusterWriter": false,
                    "DBInstanceIdentifier": "sample-cluster"
                },
                {
                    "PromotionTier": 1,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "IsClusterWriter": true,
                    "DBInstanceIdentifier": "sample-cluster2"
                }
            ],
```

```
            "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
            "VpcSecurityGroups": [
                {
                    "VpcSecurityGroupId": "sg-77186e0d",
                    "Status": "active"
                }
            ],
            "Engine": "docdb",
            "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
            "DBSubnetGroup": "default",
            "MultiAZ": true,
            "AvailabilityZones": [
                "us-west-2a",
                "us-west-2c",
                "us-west-2b"
            ],
            "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",
            "DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
            "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster",
            "BackupRetentionPeriod": 3,
            "HostedZoneId": "ZNKXH85TT8WVW",
            "StorageEncrypted": false,
            "EnabledCloudwatchLogsExports": [
                "audit"
            ],
            "AssociatedRoles": [],
            "EngineVersion": "3.6.0",
            "Port": 27017,
            "Status": "available"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的描述 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusters。

`describe-db-engine-versions`

以下代码示例演示了如何使用 `describe-db-engine-versions`。

Amazon CLI

列出可用的 Amazon DocumentDB 引擎版本

以下 `describe-db-engine-versions` 示例列出所有可用的 Amazon DocumentDB 引擎版本。

```
aws docdb describe-db-engine-versions \
    --engine docdb
```

输出：

```
{
    "DBEngineVersions": [
        {
            "DBEngineVersionDescription": "DocDB version 1.0.200837",
            "DBParameterGroupFamily": "docdb3.6",
            "EngineVersion": "3.6.0",
            "ValidUpgradeTarget": [],
            "DBEngineDescription": "Amazon DocumentDB (with MongoDB compatibility)",
            "SupportsLogExportsToCloudwatchLogs": true,
            "Engine": "docdb",
            "ExportableLogTypes": [
                "audit"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 DescribeDBEngineVersions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbEngineVersions。

`describe-db-instances`

以下代码示例演示了如何使用 `describe-db-instances`。

Amazon CLI

查找有关预置 Amazon DocumentDB 实例的信息

以下 describe-db-instances 示例显示 Amazon DocumentDB 实例 sample-cluster-instance 的详细信息。通过省略 --db-instance-identifier 参数，您最多可以获得 100 个实例的信息。

```
aws docdb describe-db-instances \
    --db-instance-identifier sample-cluster-instance
```

输出：

```
{
    "DBInstances": [
        {
            "Endpoint": {
                "HostedZoneId": "ZNKXH85TT8WVW",
                "Address": "sample-cluster-instance.corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
                "Port": 27017
            },
            "PreferredBackupWindow": "00:00-00:30",
            "DBInstanceStatus": "available",
            "DBInstanceClass": "db.r4.large",
            "EnabledCloudwatchLogsExports": [
                "audit"
            ],
            "DBInstanceIdentifier": "sample-cluster-instance",
            "DBSubnetGroup": {
                "Subnets": [
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-4e26d263",
                        "SubnetAvailabilityZone": {
                            "Name": "us-west-2a"
                        }
                    },
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-afc329f4",
                        "SubnetAvailabilityZone": {
                            "Name": "us-west-2c"
                        }
                    },
                    {
                        "SubnetStatus": "Active",
```

```
                        "SubnetIdentifier": "subnet-53ab3636",
                        "SubnetAvailabilityZone": {
                            "Name": "us-west-2d"
                        }
                    },
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-991cb8d0",
                        "SubnetAvailabilityZone": {
                            "Name": "us-west-2b"
                        }
                    }
                ],
                "DBSubnetGroupName": "default",
                "SubnetGroupStatus": "Complete",
                "DBSubnetGroupDescription": "default",
                "VpcId": "vpc-91280df6"
            },
            "InstanceCreateTime": "2019-03-15T20:36:06.338Z",
            "Engine": "docdb",
            "StorageEncrypted": false,
            "AutoMinorVersionUpgrade": true,
            "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance",
            "PreferredMaintenanceWindow": "tue:08:39-tue:09:09",
            "VpcSecurityGroups": [
                {
                    "Status": "active",
                    "VpcSecurityGroupId": "sg-77186e0d"
                }
            ],
            "DBClusterIdentifier": "sample-cluster",
            "PendingModifiedValues": {},
            "BackupRetentionPeriod": 3,
            "PubliclyAccessible": false,
            "EngineVersion": "3.6.0",
            "PromotionTier": 1,
            "AvailabilityZone": "us-west-2c",
            "DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的描述 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbInstances。

`describe-db-subnet-groups`

以下代码示例演示了如何使用 `describe-db-subnet-groups`。

Amazon CLI

检索 Amazon DocumentDB 子网说明列表

以下 `describe-db-subnet-groups` 示例描述名为 `default` 的 Amazon DocumentDB 子网的详细信息。

```
aws docdb describe-db-subnet-groups \
    --db-subnet-group-name default
```

输出：

```
{
    "DBSubnetGroups": [
        {
            "VpcId": "vpc-91280df6",
            "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:default",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-4e26d263",
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-afc329f4",
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-53ab3636",
```

```
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-991cb8d0",
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    }
                }
            ],
            "DBSubnetGroupName": "default",
            "SubnetGroupStatus": "Complete",
            "DBSubnetGroupDescription": "default"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的描述子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbSubnetGroups。

describe-engine-default-cluster-parameters

以下代码示例演示了如何使用 describe-engine-default-cluster-parameters。

Amazon CLI

描述 Amazon DocumentDB 的默认引擎和系统参数信息

以下 describe-engine-default-cluster-parameters 示例显示 Amazon DocumentDB 参数组 docdb3.6 的默认引擎和系统参数信息的详细信息。

```
aws docdb describe-engine-default-cluster-parameters \
    --db-parameter-group-family docdb3.6
```

输出：

```
{
    "EngineDefaults": {
```

```
        "DBParameterGroupFamily": "docdb3.6",
        "Parameters": [
            {
                "ApplyType": "dynamic",
                "ParameterValue": "disabled",
                "Description": "Enables auditing on cluster.",
                "Source": "system",
                "DataType": "string",
                "MinimumEngineVersion": "3.6.0",
                "AllowedValues": "enabled,disabled",
                "ParameterName": "audit_logs",
                "IsModifiable": true
            },
            {
                "ApplyType": "static",
                "ParameterValue": "enabled",
                "Description": "Config to enable/disable TLS",
                "Source": "system",
                "DataType": "string",
                "MinimumEngineVersion": "3.6.0",
                "AllowedValues": "disabled,enabled",
                "ParameterName": "tls",
                "IsModifiable": true
            },
            {
                "ApplyType": "dynamic",
                "ParameterValue": "enabled",
                "Description": "Enables TTL Monitoring",
                "Source": "system",
                "DataType": "string",
                "MinimumEngineVersion": "3.6.0",
                "AllowedValues": "disabled,enabled",
                "ParameterName": "ttl_monitor",
                "IsModifiable": true
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 DescribeEngineDefaultClusterParameters。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngineDefaultClusterParameters。

describe-event-categories

以下代码示例演示了如何使用 describe-event-categories。

Amazon CLI

描述所有 Amazon DocumentDB 事件类别

以下 describe-event-categories 示例列出 Amazon DocumentDB 事件源类型 db-instance 的所有类别。

```
aws docdb describe-event-categories \
    --source-type db-cluster
```

输出：

```
{
    "EventCategoriesMapList": [
        {
            "SourceType": "db-cluster",
            "EventCategories": [
                "failover",
                "maintenance",
                "notification",
                "failure"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的查看事件类别。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventCategories。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

列出 Amazon DocumentDB 事件

以下 describe-events 示例列出过去 24 小时（1440 分钟）内的所有 Amazon DocumentDB 事件。

```
aws docdb describe-events \
    --duration 1440
```

此命令不生成任何输出。输出：

```
{
    "Events": [
        {
            "EventCategories": [
                "failover"
            ],
            "Message": "Started cross AZ failover to DB instance: sample-cluster",
            "Date": "2019-03-18T21:36:29.807Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster",
            "SourceIdentifier": "sample-cluster",
            "SourceType": "db-cluster"
        },
        {
            "EventCategories": [
                "availability"
            ],
            "Message": "DB instance restarted",
            "Date": "2019-03-18T21:36:40.793Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster",
            "SourceIdentifier": "sample-cluster",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [],
            "Message": "A new writer was promoted. Restarting database as a
 reader.",
            "Date": "2019-03-18T21:36:43.873Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "availability"
```

```
            ],
            "Message": "DB instance restarted",
            "Date": "2019-03-18T21:36:51.257Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "failover"
            ],
            "Message": "Completed failover to DB instance: sample-cluster",
            "Date": "2019-03-18T21:36:53.462Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster",
            "SourceIdentifier": "sample-cluster",
            "SourceType": "db-cluster"
        },
        {
            "Date": "2019-03-19T16:51:48.847Z",
            "EventCategories": [
                "configuration change"
            ],
            "Message": "Updated parameter audit_logs to enabled with apply method
 pending-reboot",
            "SourceIdentifier": "custom3-6-param-grp",
            "SourceType": "db-parameter-group"
        },
        {
            "EventCategories": [
                "configuration change"
            ],
            "Message": "Applying modification to database instance class",
            "Date": "2019-03-19T17:55:20.095Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "availability"
            ],
            "Message": "DB instance shutdown",
            "Date": "2019-03-19T17:56:31.127Z",
```

```
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "configuration change"
            ],
            "Message": "Finished applying modification to DB instance class",
            "Date": "2019-03-19T18:00:45.822Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "availability"
            ],
            "Message": "DB instance restarted",
            "Date": "2019-03-19T18:00:53.397Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "availability"
            ],
            "Message": "DB instance shutdown",
            "Date": "2019-03-19T18:23:36.045Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
        {
            "EventCategories": [
                "availability"
            ],
            "Message": "DB instance restarted",
            "Date": "2019-03-19T18:23:46.209Z",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
            "SourceIdentifier": "sample-cluster2",
            "SourceType": "db-instance"
        },
```

```
        {
            "Date": "2019-03-19T18:39:05.822Z",
            "EventCategories": [
                "configuration change"
            ],
            "Message": "Updated parameter ttl_monitor to enabled with apply method
 immediate",
            "SourceIdentifier": "custom3-6-param-grp",
            "SourceType": "db-parameter-group"
        },
        {
            "Date": "2019-03-19T18:39:48.067Z",
            "EventCategories": [
                "configuration change"
            ],
            "Message": "Updated parameter audit_logs to disabled with apply method
 immediate",
            "SourceIdentifier": "custom3-6-param-grp",
            "SourceType": "db-parameter-group"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的查看 Amazon DocumentDB 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-orderable-db-instance-options

以下代码示例演示了如何使用 describe-orderable-db-instance-options。

Amazon CLI

查找可以订购的 Amazon DocumentDB 实例选项

以下 describe-orderable-db-instance-options 示例列出某个地区的 Amazon DocumentDB 的所有实例选项。

```
aws docdb describe-orderable-db-instance-options \
    --engine docdb \
    --region us-east-1
```

输出：

```
{
    "OrderableDBInstanceOptions": [
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.16xlarge",
            "LicenseModel": "na",
            "Engine": "docdb"
        },
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.2xlarge",
```

```
            "LicenseModel": "na",
            "Engine": "docdb"
        },
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.4xlarge",
            "LicenseModel": "na",
            "Engine": "docdb"
        },
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.8xlarge",
            "LicenseModel": "na",
            "Engine": "docdb"
```

```
        },
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.large",
            "LicenseModel": "na",
            "Engine": "docdb"
        },
        {
            "Vpc": true,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "EngineVersion": "3.6.0",
            "DBInstanceClass": "db.r4.xlarge",
            "LicenseModel": "na",
            "Engine": "docdb"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的向集群添加 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrderableDbInstanceOptions。

`describe-pending-maintenance-actions`

以下代码示例演示了如何使用 `describe-pending-maintenance-actions`。

Amazon CLI

列出待处理的 Amazon DocumentDB 维护操作

以下 `describe-pending-maintenance-actions` 示例列出您所有待处理的 Amazon DocumentDB 维护操作。

```
aws docdb describe-pending-maintenance-actions
```

输出：

```
{
    "PendingMaintenanceActions": []
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的维护 Amazon DocumentDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePendingMaintenanceActions。

`failover-db-cluster`

以下代码示例演示了如何使用 `failover-db-cluster`。

Amazon CLI

强制将 Amazon DocumentDB 集群失效转移到副本

以下 `failover-db-cluster` 示例导致 Amazon DocumentDB 集群 sample-cluster 中的主要实例失效转移到副本。

```
aws docdb failover-db-cluster \
    --db-cluster-identifier sample-cluster
```

输出：

```
{
    "DBCluster": {
        "AssociatedRoles": [],
        "DBClusterIdentifier": "sample-cluster",
        "EngineVersion": "3.6.0",
        "DBSubnetGroup": "default",
        "MasterUsername": "master-user",
        "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",
        "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2c",
            "us-west-2b"
        ],
        "LatestRestorableTime": "2019-03-18T21:35:23.548Z",
        "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
        "PreferredBackupWindow": "00:00-00:30",
        "Port": 27017,
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-77186e0d",
                "Status": "active"
            }
        ],
        "StorageEncrypted": false,
        "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
        "MultiAZ": true,
        "Status": "available",
        "DBClusterMembers": [
            {
                "DBClusterParameterGroupStatus": "in-sync",
                "IsClusterWriter": false,
                "DBInstanceIdentifier": "sample-cluster",
                "PromotionTier": 1
            },
            {
                "DBClusterParameterGroupStatus": "in-sync",
```

```
                "IsClusterWriter": true,
                "DBInstanceIdentifier": "sample-cluster2",
                "PromotionTier": 2
            }
        ],
        "EnabledCloudwatchLogsExports": [
            "audit"
        ],
        "DBClusterParameterGroup": "default.docdb3.6",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
        "BackupRetentionPeriod": 3,
        "DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
        "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "Engine": "docdb"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 Amazon DocumentDB 时效转移。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 FailoverDbCluster。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 Amazon DocumentDB 资源上的所有标签

以下 `list-tags-for-resource` 示例列出 Amazon DocumentDB 集群 `sample-cluster` 上的所有标签。

```
aws docdb list-tags-for-resource \
    --resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster
```

输出：

```
{
    "TagList": [
        {
```

```
            "Key": "A",
            "Value": "ALPHA"
        },
        {
            "Key": "B",
            "Value": ""
        },
        {
            "Key": "C",
            "Value": "CHARLIE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的列出 Amazon DocumentDB 资源上的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

modify-db-cluster-parameter-group

以下代码示例演示了如何使用 modify-db-cluster-parameter-group。

Amazon CLI

修改 Amazon DocumentDB DB 集群参数组

以下 modify-db-cluster-parameter-group 示例通过将两个参数 audit_logs 和 ttl_monitor 设置为启用来修改 Amazon DocumentDB 集群参数组 custom3-6-param-grp。更改将在下次重新启动时生效。

```
aws docdb modify-db-cluster-parameter-group \
    --db-cluster-parameter-group-name custom3-6-param-grp \
    --
parameters ParameterName=audit_logs,ParameterValue=enabled,ApplyMethod=pending-
reboot \

  ParameterName=ttl_monitor,ParameterValue=enabled,ApplyMethod=pending-reboot
```

输出：

```
{
    "DBClusterParameterGroupName": "custom3-6-param-grp"
```

```
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的修改 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbClusterParameterGroup。

`modify-db-cluster-snapshot-attribute`

以下代码示例演示了如何使用 `modify-db-cluster-snapshot-attribute`。

Amazon CLI

示例 1：向 Amazon DocumentDB 快照添加属性

以下 `modify-db-cluster-snapshot-attribute` 示例向 Amazon DocumentDB 集群快照添加四个属性值。

```
aws docdb modify-db-cluster-snapshot-attribute \
    --db-cluster-snapshot-identifier sample-cluster-snapshot \
    --attribute-name restore \
    --values-to-add 123456789011 123456789012 123456789013
```

输出：

```
{
    "DBClusterSnapshotAttributesResult": {
        "DBClusterSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "123456789011",
                    "123456789012",
                    "123456789013"
                ]
            }
        ],
        "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
    }
}
```

示例 2：从 Amazon DocumentDB 快照中移除属性

以下 modify-db-cluster-snapshot-attribute 示例从 Amazon DocumentDB 集群快照中移除两个属性值。

```
aws docdb modify-db-cluster-snapshot-attribute \
    --db-cluster-snapshot-identifier sample-cluster-snapshot \
    --attribute-name restore \
    --values-to-remove 123456789012
```

输出：

```
{
    "DBClusterSnapshotAttributesResult": {
        "DBClusterSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "123456789011",
                    "123456789013"
                ]
            }
        ],
        "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的 ModifyDBClusterSnapshotAttribute。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbClusterSnapshotAttribute。

modify-db-cluster

以下代码示例演示了如何使用 modify-db-cluster。

Amazon CLI

修改 Amazon DocumentDB 集群

以下 modify-db-cluster 示例修改 Amazon DocumentDB 集群 sample-cluster，将自动备份的留存期限设为 7 天，并更改备份和维护的首选时段。所有更改将在下一个维护窗口应用。

```
aws docdb modify-db-cluster \
    --db-cluster-identifier sample-cluster \
    --no-apply-immediately \
    --backup-retention-period 7 \
    --preferred-backup-window 18:00-18:30 \
    --preferred-maintenance-window sun:20:00-sun:20:30
```

输出：

```
{
    "DBCluster": {
        "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "DBClusterMembers": [
            {
                "DBClusterParameterGroupStatus": "in-sync",
                "DBInstanceIdentifier": "sample-cluster",
                "IsClusterWriter": true,
                "PromotionTier": 1
            },
            {
                "DBClusterParameterGroupStatus": "in-sync",
                "DBInstanceIdentifier": "sample-cluster2",
                "IsClusterWriter": false,
                "PromotionTier": 2
            }
        ],
        "HostedZoneId": "ZNKXH85TT8WVW",
        "StorageEncrypted": false,
        "PreferredBackupWindow": "18:00-18:30",
        "MultiAZ": true,
        "EngineVersion": "3.6.0",
        "MasterUsername": "master-user",
        "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "DBSubnetGroup": "default",
        "LatestRestorableTime": "2019-03-18T22:08:13.408Z",
        "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",
        "PreferredMaintenanceWindow": "sun:20:00-sun:20:30",
        "AssociatedRoles": [],
        "EnabledCloudwatchLogsExports": [
            "audit"
        ],
```

```
        "Engine": "docdb",
        "DBClusterParameterGroup": "default.docdb3.6",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
        "BackupRetentionPeriod": 7,
        "DBClusterIdentifier": "sample-cluster",
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2c",
            "us-west-2b"
        ],
        "Status": "available",
        "DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
        "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-77186e0d",
                "Status": "active"
            }
        ],
        "Port": 27017
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的修改 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbCluster。

`modify-db-instance`

以下代码示例演示了如何使用 `modify-db-instance`。

Amazon CLI

修改 Amazon DocumentDB 实例

以下 `modify-db-instance` 示例修改 Amazon DocumentDB 实例 `sample-cluster2`，将其实例类更改为 `db.r4.4xlarge`，推广层更改为 5。更改会立即生效，但只有在实例状态变为可用后才能看到。

```
aws docdb modify-db-instance \
    --db-instance-identifier sample-cluster2 \
    --apply-immediately \
```

```
    --db-instance-class db.r4.4xlarge \
    --promotion-tier 5
```

输出：

```
{
    "DBInstance": {
        "EngineVersion": "3.6.0",
        "StorageEncrypted": false,
        "DBInstanceClass": "db.r4.large",
        "PreferredMaintenanceWindow": "mon:08:39-mon:09:09",
        "AutoMinorVersionUpgrade": true,
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-77186e0d",
                "Status": "active"
            }
        ],
        "PreferredBackupWindow": "18:00-18:30",
        "EnabledCloudwatchLogsExports": [
            "audit"
        ],
        "AvailabilityZone": "us-west-2f",
        "DBInstanceIdentifier": "sample-cluster2",
        "InstanceCreateTime": "2019-03-15T20:36:06.338Z",
        "Engine": "docdb",
        "BackupRetentionPeriod": 7,
        "DBSubnetGroup": {
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-4e26d263",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-afc329f4",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
```

```
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-53ab3636",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-991cb8d0",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetStatus": "Active"
                }
            ],
            "VpcId": "vpc-91280df6"
        },
        "PromotionTier": 2,
        "Endpoint": {
            "Address": "sample-cluster2.corcjozrlsfc.us-west-2.docdb.amazonaws.com",
            "HostedZoneId": "ZNKXH85TT8WVW",
            "Port": 27017
        },
        "DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA",
        "DBClusterIdentifier": "sample-cluster",
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
        "PendingModifiedValues": {
            "DBInstanceClass": "db.r4.4xlarge"
        },
        "PubliclyAccessible": false,
        "DBInstanceStatus": "available"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的修改 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbInstance。

modify-db-subnet-group

以下代码示例演示了如何使用 modify-db-subnet-group。

Amazon CLI

修改 Amazon DocumentDB 子网组

以下 modify-db-subnet-group 示例通过添加指定的子网和新的说明来修改子网组 sample-subnet-group。

```
aws docdb modify-db-subnet-group \
    --db-subnet-group-name sample-subnet-group \
    --subnet-ids subnet-b3806e8f subnet-53ab3636 subnet-991cb8d0 \
    --db-subnet-group-description "New subnet description"
```

输出：

```
{
    "DBSubnetGroup": {
        "DBSubnetGroupName": "sample-subnet-group",
        "SubnetGroupStatus": "Complete",
        "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:sample-
subnet-group",
        "VpcId": "vpc-91280df6",
        "DBSubnetGroupDescription": "New subnet description",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-b3806e8f",
                "SubnetStatus": "Active",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            },
            {
                "SubnetIdentifier": "subnet-53ab3636",
                "SubnetStatus": "Active",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2c"
                }
            },
            {
                "SubnetIdentifier": "subnet-991cb8d0",
```

```
                "SubnetStatus": "Active",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的修改 Amazon DocumentDB 子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbSubnetGroup。

`reboot-db-instance`

以下代码示例演示了如何使用 `reboot-db-instance`。

Amazon CLI

重启 Amazon DocumentDB 实例

以下 `reboot-db-instance` 示例重启 Amazon DocumentDB 实例 `sample-cluster2`。

```
aws docdb reboot-db-instance \
    --db-instance-identifier sample-cluster2
```

此命令不生成任何输出。输出：

```
{
    "DBInstance": {
        "PreferredBackupWindow": "18:00-18:30",
        "DBInstanceIdentifier": "sample-cluster2",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ],
        "DBSubnetGroup": {
            "VpcId": "vpc-91280df6",
            "Subnets": [
                {
```

```
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetIdentifier": "subnet-4e26d263"
                },
                {
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    },
                    "SubnetIdentifier": "subnet-afc329f4"
                },
                {
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    },
                    "SubnetIdentifier": "subnet-53ab3636"
                },
                {
                    "SubnetStatus": "Active",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetIdentifier": "subnet-991cb8d0"
                }
            ],
            "SubnetGroupStatus": "Complete",
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default"
        },
        "PendingModifiedValues": {},
        "Endpoint": {
            "Address": "sample-cluster2.corcjozrlsfc.us-west-2.docdb.amazonaws.com",
            "HostedZoneId": "ZNKXH85TT8WVW",
            "Port": 27017
        },
        "EnabledCloudwatchLogsExports": [
            "audit"
        ],
        "StorageEncrypted": false,
        "DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA",
        "AutoMinorVersionUpgrade": true,
```

```
        "Engine": "docdb",
        "InstanceCreateTime": "2019-03-15T20:36:06.338Z",
        "EngineVersion": "3.6.0",
        "PromotionTier": 5,
        "BackupRetentionPeriod": 7,
        "DBClusterIdentifier": "sample-cluster",
        "PreferredMaintenanceWindow": "mon:08:39-mon:09:09",
        "PubliclyAccessible": false,
        "DBInstanceClass": "db.r4.4xlarge",
        "AvailabilityZone": "us-west-2d",
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
        "DBInstanceStatus": "rebooting"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的重启 Amazon DocumentDB 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootDbInstance。

`remove-tags-from-resource`

以下代码示例演示了如何使用 `remove-tags-from-resource`。

Amazon CLI

从 Amazon DocumentDB 资源中移除标签

以下 `remove-tags-from-resource` 示例从 Amazon DocumentDB 集群 `sample-cluster` 中移除键名为 B 的标签。

```
aws docdb remove-tags-from-resource \
    --resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \
    --tag-keys B
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的从 Amazon DocumentDB 资源中移除标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromResource。

reset-db-cluster-parameter-group

以下代码示例演示了如何使用 reset-db-cluster-parameter-group。

Amazon CLI

在 Amazon DocumentDB 参数组中将指定的参数值重置为默认值

以下 reset-db-cluster-parameter-group 示例将 Amazon DocumentDB 参数组 custom3-6-param-grp 中的参数 ttl_monitor 重置为默认值。

```
aws docdb reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name custom3-6-param-grp \
    --parameters ParameterName=ttl_monitor,ApplyMethod=immediate
```

输出：

```
{
    "DBClusterParameterGroupName": "custom3-6-param-grp"
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的“标题”。

在 Amazon DocumentDB 参数组中将指定的参数值或所有参数值重置为默认值

以下 reset-db-cluster-parameter-group 示例将 Amazon DocumentDB 参数组 custom3-6-param-grp 中的所有参数重置为默认值。

```
aws docdb reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name custom3-6-param-grp \
    --reset-all-parameters
```

输出：

```
{
    "DBClusterParameterGroupName": "custom3-6-param-grp"
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的重启 Amazon DocumentDB 集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetDbClusterParameterGroup。

restore-db-cluster-from-snapshot

以下代码示例演示了如何使用 restore-db-cluster-from-snapshot。

Amazon CLI

从自动或手动快照还原 Amazon DocumentDB 集群

以下 restore-db-cluster-from-snapshot 示例从快照 rds:sample-cluster-2019-03-16-00-01 创建一个名为 sample-cluster-2019-03-16-00-01-restored 的新 Amazon DocumentDB 集群。

```
aws docdb restore-db-cluster-from-snapshot \
    --db-cluster-identifier sample-cluster-2019-03-16-00-01-restored \
    --engine docdb \
    --snapshot-identifier rds:sample-cluster-2019-03-16-00-01
```

输出：

```
{
    "DBCluster": {
        "ClusterCreateTime": "2019-03-19T18:45:01.857Z",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "Engine": "docdb",
        "DBClusterMembers": [],
        "MultiAZ": false,
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2c",
            "us-west-2b"
        ],
        "StorageEncrypted": false,
        "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjozrlsfc.us-west-2.docdb.amazonaws.com",
        "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjozrlsfc.us-west-2.docdb.amazonaws.com",
        "Port": 27017,
        "PreferredBackupWindow": "00:00-00:30",
        "DBSubnetGroup": "default",
        "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
        "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
```

```
        "DBClusterParameterGroup": "default.docdb3.6",
        "DbClusterResourceId": "cluster-XOO46Q3RH4LWSYNH3NMZKXPISU",
        "MasterUsername": "master-user",
        "EngineVersion": "3.6.0",
        "BackupRetentionPeriod": 3,
        "AssociatedRoles": [],
        "Status": "creating",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的从集群快照还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbClusterFromSnapshot。

`restore-db-cluster-to-point-in-time`

以下代码示例演示了如何使用 `restore-db-cluster-to-point-in-time`。

Amazon CLI

将 Amazon DocumentDB 集群从手动快照还原到某个时间点

以下 `restore-db-cluster-to-point-in-time` 示例使用 `sample-cluster-snapshot` 创建新的 Amazon DocumentDB 集群 `sample-cluster-pit`，并使用最新的可还原时间。

```
aws docdb restore-db-cluster-to-point-in-time \
    --db-cluster-identifier sample-cluster-pit \
    --source-db-cluster-identifier arn:aws:rds:us-
west-2:123456789012:cluster:sample-cluster \
    --use-latest-restorable-time
```

输出：

```
{
    "DBCluster": {
        "StorageEncrypted": false,
        "BackupRetentionPeriod": 3,
```

```
        "MasterUsername": "master-user",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "PreferredBackupWindow": "00:00-00:30",
        "MultiAZ": false,
        "DBClusterIdentifier": "sample-cluster-pit",
        "DBSubnetGroup": "default",
        "ClusterCreateTime": "2019-04-03T15:55:21.320Z",
        "AssociatedRoles": [],
        "DBClusterParameterGroup": "default.docdb3.6",
        "DBClusterMembers": [],
        "Status": "creating",
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2d",
            "us-west-2b"
        ],
        "ReaderEndpoint": "sample-cluster-pit.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "Port": 27017,
        "Engine": "docdb",
        "EngineVersion": "3.6.0",
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-77186e0d",
                "Status": "active"
            }
        ],
        "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
        "Endpoint": "sample-cluster-pit.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
        "DbClusterResourceId": "cluster-NLCABBXOSE2QPQ4GOLZIFWEPLM",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster-
pit"
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的将快照还原到某个时间点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbClusterToPointInTime。

start-db-cluster

以下代码示例演示了如何使用 start-db-cluster。

Amazon CLI

启动已停止的 Amazon DocumentDB 集群

以下 `start-db-cluster` 示例启动指定的 Amazon DocumentDB 集群。

```
aws docdb start-db-cluster \
    --db-cluster-identifier sample-cluster
```

输出：

```
{
    "DBCluster": {
        "ClusterCreateTime": "2019-03-19T18:45:01.857Z",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "Engine": "docdb",
        "DBClusterMembers": [],
        "MultiAZ": false,
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1c",
            "us-east-1f"
        ],
        "StorageEncrypted": false,
        "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjozrlsfc.us-east-1.docdb.amazonaws.com",
        "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjozrlsfc.us-east-1.docdb.amazonaws.com",
        "Port": 27017,
        "PreferredBackupWindow": "00:00-00:30",
        "DBSubnetGroup": "default",
        "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
        "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
        "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
        "DBClusterParameterGroup": "default.docdb3.6",
        "DbClusterResourceId": "cluster-XOO46Q3RH4LWSYNH3NMZKXPISU",
        "MasterUsername": "master-user",
        "EngineVersion": "3.6.0",
        "BackupRetentionPeriod": 3,
        "AssociatedRoles": [],
        "Status": "creating",
        "VpcSecurityGroups": [
```

```
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的停止和启动 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDbCluster。

stop-db-cluster

以下代码示例演示了如何使用 stop-db-cluster。

Amazon CLI

停止运行中的 Amazon DocumentDB 集群

以下 stop-db-cluster 示例停止指定的 Amazon DocumentDB 集群。

```
aws docdb stop-db-cluster \
    --db-cluster-identifier sample-cluster
```

输出：

```
{
    "DBCluster": {
        "ClusterCreateTime": "2019-03-19T18:45:01.857Z",
        "HostedZoneId": "ZNKXH85TT8WVW",
        "Engine": "docdb",
        "DBClusterMembers": [],
        "MultiAZ": false,
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1c",
            "us-east-1f"
        ],
        "StorageEncrypted": false,
        "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjozrlsfc.us-east-1.docdb.amazonaws.com",
```

```
        "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjozrlsfc.us-east-1.docdb.amazonaws.com",
        "Port": 27017,
        "PreferredBackupWindow": "00:00-00:30",
        "DBSubnetGroup": "default",
        "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
        "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
        "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
        "DBClusterParameterGroup": "default.docdb3.6",
        "DbClusterResourceId": "cluster-XOO46Q3RH4LWSYNH3NMZKXPISU",
        "MasterUsername": "master-user",
        "EngineVersion": "3.6.0",
        "BackupRetentionPeriod": 3,
        "AssociatedRoles": [],
        "Status": "creating",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "sg-77186e0d"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DocumentDB 开发人员指南》中的停止和启动 Amazon DocumentDB 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDbCluster。

使用 Amazon CLI 的 DynamoDB 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 DynamoDB 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-get-item

以下代码示例演示了如何使用 batch-get-item。

Amazon CLI

检索表中的多个项

以下 batch-get-items 示例使用一批三个 GetItem 请求从 MusicCollection 表中读取多个项，并请求该操作所用的读取容量单位数。该命令仅返回 AlbumTitle 属性。

```
aws dynamodb batch-get-item \
    --request-items file://request-items.json \
    --return-consumed-capacity TOTAL
```

的内容request-items.json：

```
{
    "MusicCollection": {
        "Keys": [
            {
                "Artist": {"S": "No One You Know"},
                "SongTitle": {"S": "Call Me Today"}
            },
            {
                "Artist": {"S": "Acme Band"},
                "SongTitle": {"S": "Happy Day"}
            },
            {
                "Artist": {"S": "No One You Know"},
                "SongTitle": {"S": "Scared of My Shadow"}
            }
        ],
        "ProjectionExpression":"AlbumTitle"
    }
}
```

输出：

```
{
    "Responses": {
        "MusicCollection": [
            {
                "AlbumTitle": {
                    "S": "Somewhat Famous"
                }
            },
            {
                "AlbumTitle": {
                    "S": "Blue Sky Blues"
                }
            },
            {
                "AlbumTitle": {
                    "S": "Louder Than Ever"
                }
            }
        ]
    },
    "UnprocessedKeys": {},
    "ConsumedCapacity": [
        {
            "TableName": "MusicCollection",
            "CapacityUnits": 1.5
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的批量操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetItem。

batch-write-item

以下代码示例演示了如何使用 batch-write-item。

Amazon CLI

向表中添加多个项

以下 batch-write-item 示例使用一批三个 PutItem 请求向 MusicCollection 表中添加三个新项。它还会请求有关操作所用的写入容量单位数以及操作修改的任何项集合的信息。

```
aws dynamodb batch-write-item \
    --request-items file://request-items.json \
    --return-consumed-capacity INDEXES \
    --return-item-collection-metrics SIZE
```

的内容request-items.json：

```
{
    "MusicCollection": [
        {
            "PutRequest": {
                "Item": {
                    "Artist": {"S": "No One You Know"},
                    "SongTitle": {"S": "Call Me Today"},
                    "AlbumTitle": {"S": "Somewhat Famous"}
                }
            }
        },
        {
            "PutRequest": {
                "Item": {
                    "Artist": {"S": "Acme Band"},
                    "SongTitle": {"S": "Happy Day"},
                    "AlbumTitle": {"S": "Songs About Life"}
                }
            }
        },
        {
            "PutRequest": {
                "Item": {
                    "Artist": {"S": "No One You Know"},
                    "SongTitle": {"S": "Scared of My Shadow"},
                    "AlbumTitle": {"S": "Blue Sky Blues"}
                }
            }
        }
    ]
}
```

输出：

```
{
```

```
    "UnprocessedItems": {},
    "ItemCollectionMetrics": {
        "MusicCollection": [
            {
                "ItemCollectionKey": {
                    "Artist": {
                        "S": "No One You Know"
                    }
                },
                "SizeEstimateRangeGB": [
                    0.0,
                    1.0
                ]
            },
            {
                "ItemCollectionKey": {
                    "Artist": {
                        "S": "Acme Band"
                    }
                },
                "SizeEstimateRangeGB": [
                    0.0,
                    1.0
                ]
            }
        ]
    },
    "ConsumedCapacity": [
        {
            "TableName": "MusicCollection",
            "CapacityUnits": 6.0,
            "Table": {
                "CapacityUnits": 3.0
            },
            "LocalSecondaryIndexes": {
                "AlbumTitleIndex": {
                    "CapacityUnits": 3.0
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的基本操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchWriteItem。

create-backup

以下代码示例演示了如何使用 create-backup。

Amazon CLI

创建现有 DynamoDB 表的备份

以下 create-backup 示例创建 MusicCollection 表的备份。

```
aws dynamodb create-backup \
    --table-name MusicCollection \
    --backup-name MusicCollectionBackup
```

输出：

```
{
    "BackupDetails": {
        "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/
backup/01576616366715-b4e58d3a",
        "BackupName": "MusicCollectionBackup",
        "BackupSizeBytes": 0,
        "BackupStatus": "CREATING",
        "BackupType": "USER",
        "BackupCreationDateTime": 1576616366.715
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBackup。

create-global-table

以下代码示例演示了如何使用 create-global-table。

Amazon CLI

创建全局表

以下 create-global-table 示例从指定独立的 Amazon 区域中的两个相同表创建全局表。

```
aws dynamodb create-global-table \
    --global-table-name MusicCollection \
    --replication-group RegionName=us-east-2 RegionName=us-east-1 \
    --region us-east-2
```

输出：

```
{
    "GlobalTableDescription": {
        "ReplicationGroup": [
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-east-1"
            }
        ],
        "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
        "CreationDateTime": 1576625818.532,
        "GlobalTableStatus": "CREATING",
        "GlobalTableName": "MusicCollection"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGlobalTable。

create-table

以下代码示例演示了如何使用 create-table。

Amazon CLI

示例 1：创建带标签的表

以下 create-table 示例使用指定的属性和键架构创建名为 MusicCollection 的表。此表使用预调配的吞吐量，并使用 Amazon 默认拥有的 CMK 进行静态加密。该命令还将标签应用于该表，其键为 Owner，值为 blueTeam。

```
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-
definitions AttributeName=Artist,AttributeType=S AttributeName=SongTitle,AttributeType=S
 \
    --key-
schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 \
    --tags Key=Owner,Value=blueTeam
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "WriteCapacityUnits": 5,
            "ReadCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "TableName": "MusicCollection",
        "TableStatus": "CREATING",
        "KeySchema": [
            {
                "KeyType": "HASH",
                "AttributeName": "Artist"
            },
            {
                "KeyType": "RANGE",
                "AttributeName": "SongTitle"
            }
        ],
        "ItemCount": 0,
```

```
        "CreationDateTime": "2020-05-26T16:04:41.627000-07:00",
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 2：在按需模式下创建表

以下示例使用按需模式（而不是预调配吞吐量模式）创建名为 MusicCollection 的表。这对于工作负载不可预测的表很有用。

```
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-
definitions AttributeName=Artist,AttributeType=S AttributeName=SongTitle,AttributeType=S
 \
    --key-
schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
```

```
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-05-27T11:44:10.807000-07:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 0,
            "WriteCapacityUnits": 0
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "BillingModeSummary": {
            "BillingMode": "PAY_PER_REQUEST"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 3：创建表并使用客户托管的 CMK 对其进行加密

以下示例创建一个名为 MusicCollection 的表并使用客户托管的 CMK 对其进行加密。

```
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-
definitions AttributeName=Artist,AttributeType=S AttributeName=SongTitle,AttributeType=S
 \
    --key-
schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=abcd1234-abcd-1234-
a123-ab1234a1b234
```

输出：

```
{
    "TableDescription": {
```

```
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-05-27T11:12:16.431000-07:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "SSEDescription": {
            "Status": "ENABLED",
            "SSEType": "KMS",
            "KMSMasterKeyArn": "arn:aws:kms:us-west-2:123456789012:key/abcd1234-
abcd-1234-a123-ab1234a1b234"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 4：创建具有本地二级索引的表

以下示例使用指定的属性和键架构来创建名为 MusicCollection 且其本地二级索引名为 AlbumTitleIndex 的表。

```
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-
definitions AttributeName=Artist,AttributeType=S AttributeName=SongTitle,AttributeType=S Att
 \
    --key-
schema AttributeName=Artist,KeyType=HASH AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --local-secondary-indexes \
        "[
            {
                \"IndexName\": \"AlbumTitleIndex\",
                \"KeySchema\": [
                    {\"AttributeName\": \"Artist\",\"KeyType\":\"HASH\"},
                    {\"AttributeName\": \"AlbumTitle\",\"KeyType\":\"RANGE\"}
                ],
                \"Projection\": {
                    \"ProjectionType\": \"INCLUDE\",
                    \"NonKeyAttributes\": [\"Genre\", \"Year\"]
                }
            }
        ]"
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
```

```
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "LocalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitleIndex",
                "KeySchema": [
                    {
                        "AttributeName": "Artist",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "AlbumTitle",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "INCLUDE",
                    "NonKeyAttributes": [
                        "Genre",
                        "Year"
                    ]
                },
                "IndexSizeBytes": 0,
```

```
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 5：创建具有全局二级索引的表

以下示例创建一个名为 GameScores 且其全局二级索引名为 GameTitleIndex 的表。基表的分区键为 UserId，排序键为 GameTitle，可以有效地找到特定游戏的单个用户的最佳分数，而 GSI 则具有分区键 GameTitle 和排序键 TopScore，允许您快速找到特定游戏的总体最高分。

```
aws dynamodb create-table \
    --table-name GameScores \
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S Att
 \
    --key-schema AttributeName=UserId,KeyType=HASH \
                 AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --global-secondary-indexes \
        "[
            {
                \"IndexName\": \"GameTitleIndex\",
                \"KeySchema\": [
                    {\"AttributeName\":\"GameTitle\",\"KeyType\":\"HASH\"},
                    {\"AttributeName\":\"TopScore\",\"KeyType\":\"RANGE\"}
                ],
                \"Projection\": {
                    \"ProjectionType\":\"INCLUDE\",
                    \"NonKeyAttributes\":[\"UserId\"]
                },
                \"ProvisionedThroughput\": {
                    \"ReadCapacityUnits\": 10,
                    \"WriteCapacityUnits\": 5
                }
            }
        ]"
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "GameTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "TopScore",
                "AttributeType": "N"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-05-26T17:28:15.602000-07:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "GlobalSecondaryIndexes": [
            {
                "IndexName": "GameTitleIndex",
                "KeySchema": [
```

```
                    {
                        "AttributeName": "GameTitle",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "TopScore",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "INCLUDE",
                    "NonKeyAttributes": [
                        "UserId"
                    ]
                },
                "IndexStatus": "CREATING",
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 10,
                    "WriteCapacityUnits": 5
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameTitleIndex"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 6：一次创建一个具有多个全局二级索引的表

以下示例创建一个名为 GameScores 且具有两个全局二级索引的表。GSI 架构通过文件传递，而不是通过命令行传递。

```
aws dynamodb create-table \
    --table-name GameScores \
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S Att
 \
```

```
    --key-
schema AttributeName=UserId,KeyType=HASH AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --global-secondary-indexes file://gsi.json
```

的内容gsi.json：

```
[
    {
        "IndexName": "GameTitleIndex",
        "KeySchema": [
            {
                "AttributeName": "GameTitle",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "TopScore",
                "KeyType": "RANGE"
            }
        ],
        "Projection": {
            "ProjectionType": "ALL"
        },
        "ProvisionedThroughput": {
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        }
    },
    {
        "IndexName": "GameDateIndex",
        "KeySchema": [
            {
                "AttributeName": "GameTitle",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "Date",
                "KeyType": "RANGE"
            }
        ],
        "Projection": {
            "ProjectionType": "ALL"
        },
```

```
        "ProvisionedThroughput": {
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        }
    }
]
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Date",
                "AttributeType": "S"
            },
            {
                "AttributeName": "GameTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "TopScore",
                "AttributeType": "N"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-08-04T16:40:55.524000-07:00",
        "ProvisionedThroughput": {
```

```
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "GlobalSecondaryIndexes": [
            {
                "IndexName": "GameTitleIndex",
                "KeySchema": [
                    {
                        "AttributeName": "GameTitle",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "TopScore",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "IndexStatus": "CREATING",
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 10,
                    "WriteCapacityUnits": 5
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameTitleIndex"
            },
            {
                "IndexName": "GameDateIndex",
                "KeySchema": [
                    {
                        "AttributeName": "GameTitle",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "Date",
```

```
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "IndexStatus": "CREATING",
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 5,
                    "WriteCapacityUnits": 5
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameDateIndex"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 7：创建启用了 Streams 的表

以下示例创建一个名为 GameScores 且启用了 DynamoDB Streams 的表。每个项的新旧映像都将写入流中。

```
aws dynamodb create-table \
    --table-name GameScores \
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S
 \
    --key-
schema AttributeName=UserId,KeyType=HASH AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --stream-specification StreamEnabled=TRUE,StreamViewType=NEW_AND_OLD_IMAGES
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
```

```
            {
                "AttributeName": "GameTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2020-05-27T10:49:34.056000-07:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "StreamSpecification": {
            "StreamEnabled": true,
            "StreamViewType": "NEW_AND_OLD_IMAGES"
        },
        "LatestStreamLabel": "2020-05-27T17:49:34.056",
        "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/stream/2020-05-27T17:49:34.056"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表的基本操作。

示例 8：创建启用了 Keys-Only Stream 的表

以下示例将创建一个名为 GameScores 且启用了 DynamoDB Streams 的表。仅将所修改项的键属性写入流中。

```
aws dynamodb create-table \
    --table-name GameScores \
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S
 \
    --key-
schema AttributeName=UserId,KeyType=HASH AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --stream-specification StreamEnabled=TRUE,StreamViewType=KEYS_ONLY
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "GameTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2023-05-25T18:45:34.140000+00:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
```

```
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "StreamSpecification": {
            "StreamEnabled": true,
            "StreamViewType": "KEYS_ONLY"
        },
        "LatestStreamLabel": "2023-05-25T18:45:34.140",
        "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/stream/2023-05-25T18:45:34.140",
        "DeletionProtectionEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的更改 DynamoDB Streams 的数据捕获。

示例 9：使用 Standard Infrequent Access 类创建表

以下示例创建名为 GameScores 的表并分配 Standard-Infrequent Access（DynamoDB 标准-IA）表类。此表类针对主要的存储成本进行了优化。

```
aws dynamodb create-table \
    --table-name GameScores \
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S
 \
    --key-
schema AttributeName=UserId,KeyType=HASH AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --table-class STANDARD_INFREQUENT_ACCESS
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "GameTitle",
```

```
                "AttributeType": "S"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2023-05-25T18:33:07.581000+00:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "TableClassSummary": {
            "TableClass": "STANDARD_INFREQUENT_ACCESS"
        },
        "DeletionProtectionEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的表类。

示例 10：创建启用了删除保护功能的表

以下示例创建一个名为 GameScores 的表并启用删除保护。

```
aws dynamodb create-table \
    --table-name GameScores \
```

```
    --attribute-
definitions AttributeName=UserId,AttributeType=S AttributeName=GameTitle,AttributeType=S
 \
    --key-
schema AttributeName=UserId,KeyType=HASH AttributeName=GameTitle,KeyType=RANGE \
    --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
    --deletion-protection-enabled
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "GameTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            }
        ],
        "TableName": "GameScores",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "GameTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": "2023-05-25T23:02:17.093000+00:00",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 10,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
```

```
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "DeletionProtectionEnabled": true
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用删除保护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTable。

delete-backup

以下代码示例演示了如何使用 delete-backup。

Amazon CLI

删除现有 DynamoDB 备份

以下 delete-backup 示例删除指定的现有备份。

```
aws dynamodb delete-backup \
    --backup-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/
backup/01576616366715-b4e58d3a
```

输出：

```
{
    "BackupDescription": {
        "BackupDetails": {
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01576616366715-b4e58d3a",
            "BackupName": "MusicCollectionBackup",
            "BackupSizeBytes": 0,
            "BackupStatus": "DELETED",
            "BackupType": "USER",
            "BackupCreationDateTime": 1576616366.715
        },
        "SourceTableDetails": {
            "TableName": "MusicCollection",
            "TableId": "b0c04bcc-309b-4352-b2ae-9088af169fe2",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "TableSizeBytes": 0,
```

```
            "KeySchema": [
                {
                    "AttributeName": "Artist",
                    "KeyType": "HASH"
                },
                {
                    "AttributeName": "SongTitle",
                    "KeyType": "RANGE"
                }
            ],
            "TableCreationDateTime": 1576615228.571,
            "ProvisionedThroughput": {
                "ReadCapacityUnits": 5,
                "WriteCapacityUnits": 5
            },
            "ItemCount": 0,
            "BillingMode": "PROVISIONED"
        },
        "SourceTableFeatureDetails": {}
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBackup。

delete-item

以下代码示例演示了如何使用 delete-item。

Amazon CLI

示例 1：删除项

以下 delete-item 示例从 MusicCollection 表中删除项，并请求有关已删除的项以及请求使用的容量的详细信息。

```
aws dynamodb delete-item \
    --table-name MusicCollection \
    --key file://key.json \
    --return-values ALL_OLD \
    --return-consumed-capacity TOTAL \
    --return-item-collection-metrics SIZE
```

的内容key.json：

```
{
    "Artist": {"S": "No One You Know"},
    "SongTitle": {"S": "Scared of My Shadow"}
}
```

输出：

```
{
    "Attributes": {
        "AlbumTitle": {
            "S": "Blue Sky Blues"
        },
        "Artist": {
            "S": "No One You Know"
        },
        "SongTitle": {
            "S": "Scared of My Shadow"
        }
    },
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 2.0
    },
    "ItemCollectionMetrics": {
        "ItemCollectionKey": {
            "Artist": {
                "S": "No One You Know"
            }
        },
        "SizeEstimateRangeGB": [
            0.0,
            1.0
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

示例 2：有条件地删除项

以下示例仅在某项的 ProductCategory 为 Sporting Goods 或 Gardening Supplies 且其价格介于 500 和 600 之间时，才会将其从 ProductCatalog 表中删除。它会返回有关已删除项的详细信息。

```
aws dynamodb delete-item \
    --table-name ProductCatalog \
    --key '{"Id":{"N":"456"}}' \
    --condition-expression "(ProductCategory IN (:cat1, :cat2)) and (#P between :lo
 and :hi)" \
    --expression-attribute-names file://names.json \
    --expression-attribute-values file://values.json \
    --return-values ALL_OLD
```

的内容names.json：

```
{
    "#P": "Price"
}
```

的内容values.json：

```
{
    ":cat1": {"S": "Sporting Goods"},
    ":cat2": {"S": "Gardening Supplies"},
    ":lo": {"N": "500"},
    ":hi": {"N": "600"}
}
```

输出：

```
{
    "Attributes": {
        "Id": {
            "N": "456"
        },
        "Price": {
            "N": "550"
        },
        "ProductCategory": {
            "S": "Sporting Goods"
        }
    }
```

```
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteItem。

delete-table

以下代码示例演示了如何使用 delete-table。

Amazon CLI

删除表

以下 delete-table 示例将删除 MusicCollection 表。

```
aws dynamodb delete-table \
    --table-name MusicCollection
```

输出：

```
{
    "TableDescription": {
        "TableStatus": "DELETING",
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableName": "MusicCollection",
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "WriteCapacityUnits": 5,
            "ReadCapacityUnits": 5
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的删除表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTable。

describe-backup

以下代码示例演示了如何使用 describe-backup。

Amazon CLI

获取有关表的现有备份的信息

以下 describe-backup 示例显示有关指定现有备份的信息。

```
aws dynamodb describe-backup \
    --backup-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/
backup/01576616366715-b4e58d3a
```

输出：

```
{
    "BackupDescription": {
        "BackupDetails": {
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01576616366715-b4e58d3a",
            "BackupName": "MusicCollectionBackup",
            "BackupSizeBytes": 0,
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupCreationDateTime": 1576616366.715
        },
        "SourceTableDetails": {
            "TableName": "MusicCollection",
            "TableId": "b0c04bcc-309b-4352-b2ae-9088af169fe2",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "TableSizeBytes": 0,
            "KeySchema": [
                {
                    "AttributeName": "Artist",
                    "KeyType": "HASH"
                },
                {
                    "AttributeName": "SongTitle",
                    "KeyType": "RANGE"
                }
            ],
            "TableCreationDateTime": 1576615228.571,
            "ProvisionedThroughput": {
                "ReadCapacityUnits": 5,
                "WriteCapacityUnits": 5
```

```
            },
            "ItemCount": 0,
            "BillingMode": "PROVISIONED"
        },
        "SourceTableFeatureDetails": {}
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBackup。

`describe-continuous-backups`

以下代码示例演示了如何使用 `describe-continuous-backups`。

Amazon CLI

获取有关 DynamoDB 表连续备份的信息

以下 `describe-continuous-backups` 示例显示有关 `MusicCollection` 表连续备份设置的详细信息。

```
aws dynamodb describe-continuous-backups \
    --table-name MusicCollection
```

输出：

```
{
    "ContinuousBackupsDescription": {
        "ContinuousBackupsStatus": "ENABLED",
        "PointInTimeRecoveryDescription": {
            "PointInTimeRecoveryStatus": "DISABLED"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按时间点还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeContinuousBackups。

describe-contributor-insights

以下代码示例演示了如何使用 describe-contributor-insights。

Amazon CLI

查看 DynamoDB 表的 Contributor Insights 设置

以下 describe-contributor-insights 示例显示 MusicCollection 表和 AlbumTitle-index 全局二级索引的 Contributor Insights 设置。

```
aws dynamodb describe-contributor-insights \
    --table-name MusicCollection \
    --index-name AlbumTitle-index
```

输出：

```
{
    "TableName": "MusicCollection",
    "IndexName": "AlbumTitle-index",
    "ContributorInsightsRuleList": [
        "DynamoDBContributorInsights-PKC-MusicCollection-1576629651520",
        "DynamoDBContributorInsights-SKC-MusicCollection-1576629651520",
        "DynamoDBContributorInsights-PKT-MusicCollection-1576629651520",
        "DynamoDBContributorInsights-SKT-MusicCollection-1576629651520"
    ],
    "ContributorInsightsStatus": "ENABLED",
    "LastUpdateDateTime": 1576629654.78
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 的 CloudWatch Contributor Insights 分析数据访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeContributorInsights。

describe-endpoints

以下代码示例演示了如何使用 describe-endpoints。

Amazon CLI

查看区域端点信息

以下 describe-endpoints 示例显示有关当前 Amazon 区域端点的详细信息。

```
aws dynamodb describe-endpoints
```

输出：

```
{
    "Endpoints": [
        {
            "Address": "dynamodb.us-west-2.amazonaws.com",
            "CachePeriodInMinutes": 1440
        }
    ]
}
```

有关更多信息，请参阅《Amazon 一般参考》中的 Amazon DynamoDB 端点和配额。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpoints。

describe-global-table-settings

以下代码示例演示了如何使用 describe-global-table-settings。

Amazon CLI

获取有关 DynamoDB 全局表设置的信息

以下 describe-global-table-settings 示例显示 MusicCollection 全局表的设置。

```
aws dynamodb describe-global-table-settings \
    --global-table-name MusicCollection
```

输出：

```
{
    "GlobalTableName": "MusicCollection",
    "ReplicaSettings": [
        {
            "RegionName": "us-east-1",
            "ReplicaStatus": "ACTIVE",
            "ReplicaProvisionedReadCapacityUnits": 10,
            "ReplicaProvisionedReadCapacityAutoScalingSettings": {
```

```
                "AutoScalingDisabled": true
            },
            "ReplicaProvisionedWriteCapacityUnits": 5,
            "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            }
        },
        {
            "RegionName": "us-east-2",
            "ReplicaStatus": "ACTIVE",
            "ReplicaProvisionedReadCapacityUnits": 10,
            "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            },
            "ReplicaProvisionedWriteCapacityUnits": 5,
            "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGlobalTableSettings。

describe-global-table

以下代码示例演示了如何使用 describe-global-table。

Amazon CLI

显示有关 DynamoDB 全局表的信息

以下 describe-global-table 示例显示有关 MusicCollection 全局表的详细信息。

```
aws dynamodb describe-global-table \
    --global-table-name MusicCollection
```

输出：

```
{
    "GlobalTableDescription": {
```

```
        "ReplicationGroup": [
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-east-1"
            }
        ],
        "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
        "CreationDateTime": 1576625818.532,
        "GlobalTableStatus": "ACTIVE",
        "GlobalTableName": "MusicCollection"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGlobalTable。

describe-limits

以下代码示例演示了如何使用 describe-limits。

Amazon CLI

查看预置容量限制

以下 describe-limits 示例显示您账户在当前 Amazon 区域的预置容量限制。

```
aws dynamodb describe-limits
```

输出：

```
{
    "AccountMaxReadCapacityUnits": 80000,
    "AccountMaxWriteCapacityUnits": 80000,
    "TableMaxReadCapacityUnits": 40000,
    "TableMaxWriteCapacityUnits": 40000
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 限制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLimits。

describe-table-replica-auto-scaling

以下代码示例演示了如何使用 describe-table-replica-auto-scaling。

Amazon CLI

查看全局表副本之间的自动扩缩设置

以下 describe-table-replica-auto-scaling 示例显示 MusicCollection 全局表副本之间的自动扩缩设置。

```
aws dynamodb describe-table-replica-auto-scaling \
    --table-name MusicCollection
```

输出：

```
{
    "TableAutoScalingDescription": {
        "TableName": "MusicCollection",
        "TableStatus": "ACTIVE",
        "Replicas": [
            {
                "RegionName": "us-east-1",
                "GlobalSecondaryIndexes": [],
                "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
```

```
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaStatus": "ACTIVE"
            },
            {
                "RegionName": "us-east-2",
                "GlobalSecondaryIndexes": [],
                "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
```

```
                            "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaStatus": "ACTIVE"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTableReplicaAutoScaling。

describe-table

以下代码示例演示了如何使用 describe-table。

Amazon CLI

描述表

以下 describe-table 示例描述 MusicCollection 表。

```
aws dynamodb describe-table \
    --table-name MusicCollection
```

输出：

```
{
    "Table": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
```

```
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "WriteCapacityUnits": 5,
            "ReadCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "TableName": "MusicCollection",
        "TableStatus": "ACTIVE",
        "KeySchema": [
            {
                "KeyType": "HASH",
                "AttributeName": "Artist"
            },
            {
                "KeyType": "RANGE",
                "AttributeName": "SongTitle"
            }
        ],
        "ItemCount": 0,
        "CreationDateTime": 1421866952.062
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的描述表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTable。

describe-time-to-live

以下代码示例演示了如何使用 describe-time-to-live。

Amazon CLI

查看表的生存时间设置

以下 describe-time-to-live 示例显示 MusicCollection 表的生存时间设置。

```
aws dynamodb describe-time-to-live \
    --table-name MusicCollection
```

输出：

```
{
    "TimeToLiveDescription": {
        "TimeToLiveStatus": "ENABLED",
        "AttributeName": "ttl"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的生存时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTimeToLive。

get-item

以下代码示例演示了如何使用 get-item。

Amazon CLI

示例 1：读取表中的项

以下 get-item 示例将从 MusicCollection 表中检索项。该表具有 hash-and-range 主键（Artist 和 SongTitle），因此，您必须指定这两个属性。该命令还请求有关操作所用的读取容量的信息。

```
aws dynamodb get-item \
    --table-name MusicCollection \
    --key file://key.json \
    --return-consumed-capacity TOTAL
```

的内容key.json：

```
{
    "Artist": {"S": "Acme Band"},
    "SongTitle": {"S": "Happy Day"}
}
```

输出：

```
{
    "Item": {
        "AlbumTitle": {
            "S": "Songs About Life"
```

```
        },
        "SongTitle": {
            "S": "Happy Day"
        },
        "Artist": {
            "S": "Acme Band"
        }
    },
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 0.5
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的读取项。

示例 2：使用一致性读取来读取项

以下示例使用强一致性读取从 MusicCollection 表中检索项。

```
aws dynamodb get-item \
    --table-name MusicCollection \
    --key file://key.json \
    --consistent-read \
    --return-consumed-capacity TOTAL
```

的内容key.json：

```
{
    "Artist": {"S": "Acme Band"},
    "SongTitle": {"S": "Happy Day"}
}
```

输出：

```
{
    "Item": {
        "AlbumTitle": {
            "S": "Songs About Life"
        },
        "SongTitle": {
            "S": "Happy Day"
```

```
        },
        "Artist": {
            "S": "Acme Band"
        }
    },
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 1.0
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的读取项。

示例 3：检索项的特定属性

以下示例使用投影表达式仅检索所需项的三个属性。

```
aws dynamodb get-item \
    --table-name ProductCatalog \
    --key '{"Id": {"N": "102"}}' \
    --projection-expression "#T, #C, #P" \
    --expression-attribute-names file://names.json
```

的内容names.json：

```
{
    "#T": "Title",
    "#C": "ProductCategory",
    "#P": "Price"
}
```

输出：

```
{
    "Item": {
        "Price": {
            "N": "20"
        },
        "Title": {
            "S": "Book 102 Title"
        },
        "ProductCategory": {
```

```
            "S": "Book"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的读取项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetItem。

list-backups

以下代码示例演示了如何使用 list-backups。

Amazon CLI

示例 1：列出所有现有 DynamoDB 备份

以下 list-backups 示例列出您现有的所有备份。

```
aws dynamodb list-backups
```

输出：

```
{
    "BackupSummaries": [
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
            "BackupName": "MusicCollectionBackup1",
            "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 170
        },
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
```

```
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-b2abc345",
            "BackupName": "MusicCollectionBackup2",
            "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 400
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

示例 2：列出特定时间范围内用户创建的备份

以下示例仅列出用户创建的 MusicCollection 表的备份（不是由 DynamoDB 自动创建的备份），其创建日期介于 2020 年 1 月 1 日至 2020 年 3 月 1 日之间。

```
aws dynamodb list-backups \
    --table-name MusicCollection \
    --time-range-lower-bound 1577836800 \
    --time-range-upper-bound 1583020800 \
    --backup-type USER
```

输出：

```
{
    "BackupSummaries": [
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
            "BackupName": "MusicCollectionBackup1",
            "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 170
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

示例 3：限制页面大小

以下示例将返回所有现有备份的列表，但在每次调用中仅检索一个项，必要时执行多次调用以获取整个列表。在对大量资源运行列表命令时，限制页面大小非常有用，使用默认页面大小 1000 时，可能会导致“超时”错误。

```
aws dynamodb list-backups \
    --page-size 1
```

输出：

```
{
    "BackupSummaries": [
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
            "BackupName": "MusicCollectionBackup1",
            "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 170
        },
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-b2abc345",
            "BackupName": "MusicCollectionBackup2",
            "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 400
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

示例 4：限制返回项的数量

以下示例将返回项的数量限制为 1。响应包含用于检索下一页结果的 NextToken 值。

```
aws dynamodb list-backups \
    --max-items 1
```

输出：

```
{
    "BackupSummaries": [
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
            "BackupName": "MusicCollectionBackup1",
            "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 170
        }
    ],
    "NextToken":
 "abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

示例 5：检索下一页结果

以下命令使用先前对 list-backups 命令的调用中的 NextToken 值来检索另一页结果。由于本例中的响应不包含 NextToken 值，因此，我们知道已经到达结果末尾。

```
aws dynamodb list-backups \
```

```
    --starting-
token abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

Output

```
{
    "BackupSummaries": [
        {
            "TableName": "MusicCollection",
            "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-b2abc345",
            "BackupName": "MusicCollectionBackup2",
            "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
            "BackupStatus": "AVAILABLE",
            "BackupType": "USER",
            "BackupSizeBytes": 400
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBackups。

list-contributor-insights

以下代码示例演示了如何使用 list-contributor-insights。

Amazon CLI

示例 1：查看 Contributor Insights 摘要列表

以下 list-contributor-insights 示例显示 Contributor Insights 摘要列表。

```
aws dynamodb list-contributor-insights
```

输出：

```
{
```

```
    "ContributorInsightsSummaries": [
        {
            "TableName": "MusicCollection",
            "IndexName": "AlbumTitle-index",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "ProductCatalog",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "Forum",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "Reply",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "Thread",
            "ContributorInsightsStatus": "ENABLED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 的 CloudWatch Contributor Insights 分析数据访问权限。

示例 2：限制返回项的数量

以下示例将返回项的数量限制为 4。响应包含用于检索下一页结果的 NextToken 值。

```
aws dynamodb list-contributor-insights \
    --max-results 4
```

输出：

```
{
    "ContributorInsightsSummaries": [
        {
            "TableName": "MusicCollection",
            "IndexName": "AlbumTitle-index",
```

```
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "ProductCatalog",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "Forum",
            "ContributorInsightsStatus": "ENABLED"
        }
    ],
    "NextToken":
 "abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 的 CloudWatch Contributor Insights 分析数据访问权限。

示例 3：检索下一页结果

以下命令使用先前对 `list-contributor-insights` 命令的调用中的 `NextToken` 值来检索另一页结果。由于本例中的响应不包含 `NextToken` 值，因此，我们知道已经到达结果末尾。

```
aws dynamodb list-contributor-insights \
    --max-results 4 \
    --next-
token abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

输出：

```
{
    "ContributorInsightsSummaries": [
        {
            "TableName": "Reply",
            "ContributorInsightsStatus": "ENABLED"
        },
        {
            "TableName": "Thread",
            "ContributorInsightsStatus": "ENABLED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 的 CloudWatch Contributor Insights 分析数据访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContributorInsights。

`list-global-tables`

以下代码示例演示了如何使用 `list-global-tables`。

Amazon CLI

列出现有的 DynamoDB 全局表

以下 `list-global-tables` 示例列出您现有的所有全局表。

```
aws dynamodb list-global-tables
```

输出：

```
{
    "GlobalTables": [
        {
            "GlobalTableName": "MusicCollection",
            "ReplicationGroup": [
                {
                    "RegionName": "us-east-2"
                },
                {
                    "RegionName": "us-east-1"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGlobalTables。

`list-tables`

以下代码示例演示了如何使用 `list-tables`。

Amazon CLI

示例 1：列出表

以下 `list-tables` 示例列出与当前 Amazon 账户和区域关联的所有表。

```
aws dynamodb list-tables
```

输出：

```
{
    "TableNames": [
        "Forum",
        "ProductCatalog",
        "Reply",
        "Thread"
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的列出表名称。

示例 2：限制页面大小

以下示例返回所有现有表的列表，但在每次调用中仅检索一个项，必要时执行多次调用以获取整个列表。在对大量资源运行列表命令时，限制页面大小非常有用，使用默认页面大小 1000 时，可能会导致“超时”错误。

```
aws dynamodb list-tables \
    --page-size 1
```

输出：

```
{
    "TableNames": [
        "Forum",
        "ProductCatalog",
        "Reply",
        "Thread"
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的列出表名称。

示例 3：限制返回项的数量

以下示例将返回项的数量限制为 2。响应包含用于检索下一页结果的 NextToken 值。

```
aws dynamodb list-tables \
    --max-items 2
```

输出：

```
{
    "TableNames": [
        "Forum",
        "ProductCatalog"
    ],
    "NextToken":
 "abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的列出表名称。

示例 4：检索下一页结果

以下命令将使用先前对 list-tables 命令的调用中的 NextToken 值来检索另一页结果。由于本例中的响应不包含 NextToken 值，因此，我们知道已经到达结果末尾。

```
aws dynamodb list-tables \
    --starting-
token abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

输出：

```
{
    "TableNames": [
        "Reply",
        "Thread"
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的列出表名称。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTables。

list-tags-of-resource

以下代码示例演示了如何使用 list-tags-of-resource。

Amazon CLI

例 1：列出 DynamoDB 资源的标签

以下 list-tags-of-resource 示例显示 MusicCollection 表的标签。

```
aws dynamodb list-tags-of-resource \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection
```

输出：

```
{
    "Tags": [
        {
            "Key": "Owner",
            "Value": "blueTeam"
        },
        {
            "Key": "Environment",
            "Value": "Production"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的标记 DynamoDB。

示例 2：限制返回标签的数量

以下示例将返回标签的数量限制为 1。响应包含用于检索下一页结果的 NextToken 值。

```
aws dynamodb list-tags-of-resource \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --max-items 1
```

输出：

```
{
    "Tags": [
        {
```

```
            "Key": "Owner",
            "Value": "blueTeam"
        }
    ],
    "NextToken":
 "abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的标记 DynamoDB。

示例 3：检索下一页结果

以下命令使用先前对 `list-tags-of-resource` 命令的调用中的 `NextToken` 值来检索另一页结果。由于本例中的响应不包含 `NextToken` 值，因此，我们知道已经到达结果末尾。

```
aws dynamodb list-tags-of-resource \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --starting-
token abCDeFGhiJKlmnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

输出：

```
{
    "Tags": [
        {
            "Key": "Environment",
            "Value": "Production"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的标记 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsOfResource。

put-item

以下代码示例演示了如何使用 `put-item`。

Amazon CLI

示例 1：向表中添加项

以下 put-item 示例将新项添加到 MusicCollection 表中。

```
aws dynamodb put-item \
    --table-name MusicCollection \
    --item file://item.json \
    --return-consumed-capacity TOTAL \
    --return-item-collection-metrics SIZE
```

的内容item.json：

```
{
    "Artist": {"S": "No One You Know"},
    "SongTitle": {"S": "Call Me Today"},
    "AlbumTitle": {"S": "Greatest Hits"}
}
```

输出：

```
{
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 1.0
    },
    "ItemCollectionMetrics": {
        "ItemCollectionKey": {
            "Artist": {
                "S": "No One You Know"
            }
        },
        "SizeEstimateRangeGB": [
            0.0,
            1.0
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

示例 2：有条件地覆盖表中的项

仅当 MusicCollection 表中的现有项具有值为 Greatest Hits 的 AlbumTitle 属性时，以下 put-item 示例才会覆盖该项。该命令将返回该项先前的值。

```
aws dynamodb put-item \
    --table-name MusicCollection \
    --item file://item.json \
    --condition-expression "#A = :A" \
    --expression-attribute-names file://names.json \
    --expression-attribute-values file://values.json \
    --return-values ALL_OLD
```

的内容item.json：

```
{
    "Artist": {"S": "No One You Know"},
    "SongTitle": {"S": "Call Me Today"},
    "AlbumTitle": {"S": "Somewhat Famous"}
}
```

的内容names.json：

```
{
    "#A": "AlbumTitle"
}
```

的内容values.json：

```
{
    ":A": {"S": "Greatest Hits"}
}
```

输出：

```
{
    "Attributes": {
        "AlbumTitle": {
            "S": "Greatest Hits"
        },
        "Artist": {
            "S": "No One You Know"
        },
        "SongTitle": {
            "S": "Call Me Today"
        }
```

```
    }
}
```

如果键已存在，您应看到以下输出：

```
A client error (ConditionalCheckFailedException) occurred when calling the PutItem
 operation: The conditional request failed.
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutItem。

query

以下代码示例演示了如何使用 query。

Amazon CLI

示例 1：查询表

以下 query 示例查询 MusicCollection 表中的项。该表具有 hash-and-range 主键（Artist 和 SongTitle），但此查询仅指定哈希键值。它返回名为“No One You Know”的艺术家的歌名。

```
aws dynamodb query \
    --table-name MusicCollection \
    --projection-expression "SongTitle" \
    --key-condition-expression "Artist = :v1" \
    --expression-attribute-values file://expression-attributes.json \
    --return-consumed-capacity TOTAL
```

的内容expression-attributes.json：

```
{
    ":v1": {"S": "No One You Know"}
}
```

输出：

```
{
    "Items": [
        {
            "SongTitle": {
```

```
                "S": "Call Me Today"
            },
            "SongTitle": {
                "S": "Scared of My Shadow"
            }
        }
    ],
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 0.5
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的查询。

示例 2：使用强一致性读取查询表并按降序遍历索引

以下示例执行与第一个示例相同的查询，但返回结果的顺序相反，并且使用强一致性读取。

```
aws dynamodb query \
    --table-name MusicCollection \
    --projection-expression "SongTitle" \
    --key-condition-expression "Artist = :v1" \
    --expression-attribute-values file://expression-attributes.json \
    --consistent-read \
    --no-scan-index-forward \
    --return-consumed-capacity TOTAL
```

的内容expression-attributes.json：

```
{
    ":v1": {"S": "No One You Know"}
}
```

输出：

```
{
    "Items": [
        {
            "SongTitle": {
                "S": "Scared of My Shadow"
```

```
            }
        },
        {
            "SongTitle": {
                "S": "Call Me Today"
            }
        }
    ],
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 1.0
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的查询。

示例 3：筛选出特定结果

以下示例查询 MusicCollection，但不包括 AlbumTitle 属性中含特定值的结果。请注意，这不会影响 ScannedCount 或 ConsumedCapacity，因为筛选器在读取项之后应用。

```
aws dynamodb query \
    --table-name MusicCollection \
    --key-condition-expression "#n1 = :v1" \
    --filter-expression "NOT (#n2 IN (:v2, :v3))" \
    --expression-attribute-names file://names.json \
    --expression-attribute-values file://values.json \
    --return-consumed-capacity TOTAL
```

的内容values.json：

```
{
    ":v1": {"S": "No One You Know"},
    ":v2": {"S": "Blue Sky Blues"},
    ":v3": {"S": "Greatest Hits"}
}
```

的内容names.json：

```
{
    "#n1": "Artist",
```

```
    "#n2": "AlbumTitle"
}
```

输出：

```
{
    "Items": [
        {
            "AlbumTitle": {
                "S": "Somewhat Famous"
            },
            "Artist": {
                "S": "No One You Know"
            },
            "SongTitle": {
                "S": "Call Me Today"
            }
        }
    ],
    "Count": 1,
    "ScannedCount": 2,
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 0.5
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的查询。

示例 4：仅检索项计数

以下示例检索与查询匹配的项计数，但不检索任何项本身。

```
aws dynamodb query \
    --table-name MusicCollection \
    --select COUNT \
    --key-condition-expression "Artist = :v1" \
    --expression-attribute-values file://expression-attributes.json
```

的内容expression-attributes.json：

```
{
    ":v1": {"S": "No One You Know"}
```

```
}
```

输出：

```
{
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": null
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的查询。

示例 5：查询索引

以下示例查询本地二级索引 AlbumTitleIndex。该查询返回基表中已投影到本地二级索引的所有属性。请注意，查询本地二级索引或全局二级索引时，您还必须使用 table-name 参数提供基表的名称。

```
aws dynamodb query \
    --table-name MusicCollection \
    --index-name AlbumTitleIndex \
    --key-condition-expression "Artist = :v1" \
    --expression-attribute-values file://expression-attributes.json \
    --select ALL_PROJECTED_ATTRIBUTES \
    --return-consumed-capacity INDEXES
```

的内容expression-attributes.json：

```
{
    ":v1": {"S": "No One You Know"}
}
```

输出：

```
{
    "Items": [
        {
            "AlbumTitle": {
                "S": "Blue Sky Blues"
            },
            "Artist": {
                "S": "No One You Know"
```

```
            },
            "SongTitle": {
                "S": "Scared of My Shadow"
            }
        },
        {
            "AlbumTitle": {
                "S": "Somewhat Famous"
            },
            "Artist": {
                "S": "No One You Know"
            },
            "SongTitle": {
                "S": "Call Me Today"
            }
        }
    ],
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 0.5,
        "Table": {
            "CapacityUnits": 0.0
        },
        "LocalSecondaryIndexes": {
            "AlbumTitleIndex": {
                "CapacityUnits": 0.5
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Query。

`restore-table-from-backup`

以下代码示例演示了如何使用 `restore-table-from-backup`。

Amazon CLI

从现有备份还原 DynamoDB 表

以下 restore-table-from-backup 示例从现有备份还原指定表。

```
aws dynamodb restore-table-from-backup \
    --target-table-name MusicCollection \
    --backup-arnarn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/
backup/01576616366715-b4e58d3a
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection2",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": 1576618274.326,
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection2",
```

```
        "TableId": "114865c9-5ef3-496c-b4d1-c4cbdd2d44fb",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED"
        },
        "RestoreSummary": {
            "SourceBackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01576616366715-b4e58d3a",
            "SourceTableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "RestoreDateTime": 1576616366.715,
            "RestoreInProgress": true
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按需备份和还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreTableFromBackup。

`restore-table-to-point-in-time`

以下代码示例演示了如何使用 `restore-table-to-point-in-time`。

Amazon CLI

将 DynamoDB 表还原到某个时间点

以下 `restore-table-to-point-in-time` 示例将 `MusicCollection` 表还原到指定的时间点。

```
aws dynamodb restore-table-to-point-in-time \
    --source-table-name MusicCollection \
    --target-table-name MusicCollectionRestore \
    --restore-date-time 1576622404.0
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
```

```
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollectionRestore",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "CREATING",
        "CreationDateTime": 1576623311.86,
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollectionRestore",
        "TableId": "befd9e0e-1843-4dc6-a147-d6d00e85cb1f",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED"
        },
        "RestoreSummary": {
            "SourceTableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
            "RestoreDateTime": 1576622404.0,
            "RestoreInProgress": true
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按时间点还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreTableToPointInTime。

scan

以下代码示例演示了如何使用 scan。

Amazon CLI

扫描表

以下 scan 示例扫描整个 MusicCollection 表，然后将结果范围缩小到艺术家“No One You Know”的歌曲。对于每个项，仅返回专辑名称和歌曲名称。

```
aws dynamodb scan \
    --table-name MusicCollection \
    --filter-expression "Artist = :a" \
    --projection-expression "#ST, #AT" \
    --expression-attribute-names file://expression-attribute-names.json \
    --expression-attribute-values file://expression-attribute-values.json
```

的内容expression-attribute-names.json：

```
{
    "#ST": "SongTitle",
    "#AT":"AlbumTitle"
}
```

的内容expression-attribute-values.json：

```
{
    ":a": {"S": "No One You Know"}
}
```

输出：

```
{
    "Count": 2,
    "Items": [
        {
            "SongTitle": {
                "S": "Call Me Today"
            },
            "AlbumTitle": {
```

```
                "S": "Somewhat Famous"
            }
        },
        {
            "SongTitle": {
                "S": "Scared of My Shadow"
            },
            "AlbumTitle": {
                "S": "Blue Sky Blues"
            }
        }
    ],
    "ScannedCount": 3,
    "ConsumedCapacity": null
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 中的扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Scan。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

为 DynamoDB 资源添加标签

以下 `tag-resource` 示例将带有键值对的标签添加到 `MusicCollection` 表。

```
aws dynamodb tag-resource \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --tags Key=Owner,Value=blueTeam
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的标记 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`transact-get-items`

以下代码示例演示了如何使用 `transact-get-items`。

Amazon CLI

从一个或多个表中以原子方式检索多个项

以下 transact-get-items 示例以原子方式检索多个项。

```
aws dynamodb transact-get-items \
    --transact-items file://transact-items.json \
    --return-consumed-capacity TOTAL
```

的内容transact-items.json：

```
[
    {
        "Get": {
            "Key": {
                "Artist": {"S": "Acme Band"},
                "SongTitle": {"S": "Happy Day"}
            },
            "TableName": "MusicCollection"
        }
    },
    {
        "Get": {
            "Key": {
                "Artist": {"S": "No One You Know"},
                "SongTitle": {"S": "Call Me Today"}
            },
            "TableName": "MusicCollection"
        }
    }
]
```

输出：

```
{
    "ConsumedCapacity": [
        {
            "TableName": "MusicCollection",
            "CapacityUnits": 4.0,
            "ReadCapacityUnits": 4.0
        }
```

```
    ],
    "Responses": [
        {
            "Item": {
                "AlbumTitle": {
                    "S": "Songs About Life"
                },
                "Artist": {
                    "S": "Acme Band"
                },
                "SongTitle": {
                    "S": "Happy Day"
                }
            }
        },
        {
            "Item": {
                "AlbumTitle": {
                    "S": "Somewhat Famous"
                },
                "Artist": {
                    "S": "No One You Know"
                },
                "SongTitle": {
                    "S": "Call Me Today"
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Transactions 管理复杂工作流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TransactGetItems。

`transact-write-items`

以下代码示例演示了如何使用 `transact-write-items`。

Amazon CLI

示例 1：以原子方式将项写入一个或多个表

以下 transact-write-items 示例更新一个项并删除另一个项。如果任一操作失败，或者任一项目包含 Rating 属性，则操作将失败。

```
aws dynamodb transact-write-items \
    --transact-items file://transact-items.json \
    --return-consumed-capacity TOTAL \
    --return-item-collection-metrics SIZE
```

transact-items.json 文件的内容：

```
[
    {
        "Update": {
            "Key": {
                "Artist": {"S": "Acme Band"},
                "SongTitle": {"S": "Happy Day"}
            },
            "UpdateExpression": "SET AlbumTitle = :newval",
            "ExpressionAttributeValues": {
                ":newval": {"S": "Updated Album Title"}
            },
            "TableName": "MusicCollection",
            "ConditionExpression": "attribute_not_exists(Rating)"
        }
    },
    {
        "Delete": {
            "Key": {
                "Artist": {"S": "No One You Know"},
                "SongTitle": {"S": "Call Me Today"}
            },
            "TableName": "MusicCollection",
            "ConditionExpression": "attribute_not_exists(Rating)"
        }
    }
]
```

输出：

```
{
    "ConsumedCapacity": [
        {
```

```
            "TableName": "MusicCollection",
            "CapacityUnits": 10.0,
            "WriteCapacityUnits": 10.0
        }
    ],
    "ItemCollectionMetrics": {
        "MusicCollection": [
            {
                "ItemCollectionKey": {
                    "Artist": {
                        "S": "No One You Know"
                    }
                },
                "SizeEstimateRangeGB": [
                    0.0,
                    1.0
                ]
            },
            {
                "ItemCollectionKey": {
                    "Artist": {
                        "S": "Acme Band"
                    }
                },
                "SizeEstimateRangeGB": [
                    0.0,
                    1.0
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Transactions 管理复杂工作流。

示例 2：使用客户端请求令牌以原子方式写入项

以下命令使用客户端请求令牌调用 `transact-write-items` 幂等，这意味着多个调用与单个调用具有相同的效果。

```
aws dynamodb transact-write-items \
    --transact-items file://transact-items.json \
```

```
    --client-request-token abc123
```

transact-items.json 文件的内容：

```
[
    {
        "Update": {
            "Key": {
                "Artist": {"S": "Acme Band"},
                "SongTitle": {"S": "Happy Day"}
            },
            "UpdateExpression": "SET AlbumTitle = :newval",
            "ExpressionAttributeValues": {
                ":newval": {"S": "Updated Album Title"}
            },
            "TableName": "MusicCollection",
            "ConditionExpression": "attribute_not_exists(Rating)"
        }
    },
    {
        "Delete": {
            "Key": {
                "Artist": {"S": "No One You Know"},
                "SongTitle": {"S": "Call Me Today"}
            },
            "TableName": "MusicCollection",
            "ConditionExpression": "attribute_not_exists(Rating)"
        }
    }
]
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Transactions 管理复杂工作流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TransactWriteItems。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从 DynamoDB 资源中移除标签

以下 untag-resource 示例从 MusicCollection 表移除键为 Owner 的标签。

```
aws dynamodb untag-resource \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --tag-keys Owner
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的标记 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-continuous-backups

以下代码示例演示了如何使用 update-continuous-backups。

Amazon CLI

更新 DynamoDB 表的连续备份设置

以下 update-continuous-backups 示例为 MusicCollection 表启用时间点还原。

```
aws dynamodb update-continuous-backups \
    --table-name MusicCollection \
    --point-in-time-recovery-specification PointInTimeRecoveryEnabled=true
```

输出：

```
{
    "ContinuousBackupsDescription": {
        "ContinuousBackupsStatus": "ENABLED",
        "PointInTimeRecoveryDescription": {
            "PointInTimeRecoveryStatus": "ENABLED",
            "EarliestRestorableDateTime": 1576622404.0,
            "LatestRestorableDateTime": 1576622404.0
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的按时间点还原 DynamoDB。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContinuousBackups。

update-contributor-insights

以下代码示例演示了如何使用 update-contributor-insights。

Amazon CLI

在表上启用 Contributor Insights

以下 update-contributor-insights 示例在 MusicCollection 表和 AlbumTitle-index 全局二级索引上启用 Contributor Insights。

```
aws dynamodb update-contributor-insights \
    --table-name MusicCollection \
    --index-name AlbumTitle-index \
    --contributor-insights-action ENABLE
```

输出：

```
{
    "TableName": "MusicCollection",
    "IndexName": "AlbumTitle-index",
    "ContributorInsightsStatus": "ENABLING"
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB 的 CloudWatch Contributor Insights 分析数据访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContributorInsights。

update-global-table-settings

以下代码示例演示了如何使用 update-global-table-settings。

Amazon CLI

更新 DynamoDB 全局表上的预置写入容量设置

以下 update-global-table-settings 示例将 MusicCollection 全局表的预置写入容量设置为 15。

```
aws dynamodb update-global-table-settings \
    --global-table-name MusicCollection \
    --global-table-provisioned-write-capacity-units 15
```

输出：

```
{
    "GlobalTableName": "MusicCollection",
    "ReplicaSettings": [
        {
            "RegionName": "eu-west-1",
            "ReplicaStatus": "UPDATING",
            "ReplicaProvisionedReadCapacityUnits": 10,
            "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            },
            "ReplicaProvisionedWriteCapacityUnits": 10,
            "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            }
        },
        {
            "RegionName": "us-east-1",
            "ReplicaStatus": "UPDATING",
            "ReplicaProvisionedReadCapacityUnits": 10,
            "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            },
            "ReplicaProvisionedWriteCapacityUnits": 10,
            "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            }
        },
        {
            "RegionName": "us-east-2",
            "ReplicaStatus": "UPDATING",
            "ReplicaProvisionedReadCapacityUnits": 10,
            "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
            },
            "ReplicaProvisionedWriteCapacityUnits": 10,
            "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                "AutoScalingDisabled": true
```

```
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGlobalTableSettings。

update-global-table

以下代码示例演示了如何使用 update-global-table。

Amazon CLI

更新 DynamoDB 全局表

以下 update-global-table 示例将指定区域中的副本添加到 MusicCollection 全局表。

```
aws dynamodb update-global-table \
    --global-table-name MusicCollection \
    --replica-updates Create={RegionName=eu-west-1}
```

输出：

```
{
    "GlobalTableDescription": {
        "ReplicationGroup": [
            {
                "RegionName": "eu-west-1"
            },
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-east-1"
            }
        ],
        "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
        "CreationDateTime": 1576625818.532,
        "GlobalTableStatus": "ACTIVE",
```

```
        "GlobalTableName": "MusicCollection"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGlobalTable。

update-item

以下代码示例演示了如何使用 update-item。

Amazon CLI

示例 1：更新表中的项

下面的 update-item 示例更新 MusicCollection 表的项目。它会添加一个新属性（Year）并修改 AlbumTitle 属性。响应中会返回更新后显示的项中的所有属性。

```
aws dynamodb update-item \
    --table-name MusicCollection \
    --key file://key.json \
    --update-expression "SET #Y = :y, #AT = :t" \
    --expression-attribute-names file://expression-attribute-names.json \
    --expression-attribute-values file://expression-attribute-values.json  \
    --return-values ALL_NEW \
    --return-consumed-capacity TOTAL \
    --return-item-collection-metrics SIZE
```

的内容key.json：

```
{
    "Artist": {"S": "Acme Band"},
    "SongTitle": {"S": "Happy Day"}
}
```

的内容expression-attribute-names.json：

```
{
    "#Y":"Year", "#AT":"AlbumTitle"
}
```

的内容expression-attribute-values.json：

```
{
    ":y":{"N": "2015"},
    ":t":{"S": "Louder Than Ever"}
}
```

输出：

```
{
    "Attributes": {
        "AlbumTitle": {
            "S": "Louder Than Ever"
        },
        "Awards": {
            "N": "10"
        },
        "Artist": {
            "S": "Acme Band"
        },
        "Year": {
            "N": "2015"
        },
        "SongTitle": {
            "S": "Happy Day"
        }
    },
    "ConsumedCapacity": {
        "TableName": "MusicCollection",
        "CapacityUnits": 3.0
    },
    "ItemCollectionMetrics": {
        "ItemCollectionKey": {
            "Artist": {
                "S": "Acme Band"
            }
        },
        "SizeEstimateRangeGB": [
            0.0,
            1.0
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

示例 2：有条件地更新项

以下示例将更新 MusicCollection 表中的项，但前提是现有项还没有 Year 属性。

```
aws dynamodb update-item \
    --table-name MusicCollection \
    --key file://key.json \
    --update-expression "SET #Y = :y, #AT = :t" \
    --expression-attribute-names file://expression-attribute-names.json \
    --expression-attribute-values file://expression-attribute-values.json  \
    --condition-expression "attribute_not_exists(#Y)"
```

的内容key.json：

```
{
    "Artist": {"S": "Acme Band"},
    "SongTitle": {"S": "Happy Day"}
}
```

的内容expression-attribute-names.json：

```
{
    "#Y":"Year",
    "#AT":"AlbumTitle"
}
```

的内容expression-attribute-values.json：

```
{
    ":y":{"N": "2015"},
    ":t":{"S": "Louder Than Ever"}
}
```

如果该项已有 Year 属性，DynamoDB 会返回以下输出。

```
An error occurred (ConditionalCheckFailedException) when calling the UpdateItem
 operation: The conditional request failed
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的写入项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateItem。

update-table-replica-auto-scaling

以下代码示例演示了如何使用 update-table-replica-auto-scaling。

Amazon CLI

更新全局表副本之间的自动扩缩设置

以下 update-table-replica-auto-scaling 示例更新指定全局表副本之间的写入容量自动扩缩设置。

```
aws dynamodb update-table-replica-auto-scaling \
    --table-name MusicCollection \
    --provisioned-write-capacity-auto-scaling-update file://auto-scaling-policy.json
```

的内容auto-scaling-policy.json：

```
{
    "MinimumUnits": 10,
    "MaximumUnits": 100,
    "AutoScalingDisabled": false,
    "ScalingPolicyUpdate": {
        "PolicyName": "DynamoDBWriteCapacityUtilization:table/MusicCollection",
        "TargetTrackingScalingPolicyConfiguration": {
            "TargetValue": 80
        }
    }
}
```

输出：

```
{
    "TableAutoScalingDescription": {
        "TableName": "MusicCollection",
        "TableStatus": "ACTIVE",
        "Replicas": [
            {
                "RegionName": "eu-central-1",
                "GlobalSecondaryIndexes": [],
                "ReplicaProvisionedReadCapacityAutoScalingSettings": {
```

```
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                    "MinimumUnits": 10,
                    "MaximumUnits": 100,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 80.0
                            }
                        }
                    ]
                },
                "ReplicaStatus": "ACTIVE"
            },
            {
                "RegionName": "us-east-1",
                "GlobalSecondaryIndexes": [],
                "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
```

```
                            "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
                },
                "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                    "MinimumUnits": 10,
                    "MaximumUnits": 100,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 80.0
                            }
                        }
                    ]
                },
                "ReplicaStatus": "ACTIVE"
            },
            {
                "RegionName": "us-east-2",
                "GlobalSecondaryIndexes": [],
                "ReplicaProvisionedReadCapacityAutoScalingSettings": {
                    "MinimumUnits": 5,
                    "MaximumUnits": 40000,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 70.0
                            }
                        }
                    ]
```

```
                },
                "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
                    "MinimumUnits": 10,
                    "MaximumUnits": 100,
                    "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
                    "ScalingPolicies": [
                        {
                            "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                            "TargetTrackingScalingPolicyConfiguration": {
                                "TargetValue": 80.0
                            }
                        }
                    ]
                },
                "ReplicaStatus": "ACTIVE"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的 DynamoDB 全局表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTableReplicaAutoScaling。

update-table

以下代码示例演示了如何使用 update-table。

Amazon CLI

示例 1：修改表的计费模式

以下 update-table 示例增加了 MusicCollection 表上的预置读取和写入容量。

```
aws dynamodb update-table \
    --table-name MusicCollection \
    --billing-mode PROVISIONED \
    --provisioned-throughput ReadCapacityUnits=15,WriteCapacityUnits=10
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "UPDATING",
        "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": "2020-07-28T13:18:18.921000-07:00",
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 15,
            "WriteCapacityUnits": 10
        },
        "TableSizeBytes": 182,
        "ItemCount": 2,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED",
            "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
        }
```

```
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的更新表。

示例 2：创建全局二级索引

下面的示例在 MusicCollection 表上添加了全局二级索引。

```
aws dynamodb update-table \
    --table-name MusicCollection \
    --attribute-definitions AttributeName=AlbumTitle,AttributeType=S \
    --global-secondary-index-updates file://gsi-updates.json
```

的内容gsi-updates.json：

```
[
    {
        "Create": {
            "IndexName": "AlbumTitle-index",
            "KeySchema": [
                {
                    "AttributeName": "AlbumTitle",
                    "KeyType": "HASH"
                }
            ],
            "ProvisionedThroughput": {
                "ReadCapacityUnits": 10,
                "WriteCapacityUnits": 10
            },
            "Projection": {
                "ProjectionType": "ALL"
            }
        }
    }
]
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
```

```
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "UPDATING",
        "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 15,
            "WriteCapacityUnits": 10
        },
        "TableSizeBytes": 182,
        "ItemCount": 2,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED",
            "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
        },
        "GlobalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitle-index",
                "KeySchema": [
                    {
```

```
                        "AttributeName": "AlbumTitle",
                        "KeyType": "HASH"
                    }
                ],
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "IndexStatus": "CREATING",
                "Backfilling": false,
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 10,
                    "WriteCapacityUnits": 10
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的更新表。

示例 3：在表上启用 DynamoDB Streams

以下命令在 `MusicCollection` 表上启用 DynamoDB Streams。

```
aws dynamodb update-table \
    --table-name MusicCollection \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_IMAGE
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
```

```
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "UPDATING",
        "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 15,
            "WriteCapacityUnits": 10
        },
        "TableSizeBytes": 182,
        "ItemCount": 2,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED",
            "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
        },
        "LocalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitleIndex",
                "KeySchema": [
                    {
                        "AttributeName": "Artist",
                        "KeyType": "HASH"
                    },
                    {
```

```
                        "AttributeName": "AlbumTitle",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "INCLUDE",
                    "NonKeyAttributes": [
                        "Year",
                        "Genre"
                    ]
                },
                "IndexSizeBytes": 139,
                "ItemCount": 2,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
            }
        ],
        "GlobalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitle-index",
                "KeySchema": [
                    {
                        "AttributeName": "AlbumTitle",
                        "KeyType": "HASH"
                    }
                ],
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "IndexStatus": "ACTIVE",
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 10,
                    "WriteCapacityUnits": 10
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
            }
        ],
        "StreamSpecification": {
            "StreamEnabled": true,
            "StreamViewType": "NEW_IMAGE"
```

```
        },
        "LatestStreamLabel": "2020-07-28T21:53:39.112",
        "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/stream/2020-07-28T21:53:39.112"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的更新表。

示例 4：启用服务器端加密

以下示例在 MusicCollection 表上启用服务器端加密。

```
aws dynamodb update-table \
    --table-name MusicCollection \
    --sse-specification Enabled=true,SSEType=KMS
```

输出：

```
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "TableName": "MusicCollection",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
```

```
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "ACTIVE",
        "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
        "ProvisionedThroughput": {
            "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 15,
            "WriteCapacityUnits": 10
        },
        "TableSizeBytes": 182,
        "ItemCount": 2,
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
        "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
        "BillingModeSummary": {
            "BillingMode": "PROVISIONED",
            "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
        },
        "LocalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitleIndex",
                "KeySchema": [
                    {
                        "AttributeName": "Artist",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "AlbumTitle",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "INCLUDE",
                    "NonKeyAttributes": [
                        "Year",
                        "Genre"
                    ]
                },
                "IndexSizeBytes": 139,
                "ItemCount": 2,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
            }
```

```
        ],
        "GlobalSecondaryIndexes": [
            {
                "IndexName": "AlbumTitle-index",
                "KeySchema": [
                    {
                        "AttributeName": "AlbumTitle",
                        "KeyType": "HASH"
                    }
                ],
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "IndexStatus": "ACTIVE",
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "ReadCapacityUnits": 10,
                    "WriteCapacityUnits": 10
                },
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
            }
        ],
        "StreamSpecification": {
            "StreamEnabled": true,
            "StreamViewType": "NEW_IMAGE"
        },
        "LatestStreamLabel": "2020-07-28T21:53:39.112",
        "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/stream/2020-07-28T21:53:39.112",
        "SSEDescription": {
            "Status": "UPDATING"
        }
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的更新表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTable。

update-time-to-live

以下代码示例演示了如何使用 update-time-to-live。

Amazon CLI

更新表的生存时间设置

以下 update-time-to-live 示例在指定表上启用生存时间设置。

```
aws dynamodb update-time-to-live \
    --table-name MusicCollection \
    --time-to-live-specification Enabled=true,AttributeName=ttl
```

输出：

```
{
    "TimeToLiveSpecification": {
        "Enabled": true,
        "AttributeName": "ttl"
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的生存时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTimeToLive。

使用 Amazon CLI 的 DynamoDB Streams 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 DynamoDB Streams 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-stream`

以下代码示例演示了如何使用 `describe-stream`。

Amazon CLI

获取有关 DynamoDB 流的信息

以下 `describe-stream` 命令显示有关指定 DynamoDB 流的信息。

```
aws dynamodbstreams describe-stream \
    --stream-arn arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576
```

输出：

```
{
    "StreamDescription": {
        "StreamArn": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576",
        "StreamLabel": "2019-10-22T18:02:01.576",
        "StreamStatus": "ENABLED",
        "StreamViewType": "NEW_AND_OLD_IMAGES",
        "CreationRequestDateTime": 1571767321.571,
        "TableName": "Music",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "SongTitle",
                "KeyType": "RANGE"
            }
        ],
        "Shards": [
            {
                "ShardId": "shardId-00000001571767321804-697ce3d2",
                "SequenceNumberRange": {
                    "StartingSequenceNumber": "400000000000642977831",
                    "EndingSequenceNumber": "400000000000642977831"
                }
```

```
            },
            {
                "ShardId": "shardId-00000001571780995058-40810d86",
                "SequenceNumberRange": {
                    "StartingSequenceNumber": "757400000000005655171150"
                },
                "ParentShardId": "shardId-00000001571767321804-697ce3d2"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Streams 捕获表活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStream。

get-records

以下代码示例演示了如何使用 get-records。

Amazon CLI

从 Dynamodb 流中获取记录

以下 get-records 命令使用指定的 Amazon DynamoDB 分片迭代器检索记录。

```
aws dynamodbstreams get-records \
    --shard-iterator "arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576|1|
AAAAAAAAAAGgM3YZ89vLZZxjmoQeo33r9M4x3+zmmTLsiL86MfrF4+B4EbsByi52InVmiONmy6xVW4IRcIIbs1zO7MNI
+CjNPlqQjnyRSAnfOwWmKhL1/KNParWSfz2odf780oOObIDIWRRMkt7+Hyzh9SD
+hFxFAWR5C7QIlOXPc8mRBfNIazfrVCjJK8/jsjCzsqNyXKzJbhh+GXCoxYN
+Kpmg4nyj1EAsYhbGL35muvHFoHjcyuynbsczbWaXNfThDwRAyvoTmc8XhHKtAWUbJiaVd8ZPtQwDsThCrmDRPIdmTRG
+w/lEGS05ha1qNP+Vl4+tuhz2TRnhnJo/pny9GI/yGpce97mWvSPr5KPwy+Dtcm5BHayBs
+PVYHITaTliInFlT
+LCwvaz1QH3MY3b8A05Z800wjpktm60iQqtMeDwN4NX6FrcxR34JoFKGsgR8XkHVJzz2xr1xqSJ12ycpNTyHnndusw==
```

输出：

```
{
    "Records": [
        {
```

```
            "eventID": "c3b5d798eef6215d42f8137b19a88e50",
            "eventName": "INSERT",
            "eventVersion": "1.1",
            "eventSource": "aws:dynamodb",
            "awsRegion": "us-west-1",
            "dynamodb": {
                "ApproximateCreationDateTime": 1571849028.0,
                "Keys": {
                    "Artist": {
                        "S": "No One You Know"
                    },
                    "SongTitle": {
                        "S": "Call Me Today"
                    }
                },
                "NewImage": {
                    "AlbumTitle": {
                        "S": "Somewhat Famous"
                    },
                    "Artist": {
                        "S": "No One You Know"
                    },
                    "Awards": {
                        "N": "1"
                    },
                    "SongTitle": {
                        "S": "Call Me Today"
                    }
                },
                "SequenceNumber": "700000000013256296913",
                "SizeBytes": 119,
                "StreamViewType": "NEW_AND_OLD_IMAGES"
            }
        },
        {
            "eventID": "878960a6967867e2da16b27380a27328",
            "eventName": "INSERT",
            "eventVersion": "1.1",
            "eventSource": "aws:dynamodb",
            "awsRegion": "us-west-1",
            "dynamodb": {
                "ApproximateCreationDateTime": 1571849029.0,
                "Keys": {
                    "Artist": {
```

```
                    "S": "Acme Band"
                },
                "SongTitle": {
                    "S": "Happy Day"
                }
            },
            "NewImage": {
                "AlbumTitle": {
                    "S": "Songs About Life"
                },
                "Artist": {
                    "S": "Acme Band"
                },
                "Awards": {
                    "N": "10"
                },
                "SongTitle": {
                    "S": "Happy Day"
                }
            },
            "SequenceNumber": "800000000013256297217",
            "SizeBytes": 100,
            "StreamViewType": "NEW_AND_OLD_IMAGES"
        }
    },
    {
        "eventID": "520fabde080e159fc3710b15ee1d4daa",
        "eventName": "MODIFY",
        "eventVersion": "1.1",
        "eventSource": "aws:dynamodb",
        "awsRegion": "us-west-1",
        "dynamodb": {
            "ApproximateCreationDateTime": 1571849734.0,
            "Keys": {
                "Artist": {
                    "S": "Acme Band"
                },
                "SongTitle": {
                    "S": "Happy Day"
                }
            },
            "NewImage": {
                "AlbumTitle": {
                    "S": "Updated Album Title"
```

```
                    },
                    "Artist": {
                        "S": "Acme Band"
                    },
                    "Awards": {
                        "N": "10"
                    },
                    "SongTitle": {
                        "S": "Happy Day"
                    }
                },
                "OldImage": {
                    "AlbumTitle": {
                        "S": "Songs About Life"
                    },
                    "Artist": {
                        "S": "Acme Band"
                    },
                    "Awards": {
                        "N": "10"
                    },
                    "SongTitle": {
                        "S": "Happy Day"
                    }
                },
                "SequenceNumber": "900000000013256687845",
                "SizeBytes": 170,
                "StreamViewType": "NEW_AND_OLD_IMAGES"
            }
        }
    ],
    "NextShardIterator": "arn:aws:dynamodb:us-west-1:123456789012:table/
Music/stream/2019-10-23T16:41:08.740|1|AAAAAAAAAAEhEI04jkFLW
+LKOwivjT8d/IHEh3iExV2xK00aTxEzVy1C1C7Kbb5+ZOW6bT9VQ2n1/
mrs7+PRiaOZCHJu7JHJVW7zlsqOi/ges3fw8GYEymyL+piEk35cx67rQqwKKyq
+Q6w9JyjreIOj4F2lWLV26lBwRTrIYC4IB7C3BZZK4715QwYdDxNdVHiSBRZX8UqoS6WOt0F87xZLNB9F/
NhYBLXi/wcGvAcBcC0TNIOH+N0NqwtoB/
FGCkNrf8YZ0xRoNN6RgGuVWHF3pxOhxEJeFZoSoJTIKeG9YcYxzi5Ci/
mhdtm7tBXnbw5c6xmsGsBqTirNjlDyJLcWl8Cl0UOLX63Ufo/5QliztcjEbKsQe28x8LM8o7VH1Is0fF/
ITt8awSA4igyJS0P87GN8Qri8kj8iaE35805jBHWF2wvwT6Iy2xGrR2r2HzYps9dwGOarVdEITaJfWzNoL4HajMhmREZ
+V04i1YIeHMXJfcwetNRuIbdQXfJht2NQZa4PVV6iknY6d19MrdbSTMKoqAuvp6g3Q2jH4t7GKCLWgodcPAn8g5+43Da
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Streams 捕获表活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRecords。

get-shard-iterator

以下代码示例演示了如何使用 get-shard-iterator。

Amazon CLI

获取分片迭代器

以下 get-shard-iterator 命令检索指定分片的分片迭代器。

```
aws dynamodbstreams get-shard-iterator \
    --stream-arn arn:aws:dynamodb:us-west-1:12356789012:table/Music/
stream/2019-10-22T18:02:01.576 \
    --shard-id shardId-00000001571780995058-40810d86 \
    --shard-iterator-type LATEST
```

输出：

```
{
    "ShardIterator": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576|1|
AAAAAAAAAAGgM3YZ89vLZZxjmoQeo33r9M4x3+zmmTLsiL86MfrF4+B4EbsByi52InVmiONmy6xVW4IRcIIbs1zO7MNI
+CjNPlqQjnyRSAnfOwWmKhL1/KNParWSfz2odf780oOObIDIWRRMkt7+Hyzh9SD
+hFxFAWR5C7QIlOXPc8mRBfNIazfrVCjJK8/jsjCzsqNyXKzJbhh+GXCoxYN
+Kpmg4nyj1EAsYhbGL35muvHFoHjcyuynbsczbWaXNfThDwRAyvoTmc8XhHKtAWUbJiaVd8ZPtQwDsThCrmDRPIdmTRG
+w/lEGS05ha1qNP+Vl4+tuhz2TRnhnJo/pny9GI/yGpce97mWvSPr5KPwy+Dtcm5BHayBs
+PVYHITaTliInFlT
+LCwvaz1QH3MY3b8A05Z800wjpktm60iQqtMeDwN4NX6FrcxR34JoFKGsgR8XkHVJzz2xr1xqSJ12ycpNTyHnndusw==
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Streams 捕获表活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetShardIterator。

`list-streams`

以下代码示例演示了如何使用 `list-streams`。

Amazon CLI

列出 DynamoDB 流

以下 `list-streams` 命令列出默认 Amazon 区域内的所有现有 Amazon DynamoDB Streams。

```
aws dynamodbstreams list-streams
```

输出：

```
{
    "Streams": [
        {
            "StreamArn": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576",
            "TableName": "Music",
            "StreamLabel": "2019-10-22T18:02:01.576"
        }
    ]
}
```

有关更多信息，请参阅《Amazon DynamoDB 开发人员指南》中的使用 DynamoDB Streams 捕获表活动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreams。

使用 Amazon CLI 的 Amazon EC2 示例

以下代码示例演示了如何通过将 Amazon Command Line Interface与 Amazon EC2 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

accept-address-transfer

以下代码示例演示了如何使用 accept-address-transfer。

Amazon CLI

接受转移到您的账户的弹性 IP 地址

以下 accept-address-transfer 示例接受将指定的弹性 IP 地址转移到您的账户。

```
aws ec2 accept-address-transfer \
    --address 100.21.184.216
```

输出：

```
{
    "AddressTransfer": {
        "PublicIp": "100.21.184.216",
        "AllocationId": "eipalloc-09ad461b0d03f6aaf",
        "TransferAccountId": "123456789012",
        "TransferOfferExpirationTimestamp": "2023-02-22T20:51:10.000Z",
        "TransferOfferAcceptedTimestamp": "2023-02-22T22:52:54.000Z",
        "AddressTransferStatus": "accepted"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的转移弹性 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptAddressTransfer。

accept-reserved-instances-exchange-quote

以下代码示例演示了如何使用 accept-reserved-instances-exchange-quote。

Amazon CLI

执行可转换预留实例交换

此示例执行指定可转换预留实例的交换。

命令:

```
aws ec2 accept-reserved-instances-exchange-quote --reserved-
instance-ids 7b8750c3-397e-4da4-bbcb-a45ebexample --target-
configurations OfferingId=b747b472-423c-48f3-8cee-679bcexample
```

输出：

```
{
  "ExchangeId": "riex-e68ed3c1-8bc8-4c17-af77-811afexample"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptReservedInstancesExchangeQuote。

accept-transit-gateway-peering-attachment

以下代码示例演示了如何使用 accept-transit-gateway-peering-attachment。

Amazon CLI

接受中转网关对等连接

以下 accept-transit-gateway-peering-attachment 示例接受指定的中转网关对等连接。--region 参数指定接受方中转网关所在的区域。

```
aws ec2 accept-transit-gateway-peering-attachment \
    --transit-gateway-attachment-id tgw-attach-4455667788aabbccd \
    --region us-east-2
```

输出：

```
{
    "TransitGatewayPeeringAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
        "RequesterTgwInfo": {
            "TransitGatewayId": "tgw-123abc05e04123abc",
            "OwnerId": "123456789012",
            "Region": "us-west-2"
```

```
        },
        "AccepterTgwInfo": {
            "TransitGatewayId": "tgw-11223344aabbcc112",
            "OwnerId": "123456789012",
            "Region": "us-east-2"
        },
        "State": "pending",
        "CreationTime": "2019-12-09T11:38:31.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关对等连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptTransitGatewayPeeringAttachment。

accept-transit-gateway-vpc-attachment

以下代码示例演示了如何使用 accept-transit-gateway-vpc-attachment。

Amazon CLI

接受将 VPC 连接到中转网关的请求。

以下 accept-transit-gateway-vpc-attachment 示例接受连接到指定网关的请求。

```
aws ec2 accept-transit-gateway-vpc-attachment \
    --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
        "VpcId": "vpc-07e8ffd50fEXAMPLE",
        "VpcOwnerId": "123456789012",
        "State": "pending",
        "SubnetIds": [
            "subnet-0752213d59EXAMPLE"
        ],
        "CreationTime": "2019-07-10T17:33:46.000Z",
```

```
        "Options": {
            "DnsSupport": "enable",
            "Ipv6Support": "disable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 VPC 的中转网关连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptTransitGatewayVpcAttachment。

accept-vpc-endpoint-connections

以下代码示例演示了如何使用 accept-vpc-endpoint-connections。

Amazon CLI

接受接口端点连接请求

此示例接受指定端点服务的指定端点连接请求。

命令:

```
aws ec2 accept-vpc-endpoint-connections --service-id vpce-svc-03d5ebb7d9579a2b3 --
vpc-endpoint-ids vpce-0c1308d7312217abc
```

输出：

```
{
  "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptVpcEndpointConnections。

accept-vpc-peering-connection

以下代码示例演示了如何使用 accept-vpc-peering-connection。

Amazon CLI

接受 VPC 对等连接

此示例接受指定的 VPC 对等连接请求。

命令:

```
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

输出：

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Provisioning",
      "Code": "provisioning"
    },
    "Tags": [],
    "AccepterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-44455566",
      "CidrBlock": "10.0.1.0/28"
    },
    "VpcPeeringConnectionId": "pcx-1a2b3c4d",
    "RequesterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-111abc45",
      "CidrBlock": "10.0.0.0/28"
    }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptVpcPeeringConnection。

advertise-byoip-cidr

以下代码示例演示了如何使用 advertise-byoip-cidr。

Amazon CLI

公告地址范围

以下 advertise-byoip-cidr 示例公告指定的公有 IPv4 地址范围。

```
aws ec2 advertise-byoip-cidr \
```

```
    --cidr 203.0.113.25/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "203.0.113.25/24",
        "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",
        "State": "provisioned"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdvertiseByoipCidr。

`allocate-address`

以下代码示例演示了如何使用 `allocate-address`。

Amazon CLI

示例 1：从 Amazon 的地址池中分配弹性 IP 地址

以下 `allocate-address` 示例分配弹性 IP 地址。Amazon EC2 从 Amazon 的地址池中选择地址。

```
aws ec2 allocate-address
```

输出：

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-01435ba59eEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2",
    "Domain": "vpc"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性 IP 地址。

示例 2：分配弹性 IP 地址并将其与网络边界组关联

以下 allocate-address 示例分配弹性 IP 地址并将其与指定的网络边界组关联。

```
aws ec2 allocate-address \
    --network-border-group us-west-2-lax-1
```

输出：

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-e03dd489ceEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2-lax-1",
    "Domain": "vpc"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性 IP 地址。

示例 3：从自己拥有的地址池中分配弹性 IP 地址

以下 allocate-address 示例从您引入 Amazon Web Services 账户的地址池中，分配弹性 IP 地址。Amazon EC2 从地址池中选择地址。

```
aws ec2 allocate-address \
    --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

输出：

```
{
    "AllocationId": "eipalloc-02463d08ceEXAMPLE",
    "NetworkBorderGroup": "us-west-2",
    "CustomerOwnedIp": "18.218.95.81",
    "CustomerOwnedIpv4Pool": "ipv4pool-ec2-1234567890abcdef0",
    "Domain": "vpc"
    "NetworkBorderGroup": "us-west-2",
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性 IP 地址。

示例 4：从 IPAM 池中分配弹性 IP 地址

以下 allocate-address 示例从 Amazon VPC IP 地址管理器（IPAM）池中分配特定的 /32 弹性 IP 地址。

```
aws ec2 allocate-address \
    --region us-east-1 \
    --ipam-pool-id ipam-pool-1234567890abcdef0 \
    --address 192.0.2.0
```

输出：

```
{
    "PublicIp": "192.0.2.0",
    "AllocationId": "eipalloc-abcdef01234567890",
    "PublicIpv4Pool": "ipam-pool-1234567890abcdef0",
    "NetworkBorderGroup": "us-east-1",
    "Domain": "vpc"
}
```

有关更多信息，请参阅《Amazon VPC IPAM User Guide》中的 Allocate sequential Elastic IP addresses from an IPAM pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateAddress。

allocate-hosts

以下代码示例演示了如何使用 allocate-hosts。

Amazon CLI

示例 1：分配专属主机

以下 allocate-hosts 示例在 eu-west-1a 可用区中分配一个专属主机，您可以在其上启动 m5.large 实例。默认情况下，专属主机仅接受目标实例启动，不支持主机恢复。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --quantity 1
```

输出：

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
```

```
    ]
}
```

示例 2：分配启用了自动置放和主机恢复的专属主机

以下 allocate-hosts 示例在启用了自动置放和主机恢复的 eu-west-1a 可用区中分配单个专属主机。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --auto-placement on \
    --host-recovery on \
    --quantity 1
```

输出：

```
{
     "HostIds": [
         "h-07879acf49EXAMPLE"
     ]
}
```

示例 3：分配带有标签的专属主机

以下 allocate-hosts 示例分配单个专属主机，并应用具有名为 purpose 的键和值为 production 的标签。

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --quantity 1 \
    --tag-specifications 'ResourceType=dedicated-
host,Tags={Key=purpose,Value=production}'
```

输出：

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
    ]
```

```
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的分配专属主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateHosts。

`allocate-ipam-pool-cidr`

以下代码示例演示了如何使用 `allocate-ipam-pool-cidr`。

Amazon CLI

从 IPAM 池分配 CIDR

以下 `allocate-ipam-pool-cidr` 示例从 IPAM 池分配 CIDR。

（Linux）：

```
aws ec2 allocate-ipam-pool-cidr \
    --ipam-pool-id ipam-pool-0533048da7d823723 \
    --netmask-length 24
```

（Windows）：

```
aws ec2 allocate-ipam-pool-cidr ^
   --ipam-pool-id ipam-pool-0533048da7d823723 ^
   --netmask-length 24
```

输出：

```
{
    "IpamPoolAllocation": {
        "Cidr": "10.0.0.0/24",
        "IpamPoolAllocationId": "ipam-pool-alloc-018ecc28043b54ba38e2cd99943cebfbd",
        "ResourceType": "custom",
        "ResourceOwner": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的手动将 CIDR 分配到池以预留 IP 地址空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateIpamPoolCidr。

apply-security-groups-to-client-vpn-target-network

以下代码示例演示了如何使用 apply-security-groups-to-client-vpn-target-network。

Amazon CLI

将安全组应用于 Client VPN 端点的目标网络

以下 apply-security-groups-to-client-vpn-target-network 示例将安全组 sg-01f6e627a89f4db32 应用于指定目标网络和 Client VPN 端点之间的关联。

```
aws ec2 apply-security-groups-to-client-vpn-target-network \
    --security-group-ids sg-01f6e627a89f4db32 \
    --vpc-id vpc-0e2110c2f324332e0 \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "SecurityGroupIds": [
        "sg-01f6e627a89f4db32"
    ]
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的目标网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ApplySecurityGroupsToClientVpnTargetNetwork。

assign-ipv6-addresses

以下代码示例演示了如何使用 assign-ipv6-addresses。

Amazon CLI

将特定 IPv6 地址分配给网络接口

此示例将指定的 IPv6 地址分配给指定的网络接口。

命令:

```
aws ec2 assign-ipv6-addresses --network-interface-id eni-38664473 --ipv6-
addresses 2001:db8:1234:1a00:3304:8879:34cf:4071 2001:db8:1234:1a00:9691:9503:25ad:1761
```

输出：

```
{
  "AssignedIpv6Addresses": [
      "2001:db8:1234:1a00:3304:8879:34cf:4071",
      "2001:db8:1234:1a00:9691:9503:25ad:1761"
  ],
  "NetworkInterfaceId": "eni-38664473"
}
```

将 Amazon 选择的 IPv6 地址分配给网络接口

此示例将两个 IPv6 地址分配给指定的网络接口。Amazon 会自动从子网的 IPv6 CIDR 块范围内的可用 IPv6 地址分配这些 IPv6 地址。

命令:

```
aws ec2 assign-ipv6-addresses --network-interface-id eni-38664473 --ipv6-address-
count 2
```

输出：

```
{
  "AssignedIpv6Addresses": [
      "2001:db8:1234:1a00:3304:8879:34cf:4071",
      "2001:db8:1234:1a00:9691:9503:25ad:1761"
  ],
  "NetworkInterfaceId": "eni-38664473"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssignIpv6Addresses。

assign-private-ip-addresses

以下代码示例演示了如何使用 assign-private-ip-addresses。

Amazon CLI

为网络接口分配特定的辅助私有 IP 地址

此示例将指定的辅助私有 IP 地址分配给指定的网络接口。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --private-ip-addresses 10.0.0.82
```

将 Amazon EC2 选择的辅助私有 IP 地址分配给网络接口

此示例将两个辅助私有 IP 地址分配给指定的网络接口。Amazon EC2 会从与网络接口关联的子网的 CIDR 块范围内的可用 IP 地址自动分配这些 IP 地址。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --secondary-private-ip-address-count 2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssignPrivateIpAddresses。

assign-private-nat-gateway-address

以下代码示例演示了如何使用 `assign-private-nat-gateway-address`。

Amazon CLI

将私有 IP 地址分配给私有 NAT 网关

以下 `assign-private-nat-gateway-address` 示例将两个私有 IP 地址分配给指定的私有 NAT 网关。

```
aws ec2 assign-private-nat-gateway-address \
    --nat-gateway-id nat-1234567890abcdef0 \
    --private-ip-address-count 2
```

输出：

```
{
```

```
    "NatGatewayId": "nat-1234567890abcdef0",
    "NatGatewayAddresses": [
        {
            "NetworkInterfaceId": "eni-0065a61b324d1897a",
            "IsPrimary": false,
            "Status": "assigning"
        },
        {
            "NetworkInterfaceId": "eni-0065a61b324d1897a",
            "IsPrimary": false,
            "Status": "assigning"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssignPrivateNatGatewayAddress。

associate-address

以下代码示例演示了如何使用 associate-address。

Amazon CLI

示例 1：关联弹性 IP 地址和实例

以下 associate-address 示例将弹性 IP 地址与指定的 EC2 实例相关联。

```
aws ec2 associate-address \
    --instance-id i-0b263919b6498b123 \
    --allocation-id eipalloc-64d5890a
```

输出：

```
{
    "AssociationId": "eipassoc-2bebb745"
}
```

示例 2：将弹性 IP 地址与网络接口相关联

以下 associate-address 示例将指定的弹性 IP 地址与指定的网络接口相关联。

```
aws ec2 associate-address
    --allocation-id eipalloc-64d5890a \
    --network-interface-id eni-1a2b3c4d
```

输出：

```
{
    "AssociationId": "eipassoc-2bebb745"
}
```

示例 3：将弹性 IP 地址与私有 IP 地址关联

以下 associate-address 示例将指定的弹性 IP 地址与指定网络接口中的指定私有 IP 地址相关联。

```
aws ec2 associate-address \
    --allocation-id eipalloc-64d5890a \
    --network-interface-id eni-1a2b3c4d \
    --private-ip-address 10.0.0.85
```

输出：

```
{
    "AssociationId": "eipassoc-2bebb745"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAddress。

associate-client-vpn-target-network

以下代码示例演示了如何使用 associate-client-vpn-target-network。

Amazon CLI

将目标网络与 Client VPN 端点相关联

以下 associate-client-vpn-target-network 示例将子网与指定的 Client VPN 端点相关联。

```
aws ec2 associate-client-vpn-target-network \
    --subnet-id subnet-0123456789abcabca \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "AssociationId": "cvpn-assoc-12312312312312312",
    "Status": {
        "Code": "associating"
    }
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的目标网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateClientVpnTargetNetwork。

associate-dhcp-options

以下代码示例演示了如何使用 associate-dhcp-options。

Amazon CLI

将 DHCP 选项集与 VPC 相关联

此示例将指定的 DHCP 选项集与指定的 VPC 相关联。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 associate-dhcp-options --dhcp-options-id dopt-d9070ebb --vpc-id vpc-a01106c2
```

将默认 DHCP 选项集与 VPC 相关联

此示例将默认 DHCP 选项集与指定的 VPC 相关联。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 associate-dhcp-options --dhcp-options-id default --vpc-id vpc-a01106c2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateDhcpOptions。

associate-iam-instance-profile

以下代码示例演示了如何使用 associate-iam-instance-profile。

Amazon CLI

将 IAM 实例配置文件与实例相关联

此示例将名为 admin-role 的 IAM 实例配置文件与实例 i-123456789abcde123 相关联。

命令:

```
aws ec2 associate-iam-instance-profile --instance-id i-123456789abcde123 --iam-instance-profile Name=admin-role
```

输出：

```
{
  "IamInstanceProfileAssociation": {
      "InstanceId": "i-123456789abcde123",
      "State": "associating",
      "AssociationId": "iip-assoc-0e7736511a163c209",
      "IamInstanceProfile": {
          "Id": "AIPAJBLK7RKJKWDXVHIEC",
          "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateIamInstanceProfile。

associate-instance-event-window

以下代码示例演示了如何使用 associate-instance-event-window。

Amazon CLI

示例 1：将一个或多个实例与事件窗口相关联

以下 associate-instance-event-window 示例将一个或多个实例与事件窗口相关联。

```
aws ec2 associate-instance-event-window \
```

```
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-1234567890abcdef0",
                "i-0598c7d356eba48d7"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 2：将实例标签与事件窗口相关联

以下 associate-instance-event-window 示例将实例标签与事件窗口相关联。输入 instance-event-window-id 参数以指定事件窗口。要关联实例标签，请指定 association-target 参数，并为参数值指定一个或多个标签。

```
aws ec2 associate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
```

```
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [
                {
                    "Key": "k2",
                    "Value": "v2"
                },
                {
                    "Key": "k1",
                    "Value": "v1"
                }
            ],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 3：将专属主机与事件窗口相关联

以下 associate-instance-event-window 示例将专属主机与事件窗口相关联。输入 instance-event-window-id 参数以指定事件窗口。要关联专属主机，请指定 --association-target 参数，并为参数值指定一个或多个专属主机 ID。

```
aws ec2 associate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "DedicatedHostIds=h-029fa35a02b99801d"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": [
```

```
                "h-029fa35a02b99801d"
            ]
        },
        "State": "creating"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateInstanceEventWindow。

associate-ipam-resource-discovery

以下代码示例演示了如何使用 associate-ipam-resource-discovery。

Amazon CLI

将资源发现与 IPAM 相关联

在此示例中，您是 IPAM 委派管理员，并且另一个 Amazon 账户已创建并与您共享资源发现，以便您可以使用 IPAM 管理和监控由其他账户拥有的资源 CIDR。

注意

要完成此请求，您需要资源发现 ID（可通过 describe-ipam-resource-discoveries 获取）和 IPAM ID（可通过 describe-ipams 获取）。您所关联的资源发现必须首先使用 Amazon RAM 与您的账户共享。您输入的 --region 必须与要与之关联的 IPAM 的主区域相匹配。

以下 associate-ipam-resource-discovery 示例将资源发现与 IPAM 相关联。

```
aws ec2 associate-ipam-resource-discovery \
    --ipam-id ipam-005f921c17ebd5107 \
    --ipam-resource-discovery-id ipam-res-disco-03e0406de76a044ee \
    --tag-specifications 'ResourceType=ipam-resource-discovery,Tags=[{Key=cost-
center,Value=cc123}]' \
    --region us-east-1
```

输出：

```
{
    {
        "IpamResourceDiscoveryAssociation": {
            "OwnerId": "320805250157",
```

```
            "IpamResourceDiscoveryAssociationId": "ipam-res-disco-
assoc-04382a6346357cf82",
            "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::320805250157:ipam-
resource-discovery-association/ipam-res-disco-assoc-04382a6346357cf82",
            "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
            "IpamId": "ipam-005f921c17ebd5107",
            "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
            "IpamRegion": "us-east-1",
            "IsDefault": false,
            "ResourceDiscoveryStatus": "active",
            "State": "associate-in-progress",
            "Tags": []
        }
    }
}
```

关联资源发现后，您可以监控和/或管理由其他账户创建的资源的 IP 地址。有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateIpamResourceDiscovery。

associate-nat-gateway-address

以下代码示例演示了如何使用 associate-nat-gateway-address。

Amazon CLI

将弹性 IP 地址与公共 NAT 网关相关联

以下 associate-nat-gateway-address 示例将指定的弹性 IP 地址与指定的公共 NAT 网关相关联。Amazon 会自动分配辅助私有 IPv4 地址。

```
aws ec2 associate-nat-gateway-address \
    --nat-gateway-id nat-1234567890abcdef0 \
    --allocation-ids eipalloc-0be6ecac95EXAMPLE
```

输出：

```
{
    "NatGatewayId": "nat-1234567890abcdef0",
    "NatGatewayAddresses": [
        {
```

```
            "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
            "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
            "IsPrimary": false,
            "Status": "associating"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateNatGatewayAddress。

associate-route-table

以下代码示例演示了如何使用 associate-route-table。

Amazon CLI

将路由表与子网相关联

此示例将指定的路由表与指定的子网相关联。

命令:

```
aws ec2 associate-route-table --route-table-id rtb-22574640 --subnet-
id subnet-9d4a7b6c
```

输出：

```
{
    "AssociationId": "rtbassoc-781d0d1a"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateRouteTable。

associate-security-group-vpc

以下代码示例演示了如何使用 associate-security-group-vpc。

Amazon CLI

将安全组与其它 VPC 关联

以下 associate-security-group-vpc 示例将指定的安全组与指定的 VPC 相关联。

```
aws ec2 associate-security-group-vpc \
    --group-id sg-04dbb43907d3f8a78 \
    --vpc-id vpc-0bf4c2739bc05a694
```

输出：

```
{
    "State": "associating"
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的将安全组与多个 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateSecurityGroupVpc。

associate-subnet-cidr-block

以下代码示例演示了如何使用 associate-subnet-cidr-block。

Amazon CLI

将 IPv6 CIDR 块与子网相关联

此示例将 IPv6 CIDR 块与指定子网相关联。

命令:

```
aws ec2 associate-subnet-cidr-block --subnet-id subnet-5f46ec3b --ipv6-cidr-
block 2001:db8:1234:1a00::/64
```

输出：

```
{
  "SubnetId": "subnet-5f46ec3b",
  "Ipv6CidrBlockAssociation": {
      "Ipv6CidrBlock": "2001:db8:1234:1a00::/64",
      "AssociationId": "subnet-cidr-assoc-3aa54053",
      "Ipv6CidrBlockState": {
          "State": "associating"
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateSubnetCidrBlock。

associate-transit-gateway-multicast-domain

以下代码示例演示了如何使用 associate-transit-gateway-multicast-domain。

Amazon CLI

将中转网关与组播域相关联

以下 associate-transit-gateway-multicast-domain 示例将指定的子网和连接与指定的组播域相关联。

```
aws ec2 associate-transit-gateway-multicast-domain \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
    --transit-gateway-attachment-id tgw-attach-028c1dd0f8f5cbe8e \
    --subnet-ids subnet-000de86e3b49c932a \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

输出：

```
{
    "Associations": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
        "TransitGatewayAttachmentId": "tgw-attach-028c1dd0f8f5cbe8e",
        "ResourceId": "vpc-01128d2c240c09bd5",
        "ResourceType": "vpc",
        "Subnets": [
            {
                "SubnetId": "subnet-000de86e3b49c932a",
                "State": "associating"
            }
        ]
    }
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Multicast domains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateTransitGatewayMulticastDomain。

associate-transit-gateway-route-table

以下代码示例演示了如何使用 associate-transit-gateway-route-table。

Amazon CLI

将中转网关路由表与中转网关连接相关联

以下示例将指定的中转网关路由表与指定的 VPC 连接相关联。

```
aws ec2 associate-transit-gateway-route-table \
    --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE \
    --transit-gateway-attachment-id tgw-attach-0b5968d3b6EXAMPLE
```

输出：

```
{
    "Association": {
        "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
        "ResourceId": "vpc-0065acced4EXAMPLE",
        "ResourceType": "vpc",
        "State": "associating"
    }
}
```

有关更多信息，请参阅《Amazon 中转网关指南》中的关联中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateTransitGatewayRouteTable。

associate-vpc-cidr-block

以下代码示例演示了如何使用 associate-vpc-cidr-block。

Amazon CLI

示例 1：将 Amazon 提供的 IPv6 CIDR 块与 VPC 相关联

以下 associate-vpc-cidr-block 示例将 IPv6 CIDR 块与指定的 VPC 相关联。

```
aws ec2 associate-vpc-cidr-block \
    --amazon-provided-ipv6-cidr-block \
    --ipv6-cidr-block-network-border-group us-west-2-lax-1  \
    --vpc-id vpc-8EXAMPLE
```

输出：

```
{
    "Ipv6CidrBlockAssociation": {
        "AssociationId": "vpc-cidr-assoc-0838ce7d9dEXAMPLE",
        "Ipv6CidrBlockState": {
            "State": "associating"
        },
        "NetworkBorderGroup": "us-west-2-lax-1"
    },
    "VpcId": "vpc-8EXAMPLE"
}
```

示例 2：将额外的 IPv4 CIDR 块与 VPC 相关联

以下 associate-vpc-cidr-block 示例将 IPv4 CIDR 块 10.2.0.0/16 与指定的 VPC 相关联。

```
aws ec2 associate-vpc-cidr-block \
    --vpc-id vpc-1EXAMPLE \
    --cidr-block 10.2.0.0/16
```

输出：

```
{
    "CidrBlockAssociation": {
        "AssociationId": "vpc-cidr-assoc-2EXAMPLE",
        "CidrBlock": "10.2.0.0/16",
        "CidrBlockState": {
            "State": "associating"
        }
    },
    "VpcId": "vpc-1EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateVpcCidrBlock。

attach-classic-link-vpc

以下代码示例演示了如何使用 attach-classic-link-vpc。

Amazon CLI

将 EC2-Classic 实例链接（连接）到 VPC

此示例通过 VPC 安全组 sg-12312312 将实例 i-1234567890abcdef0 链接到 VPC vpc-88888888。

命令:

```
aws ec2 attach-classic-link-vpc --instance-id  i-1234567890abcdef0 --vpc-
id vpc-88888888 --groups sg-12312312
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachClassicLinkVpc。

attach-internet-gateway

以下代码示例演示了如何使用 attach-internet-gateway。

Amazon CLI

将互联网网关连接到 VPC

以下 attach-internet-gateway 示例将指定的互联网网关连接到特定 VPC。

```
aws ec2 attach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC 用户指南》中的互联网网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachInternetGateway。

attach-network-interface

以下代码示例演示了如何使用 attach-network-interface。

Amazon CLI

示例 1：将网络接口连接到实例

以下 attach-network-interface 示例将指定的网络接口连接到指定的实例。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-0dc56a8d4640ad10a \
    --instance-id i-1234567890abcdef0 \
    --device-index 1
```

输出：

```
{
    "AttachmentId": "eni-attach-01a8fc87363f07cf9"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性网络接口。

示例 2：将网络接口连接到具有多个网卡的实例

以下 attach-network-interface 示例将指定的网络接口连接到指定的实例和网卡。

```
aws ec2 attach-network-interface \
    --network-interface-id eni-07483b1897541ad83 \
    --instance-id i-01234567890abcdef \
    --network-card-index 1 \
    --device-index 1
```

输出：

```
{
    "AttachmentId": "eni-attach-0fbd7ee87a88cd06c"
```

```
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性网络接口。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachNetworkInterface。

attach-verified-access-trust-provider

以下代码示例演示了如何使用 attach-verified-access-trust-provider。

Amazon CLI

将信任提供商连接到实例

以下 attach-verified-access-trust-provider 示例将指定的已验证访问信任提供商连接到指定的已验证访问实例。

```
aws ec2 attach-verified-access-trust-provider \
    --verified-access-instance-id vai-0ce000c0b7643abea \
    --verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

输出：

```
{
    "VerifiedAccessTrustProvider": {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "Description": "",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center",
        "PolicyReferenceName": "idc",
        "CreationTime": "2023-08-25T19:00:38",
        "LastUpdatedTime": "2023-08-25T19:00:38"
    },
    "VerifiedAccessInstance": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "",
        "VerifiedAccessTrustProviders": [
            {
                "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
                "TrustProviderType": "user",
                "UserTrustProviderType": "iam-identity-center"
            }
        ],
```

```
        "CreationTime": "2023-08-25T18:27:56",
        "LastUpdatedTime": "2023-08-25T18:27:56"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachVerifiedAccessTrustProvider。

attach-volume

以下代码示例演示了如何使用 attach-volume。

Amazon CLI

将卷连接到实例

此示例命令将卷 (vol-1234567890abcdef0) 作为 /dev/sdf 连接到实例 (i-01474ef662b89480)。

命令:

```
aws ec2 attach-volume --volume-id vol-1234567890abcdef0 --instance-
id i-01474ef662b89480 --device /dev/sdf
```

输出：

```
{
    "AttachTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "InstanceId": "i-01474ef662b89480",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "attaching",
    "Device": "/dev/sdf"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachVolume。

attach-vpn-gateway

以下代码示例演示了如何使用 attach-vpn-gateway。

Amazon CLI

将虚拟专用网关连接到 VPC

以下 attach-vpn-gateway 示例将指定的虚拟专用网关连接到指定的 VPC。

```
aws ec2 attach-vpn-gateway \
    --vpn-gateway-id vgw-9a4cacf3 \
    --vpc-id vpc-a01106c2
```

输出：

```
{
    "VpcAttachment": {
        "State": "attaching",
        "VpcId": "vpc-a01106c2"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachVpnGateway。

authorize-client-vpn-ingress

以下代码示例演示了如何使用 authorize-client-vpn-ingress。

Amazon CLI

为 Client VPN 端点添加授权规则

以下 authorize-client-vpn-ingress 示例添加了一个入口授权规则，该规则允许所有客户端访问互联网（0.0.0.0/0）。

```
aws ec2 authorize-client-vpn-ingress \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --target-network-cidr 0.0.0.0/0 \
    --authorize-all-groups
```

输出：

```
{
```

```
    "Status": {
        "Code": "authorizing"
    }
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的授权规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeClientVpnIngress。

authorize-security-group-egress

以下代码示例演示了如何使用 authorize-security-group-egress。

Amazon CLI

示例 1：添加允许指向特定地址范围的出站流量的规则

以下 authorize-security-group-egress 示例添加授予对 TCP 端口 80 上指定地址范围的访问权限的规则。

```
aws ec2 authorize-security-group-egress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0b15794cdb17bf29c",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": true,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "10.0.0.0/16"
        }
    ]
}
```

示例 2：添加允许指向特定安全组的出站流量的规则

以下 authorize-security-group-egress 示例添加授予对 TCP 端口 80 上指定安全组的访问权限的规则。

```
aws ec2 authorize-security-group-egress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs=[{GroupId=sg-0aad1c26bbeec5c22}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0b5dd815afcea9cc3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": true,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "ReferencedGroupInfo": {
                "GroupId": "sg-0aad1c26bbeec5c22",
                "UserId": "123456789012"
            }
        }
    ]
}
```

有关更多信息，请参阅 Amazon VPC 用户指南中的安全组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeSecurityGroupEgress。

authorize-security-group-ingress

以下代码示例演示了如何使用 authorize-security-group-ingress。

Amazon CLI

示例 1：添加允许入站 SSH 流量的规则

以下 authorize-security-group-ingress 示例将添加一个规则，该规则允许入站流量通过 TCP 端口 22（SSH）。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01afa97ef3e1bedfc",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "203.0.113.0/24"
        }
    ]
}
```

示例 2：添加允许来自其他安全组的入站 HTTP 流量的规则

以下 authorize-security-group-ingress 示例将添加一个规则，该规则允许源安全组通过 TCP 端口 80 进行入站访问sg-1a2b3c4d。源组必须在同一个 VPC 中，或者在对等 VPC 中（需要 VPC 对等连接）。允许的传入流量基于与源安全组相关联实例的私有 IP 地址（而不是公有 IP 或弹性 IP 地址）。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 80 \
    --source-group sg-1a2b3c4d
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01f4be99110f638a7",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "ReferencedGroupInfo": {
                "GroupId": "sg-1a2b3c4d",
                "UserId": "123456789012"
            }
        }
    ]
}
```

示例 3：在同一个调用中添加多个规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加两个入站规则，一个允许在 TCP 端口 3389（RDP）上进行入站访问，另一个启用 ping/ICMP。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=3389,ToPort=3389,IpRanges=[{CidrIp=172.31.0.0/16}]"
 "IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=172.31.0.0/16}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-00e06e5d3690f29f3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
```

```
            "ToPort": 3389,
            "CidrIpv4": "172.31.0.0/16"
        },
        {
            "SecurityGroupRuleId": "sgr-0a133dd4493944b87",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv4": "172.31.0.0/16"
        }
    ]
}
```

示例 4：为 ICMP 流量添加规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加一个入站规则，该规则允许来自任何地方的 ICMP 消息 Destination Unreachable: Fragmentation Needed and Don't Fragment was Set（类型 3，代码 4）。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0de3811019069b787",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmp",
            "FromPort": 3,
            "ToPort": 4,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
```

```
}
```

示例 5：为 IPv6 流量添加规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加一个入站规则，该规则允许从 IPv6 范围 2001:db8:1234:1a00::/64 进行 SSH 访问（端口 22）。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=22,ToPort=22,Ipv6Ranges=[{CidrIpv6=2001:db8:1234:1a00::/64}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0455bc68b60805563",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv6": "2001:db8:1234:1a00::/64"
        }
    ]
}
```

示例 6：为 ICMPv6 流量添加规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加一个入站规则，该规则允许来自任何地方的 ICMPv6 流量。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions 'IpProtocol=icmpv6,Ipv6Ranges=[{CidrIpv6=::/0}]'
```

输出：

```
{
```

```
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-04b612d9363ab6327",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmpv6",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv6": "::/0"
        }
    ]
}
```

示例 7：添加带有描述的规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加一个入站规则，该规则允许来自指定 IPv4 地址范围的 RDP 流量。该规则包含描述，可帮助以后识别。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --ip-permissions
 'IpProtocol=tcp,FromPort=3389,ToPort=3389,IpRanges=[{CidrIp=203.0.113.0/24,Description='RDP
 office'}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0397bbcc01e974db3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
            "ToPort": 3389,
            "CidrIpv4": "203.0.113.0/24",
            "Description": "RDP access from NY office"
        }
    ]
```

```
}
```

示例 8：添加使用前缀列表的入站规则

以下 authorize-security-group-ingress 示例使用 ip-permissions 参数添加一个入站规则，该规则允许指定前缀列表中 CIDR 范围的所有流量。

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-04a351bfe432d4e71 \
    --ip-permissions
 'IpProtocol=all,PrefixListIds=[{PrefixListId=pl-002dc3ec097de1514}]'
```

输出：

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-09c74b32f677c6c7c",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "PrefixListId": "pl-0721453c7ac4ec009"
        }
    ]
}
```

有关更多信息，请参阅 Amazon VPC 用户指南中的安全组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeSecurityGroupIngress。

bundle-instance

以下代码示例演示了如何使用 bundle-instance。

Amazon CLI

捆绑实例

此示例将实例 i-1234567890abcdef0 捆绑到名为 bundletasks 的存储桶中。在指定访问密钥 ID 的值之前，请查看并遵循“管理 Amazon 访问密钥的最佳实践”中的指导。

命令:

```
aws ec2 bundle-instance --instance-id i-1234567890abcdef0 --bucket bundletasks --
prefix winami --owner-akid AK12AJEXAMPLE --owner-sak example123example
```

输出：

```
{
  "BundleTask": {
    "UpdateTime": "2015-09-15T13:30:35.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "Storage": {
      "S3": {
        "Prefix": "winami",
        "Bucket": "bundletasks"
      }
    },
    "State": "pending",
    "StartTime": "2015-09-15T13:30:35.000Z",
    "BundleId": "bun-294e041f"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BundleInstance。

cancel-bundle-task

以下代码示例演示了如何使用 cancel-bundle-task。

Amazon CLI

取消捆绑任务

此示例将取消捆绑任务 bun-2a4e041c。

命令:

```
aws ec2 cancel-bundle-task --bundle-id bun-2a4e041c
```

输出：

```
{
  "BundleTask": {
    "UpdateTime": "2015-09-15T13:27:40.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "Storage": {
      "S3": {
        "Prefix": "winami",
        "Bucket": "bundletasks"
      }
    },
    "State": "cancelling",
    "StartTime": "2015-09-15T13:24:35.000Z",
    "BundleId": "bun-2a4e041c"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelBundleTask。

cancel-capacity-reservation-fleets

以下代码示例演示了如何使用 cancel-capacity-reservation-fleets。

Amazon CLI

取消容量预留实例集

以下 cancel-capacity-reservation-fleets 示例将取消指定的容量预留实例集及其预留的容量。当您取消实例集时，实例集的状态将更改为 cancelled，且其不能再创建新的容量预留。此外，实例集中的所有单个容量预留都将被取消，之前在预留容量中运行的实例将继续以共享容量正常运行。

```
aws ec2 cancel-capacity-reservation-fleets \
    --capacity-reservation-fleet-ids crf-abcdef01234567890
```

输出：

```
{
    "SuccessfulFleetCancellations": [
        {
```

```
            "CurrentFleetState": "cancelling",
            "PreviousFleetState": "active",
            "CapacityReservationFleetId": "crf-abcdef01234567890"
        }
    ],
    "FailedFleetCancellations": []
}
```

有关容量预留实例集的更多信息，请参阅《Amazon EC2 用户指南》中的容量预留实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelCapacityReservationFleets。

cancel-capacity-reservation

以下代码示例演示了如何使用 cancel-capacity-reservation。

Amazon CLI

取消容量预留

以下 cancel-capacity-reservation 示例将取消指定的容量预留。

```
aws ec2 cancel-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的取消容量预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelCapacityReservation。

cancel-conversion-task

以下代码示例演示了如何使用 cancel-conversion-task。

Amazon CLI

取消实例或卷的活动转换

此示例将取消与作业 ID import-i-fh95npoc 相关联的上传。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 cancel-conversion-task --conversion-task-id import-i-fh95npoc
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelConversionTask。

cancel-export-task

以下代码示例演示了如何使用 cancel-export-task。

Amazon CLI

取消活动导出任务

此示例将取消作业 ID 为 export-i-fgelt0i7 的活动导出任务。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 cancel-export-task --export-task-id export-i-fgelt0i7
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelExportTask。

cancel-image-launch-permission

以下代码示例演示了如何使用 cancel-image-launch-permission。

Amazon CLI

取消与您的 Amazon Web Services 账户共享 AMI

以下 cancel-image-launch-permission 示例从指定 AMI 的启动权限中删除您的账户。

```
aws ec2 cancel-image-launch-permission \
    --image-id ami-0123456789example \
    --region us-east-1
```

输出：

```
{
```

```
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的取消与您的 Amazon Web Services 账户共享 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelImageLaunchPermission。

cancel-import-task

以下代码示例演示了如何使用 cancel-import-task。

Amazon CLI

取消导入任务

以下 cancel-import-task 示例将取消指定导入映像任务。

```
aws ec2 cancel-import-task \
    --import-task-id import-ami-1234567890abcdef0
```

输出：

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
    "PreviousState": "active",
    "State": "deleting"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelImportTask。

cancel-reserved-instances-listing

以下代码示例演示了如何使用 cancel-reserved-instances-listing。

Amazon CLI

取消预留实例列示

以下 cancel-reserved-instances-listing 示例将取消指定的预留实例列示。

```
aws ec2 cancel-reserved-instances-listing \
```

```
    --reserved-instances-listing-id 5ec28771-05ff-4b9b-aa31-9e57dexample
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelReservedInstancesListing。

cancel-spot-fleet-requests

以下代码示例演示了如何使用 cancel-spot-fleet-requests。

Amazon CLI

示例 1：取消竞价型实例集请求并终止关联的实例

以下 cancel-spot-fleet-requests 示例将取消竞价型实例集请求并终止关联的按需型实例和竞价型实例。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

输出：

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_terminating",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

示例 2：取消竞价型实例集请求而不终止关联的实例

以下 cancel-spot-fleet-requests 示例将取消竞价型实例集请求而不终止关联的按需型实例和竞价型实例。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

输出：

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_running",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的取消竞价型实例集请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelSpotFleetRequests。

cancel-spot-instance-requests

以下代码示例演示了如何使用 cancel-spot-instance-requests。

Amazon CLI

取消竞价型实例请求

此示例命令将取消竞价型实例请求。

命令：

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-08b93456
```

输出：

```
{
    "CancelledSpotInstanceRequests": [
        {
            "State": "cancelled",
            "SpotInstanceRequestId": "sir-08b93456"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelSpotInstanceRequests。

confirm-product-instance

以下代码示例演示了如何使用 confirm-product-instance。

Amazon CLI

确认产品实例

此示例确定指定的产品代码是否与指定的实例相关联。

命令:

```
aws ec2 confirm-product-instance --product-code 774F4FF8 --instance-
id i-1234567890abcdef0
```

输出：

```
{
  "OwnerId": "123456789012"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmProductInstance。

copy-fpga-image

以下代码示例演示了如何使用 copy-fpga-image。

Amazon CLI

复制 Amazon FPGA 映像

此示例将指定的 AFI 从 us-east-1 区域复制到当前区域 (eu-west-1)。

命令:

```
aws ec2 copy-fpga-image --name copy-afi --source-fpga-image-id afi-0d123e123bfc85abc
 --source-region us-east-1 --region eu-west-1
```

输出：

```
{
  "FpgaImageId": "afi-06b12350a123fbabc"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyFpgaImage。

copy-image

以下代码示例演示了如何使用 copy-image。

Amazon CLI

示例 1：将 AMI 复制到其他区域

以下 copy-image 示例命令将指定的 AMI 从 us-west-2 区域复制到 us-east-1 区域，并添加简短描述。

```
aws ec2 copy-image \
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image."
```

输出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的复制 AMI。

示例 2：将 AMI 复制到其他区域并加密备份快照

以下 copy-image 命令将指定的 AMI 从 us-west-2 区域复制到当前区域，并使用指定的 KMS 密钥加密备份快照。

```
aws ec2 copy-image \
    --source-region us-west-2 \
    --name ami-name \
    --source-image-id ami-066877671789bd71b \
```

```
    --encrypted \
    --kms-key-id alias/my-kms-key
```

输出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的复制 AMI。

示例 3：在复制 AMI 时包含用户定义的 AMI 标签

在复制 AMI 时，以下 copy-image 命令使用 --copy-image-tags 参数复制用户定义的 AMI 标签。

```
aws ec2 copy-image \
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image."
    --copy-image-tags
```

输出：

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的复制 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyImage。

copy-snapshot

以下代码示例演示了如何使用 copy-snapshot。

Amazon CLI

示例 1：将快照复制到其他区域

以下 copy-snapshot 示例命令将指定的快照从 us-west-2 区域复制到 us-east-1 区域，并添加简短描述。

```
aws ec2 copy-snapshot \
    --region us-east-1 \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --description 'This is my copied snapshot.'
```

输出：

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

示例 2：复制未加密的快照并加密新快照

以下 copy-snapshot 命令将指定的未加密快照从 us-west-2 区域复制到当前区域，并使用指定的 KMS 密钥加密新快照。

```
aws ec2 copy-snapshot \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

输出：

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

有关更多信息，请参阅《Amazon EBS User Guide》中的 Copy an Amazon EBS snapshot。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopySnapshot。

create-capacity-reservation-fleet

以下代码示例演示了如何使用 create-capacity-reservation-fleet。

Amazon CLI

创建容量预留实例集

以下 create-capacity-reservation-fleet 示例为请求中指定的实例类型创建一个容量预留实例集，直至达到指定的总目标容量。容量预留机群为其预留容量的实例数取决于总目标容量和您在请求中指定的实例类型权重。指定要使用的实例类型以及每种指定实例类型的优先级。

```
aws ec2 create-capacity-reservation-fleet \
--total-target-capacity 24 \
--allocation-strategy prioritized \
--instance-match-criteria open \
--tenancy default \
--end-date 2022-12-31T23:59:59.000Z \
--instance-type-specifications file://instanceTypeSpecification.json
```

的内容instanceTypeSpecification.json：

```
[
    {
        "InstanceType": "m5.xlarge",
        "InstancePlatform": "Linux/UNIX",
        "Weight": 3.0,
        "AvailabilityZone":"us-east-1a",
        "EbsOptimized": true,
        "Priority" : 1
    }
]
```

输出：

```
{
    "Status": "submitted",
    "TotalFulfilledCapacity": 0.0,
    "CapacityReservationFleetId": "crf-abcdef01234567890",
    "TotalTargetCapacity": 24
}
```

有关容量预留实例集的更多信息，请参阅《Amazon EC2 用户指南》中的容量预留实例集。

有关实例类型权重和总目标容量的更多信息，请参阅《Amazon EC2 用户指南》中的实例类型权重和总目标容量。

有关为指定的实例类型指定优先级的更多信息，请参阅《Amazon EC2 用户指南》中的分配策略和实例类型优先级。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCapacityReservationFleet。

create-capacity-reservation

以下代码示例演示了如何使用 create-capacity-reservation。

Amazon CLI

示例 1：创建容量预留

以下 create-capacity-reservation 示例在 eu-west-1a 可用区中创建一个容量预留，您可以在其中启动三个运行 Linux/Unix 操作系统的 t2.medium 实例。默认情况下，容量预留是在开放实例匹配条件下创建的，不支持临时存储，在您手动取消容量预留之前，它将保持活动状态。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type t2.medium \
    --instance-platform Linux/UNIX \
    --instance-count 3
```

输出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T09:27:35.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "t2.medium"
    }
}
```

示例 2：创建在指定日期/时间自动结束的容量预留

以下 create-capacity-reservation 示例在 eu-west-1a 可用区中创建一个容量预留，您可以在其中启动三个运行 Linux/Unix 操作系统的 m5.large 实例。此容量预留于 2019 年 8 月 31 日 23:59:59 自动结束。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

输出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "limited",
        "AvailabilityZone": "eu-west-1a",
        "EndDate": "2019-08-31T23:59:59.000Z",
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:15:53.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
}
```

示例 3：创建只接受目标实例启动的容量预留

以下 create-capacity-reservation 示例将创建一个仅接受目标实例启动的容量预留。

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
```

```
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

输出：

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "targeted",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:21:57.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的创建容量预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCapacityReservation。

create-carrier-gateway

以下代码示例演示了如何使用 create-carrier-gateway。

Amazon CLI

创建运营商网关

以下 create-carrier-gateway 示例为指定的 VPC 创建运营商网关。

```
aws ec2 create-carrier-gateway \
    --vpc-id vpc-0c529aEXAMPLE1111
```

输出：

```
{
    "CarrierGateway": {
        "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",
        "VpcId": "vpc-0c529aEXAMPLE1111",
        "State": "pending",
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon Wavelength 用户指南》中的运营商网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCarrierGateway。

`create-client-vpn-endpoint`

以下代码示例演示了如何使用 `create-client-vpn-endpoint`。

Amazon CLI

创建 Client VPN 端点

以下 `create-client-vpn-endpoint` 示例创建一个使用双向认证的 Client VPN 端点，并为客户端 CIDR 块指定一个值。

```
aws ec2 create-client-vpn-endpoint \
    --client-cidr-block "172.31.0.0/16" \
    --server-certificate-arn arn:aws:acm:ap-south-1:123456789012:certificate/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --authentication-options Type=certificate-
authentication,MutualAuthentication={ClientRootCertificateChainArn=arn:aws:acm:ap-
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE} \
    --connection-log-options Enabled=false
```

输出：

```
{
    "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
    "Status": {
        "Code": "pending-associate"
    },
    "DnsName": "cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-
south-1.amazonaws.com"
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的 Client VPN 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClientVpnEndpoint。

`create-client-vpn-route`

以下代码示例演示了如何使用 `create-client-vpn-route`。

Amazon CLI

为 Client VPN 端点创建路由

以下 `create-client-vpn-route` 示例为 Client VPN 端点的指定子网添加指向互联网（`0.0.0.0/0`）的路由。

```
aws ec2 create-client-vpn-route \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --destination-cidr-block 0.0.0.0/0  \
    --target-vpc-subnet-id subnet-0123456789abcabca
```

输出：

```
{
    "Status": {
        "Code": "creating"
    }
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClientVpnRoute。

`create-coip-cidr`

以下代码示例演示了如何使用 `create-coip-cidr`。

Amazon CLI

创建客户拥有的 IP（CoIP）地址范围

以下 create-coip-cidr 示例在指定的 CoIP 池中创建指定范围的 CoIP 地址。

```
aws ec2 create-coip-cidr \
    --cidr 15.0.0.0/24 \
    --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

输出：

```
{
    "CoipCidr": {
        "Cidr": "15.0.0.0/24",
        "CoipPoolId": "ipv4pool-coip-1234567890abcdefg",
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCoipCidr。

create-coip-pool

以下代码示例演示了如何使用 create-coip-pool。

Amazon CLI

创建客户拥有的 IP（CoIP）地址池

以下 create-coip-pool 示例为指定的本地网关路由表中的 CoIP 地址创建 CoIP 池。

```
aws ec2 create-coip-pool \
    --local-gateway-route-table-id lgw-rtb-abcdefg1234567890
```

输出：

```
{
    "CoipPool": {
        "PoolId": "ipv4pool-coip-1234567890abcdefg",
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
        "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-
coip-1234567890abcdefg"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCoipPool。

create-customer-gateway

以下代码示例演示了如何使用 create-customer-gateway。

Amazon CLI

创建客户网关

此示例为其外部接口创建一个具有指定 IP 地址的客户网关。

命令:

```
aws ec2 create-customer-gateway --type ipsec.1 --public-ip 12.1.2.3 --bgp-asn 65534
```

输出：

```
{
    "CustomerGateway": {
        "CustomerGatewayId": "cgw-0e11f167",
        "IpAddress": "12.1.2.3",
        "State": "available",
        "Type": "ipsec.1",
        "BgpAsn": "65534"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomerGateway。

create-default-subnet

以下代码示例演示了如何使用 create-default-subnet。

Amazon CLI

创建默认子网

此示例在可用区 us-east-2a 中创建默认子网。

命令:

```
 aws ec2 create-default-subnet --availability-zone us-east-2a

{
   "Subnet": {
       "AvailabilityZone": "us-east-2a",
       "Tags": [],
       "AvailableIpAddressCount": 4091,
       "DefaultForAz": true,
       "Ipv6CidrBlockAssociationSet": [],
       "VpcId": "vpc-1a2b3c4d",
       "State": "available",
       "MapPublicIpOnLaunch": true,
       "SubnetId": "subnet-1122aabb",
       "CidrBlock": "172.31.32.0/20",
       "AssignIpv6AddressOnCreation": false
   }
 }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDefaultSubnet。

create-default-vpc

以下代码示例演示了如何使用 create-default-vpc。

Amazon CLI

创建默认 VPC

此示例创建默认 VPC。

命令:

```
aws ec2 create-default-vpc
```

输出：

```
{
   "Vpc": {
```

```
        "VpcId": "vpc-8eaae5ea",
        "InstanceTenancy": "default",
        "Tags": [],
        "Ipv6CidrBlockAssociationSet": [],
        "State": "pending",
        "DhcpOptionsId": "dopt-af0c32c6",
        "CidrBlock": "172.31.0.0/16",
        "IsDefault": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDefaultVpc。

create-dhcp-options

以下代码示例演示了如何使用 create-dhcp-options。

Amazon CLI

创建 DHCP 选项集

以下 create-dhcp-options 示例创建 DHCP 选项集，该选项集用于指定域名、域名服务器和 NetBIOS 节点类型。

```
aws ec2 create-dhcp-options \
    --dhcp-configuration \
        "Key=domain-name-servers,Values=10.2.5.1,10.2.5.2" \
        "Key=domain-name,Values=example.com" \
        "Key=netbios-node-type,Values=2"
```

输出：

```
{
    "DhcpOptions": {
        "DhcpConfigurations": [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
```

```
            },
            {
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "10.2.5.1"
                    },
                    {
                        "Value": "10.2.5.2"
                    }
                ]
            },
            {
                "Key": "netbios-node-type",
                "Values": [
                    {
                        "Value": "2"
                    }
                ]
            }
        ],
        "DhcpOptionsId": "dopt-06d52773eff4c55f3"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDhcpOptions。

create-egress-only-internet-gateway

以下代码示例演示了如何使用 create-egress-only-internet-gateway。

Amazon CLI

创建仅出口互联网网关

此示例为指定的 VPC 创建仅出口互联网网关。

命令:

```
aws ec2 create-egress-only-internet-gateway --vpc-id vpc-0c62a468
```

输出：

```
{
  "EgressOnlyInternetGateway": {
      "EgressOnlyInternetGatewayId": "eigw-015e0e244e24dfe8a",
      "Attachments": [
          {
              "State": "attached",
              "VpcId": "vpc-0c62a468"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEgressOnlyInternetGateway。

create-fleet

以下代码示例演示了如何使用 create-fleet。

Amazon CLI

示例 1：创建启动竞价型实例作为默认购买模式的 EC2 实例集

以下 create-fleet 示例使用启动实例集所需的最少参数创建 EC2 实例集：启动模板、目标容量和默认购买模式。启动模板由其启动模板 ID 和版本号标识。实例集的目标容量为 2 个实例，默认购买模式为 spot，因此实例集启动两个竞价型实例。

在创建 EC2 实例集时，可以使用 JSON 文件指定要启动的实例的有关信息。

```
aws ec2 create-fleet \
    --cli-input-json file://file_name.json
```

file_name.json 的内容：

```
{
    "LaunchTemplateConfigs": [
    {
        "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
        "Version": "1"
        }
```

```
    }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "spot"
    }
}
```

输出：

```
{
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

示例 2：创建将按需型实例作为默认购买模式启动的 EC2 实例集

以下 `create-fleet` 示例使用启动实例集所需的最少参数创建 EC2 实例集：启动模板、目标容量和默认购买模式。启动模板由其启动模板 ID 和版本号标识。实例集的目标容量为 2 个实例，默认购买模式为 `on-demand`，因此实例集启动两个按需型实例。

在创建 EC2 实例集时，可以使用 JSON 文件指定要启动的实例的有关信息。

```
aws ec2 create-fleet \
    --cli-input-json file://file_name.json
```

file_name.json 的内容：

```
{
    "LaunchTemplateConfigs": [
    {
        "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
        "Version": "1"
        }
    }
    ],
    "TargetCapacitySpecification": {
    "TotalTargetCapacity": 2,
    "DefaultTargetCapacityType": "on-demand"
    }
}
```

输出：

```
{
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

示例 3：创建将按需型实例作为主容量启动的 EC2 实例集

以下 `create-fleet` 示例创建一个 EC2 实例集，该实例集指定实例集的 2 个实例的总目标容量，以及 1 个按需型实例的目标容量。默认的购买模式为 `spot`。实例集按照指定的方式启动 1 个按需型实例，但需要再启动一个实例以满足总目标容量要求。差值的购买模式是通过 `TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType` 计算得出的，因而实例集会启动 1 个竞价型实例。

在创建 EC2 实例集时，可以使用 JSON 文件指定要启动的实例的有关信息。

```
aws ec2 create-fleet \
    --cli-input-json file://file_name.json
```

file_name.json 的内容：

```
{
    "LaunchTemplateConfigs": [
    {
        "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
        "Version": "1"
        }
    }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "OnDemandTargetCapacity":1,
        "DefaultTargetCapacityType": "spot"
    }
}
```

输出：

```
{
```

```
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

示例 4：创建使用价格最低的分配策略启动竞价型实例的 EC2 实例集

如果未指定竞价型实例的分配策略，则使用默认分配策略 lowest-price。以下 create-fleet 示例使用 lowest-price 分配策略创建 EC2 实例集。覆盖启动模板的三个启动规范有不同的实例类型，但有相同的权重容量和子网。总目标容量为两个实例，默认购买模式为 spot。EC2 实例集按启动规范的最低价格实例类型启动两个竞价型实例。

在创建 EC2 实例集时，可以使用 JSON 文件指定要启动的实例的有关信息。

```
aws ec2 create-fleet \
    --cli-input-json file://file_name.jsonContents of file_name.json::

{
    "LaunchTemplateConfigs": [
    {
        "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
        "Version": "1"
        },
        "Overrides": [
            {
                "InstanceType": "c4.large",
                "WeightedCapacity": 1,
                "SubnetId": "subnet-a4f6c5d3"
            },
            {
                "InstanceType": "c3.large",
                "WeightedCapacity": 1,
                "SubnetId": "subnet-a4f6c5d3"
            },
            {
                "InstanceType": "c5.large",
                "WeightedCapacity": 1,
                "SubnetId": "subnet-a4f6c5d3"
            }
        ]
    }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
```

```
        "DefaultTargetCapacityType": "spot"

    }
}
```

输出：

```
{
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFleet。

create-flow-logs

以下代码示例演示了如何使用 create-flow-logs。

Amazon CLI

示例 1：创建流日志

以下 create-flow-logs 示例将创建流日志，该日志将捕获指定网络接口的所有已拒绝流量。使用指定 IAM 角色中的权限将流日志传送到 CloudWatch Logs 中的日志组。

```
aws ec2 create-flow-logs \
    --resource-type NetworkInterface \
    --resource-ids eni-11223344556677889 \
    --traffic-type REJECT \
    --log-group-name my-flow-logs \
    --deliver-logs-permission-arn arn:aws:iam::123456789101:role/publishFlowLogs
```

输出：

```
{
    "ClientToken": "so0eNA2uSHUNlHI0S2cJ305GuIX1CezaRdGtexample",
    "FlowLogIds": [
        "fl-12345678901234567"
    ],
    "Unsuccessful": []
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 流日志。

示例 2：使用自定义格式创建流日志

以下 create-flow-logs 示例将创建流日志，以捕获指定 VPC 的所有流量并将流日志传输到 Amazon S3 存储桶。--log-format 参数指定流日志记录的自定义格式。要在 Windows 上运行此命令，请将单引号 (') 更改为双引号 (")。

```
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --log-format '${version} ${vpc-id} ${subnet-id} ${instance-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${tcp-flags} ${type} ${pkt-srcaddr}
 ${pkt-dstaddr}'
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 流日志。

示例 3：创建最大聚合间隔为一分钟的流日志

以下 create-flow-logs 示例将创建流日志，以捕获指定 VPC 的所有流量并将流日志传输到 Amazon S3 存储桶。--max-aggregation-interval 参数指定的最大聚合间隔为 60 秒（1 分钟）。

```
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --max-aggregation-interval 60
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 流日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFlowLogs。

create-fpga-image

以下代码示例演示了如何使用 create-fpga-image。

Amazon CLI

创建 Amazon FPGA 映像

此示例从指定存储桶中的指定压缩包创建 AFI。

命令:

```
aws ec2 create-fpga-image --name my-afi --description test-afi --input-storage-location Bucket=my-fpga-bucket,Key=dcp/17_12_22-103226.Developer_CL.tar --logs-storage-location Bucket=my-fpga-bucket,Key=logs
```

输出：

```
{
  "FpgaImageId": "afi-0d123e123bfc85abc",
  "FpgaImageGlobalId": "agfi-123cb27b5e84a0abc"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFpgaImage。

create-image

以下代码示例演示了如何使用 create-image。

Amazon CLI

示例 1：从 Amazon EBS 支持的实例创建 AMI

以下 create-image 示例从指定的实例创建 AMI。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --description "An AMI for my server"
```

输出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

有关为 AMI 指定块设备映射的更多信息，请参阅《Amazon EC2 用户指南》中的为 AMI 指定块设备映射。

示例 2：在不重启的情况下从 Amazon EBS 支持的实例创建 AMI

以下 create-image 示例创建 AMI 并设置 --no-reboot 参数，这样在创建映像之前就不会重启实例。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --no-reboot
```

输出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

有关为 AMI 指定块设备映射的更多信息，请参阅《Amazon EC2 用户指南》中的为 AMI 指定块设备映射。

示例 3：在创建时标记 AMI 和快照

以下 create-image 示例创建 AMI，并使用相同的标签 cost-center=cc123 标记 AMI 和快照。

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --tag-specifications "ResourceType=image,Tags=[{Key=cost-
center,Value=cc123}]" "ResourceType=snapshot,Tags=[{Key=cost-center,Value=cc123}]"
```

输出：

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

有关在创建时标记资源的更多信息，请参阅《Amazon EC2 用户指南》中的在创建资源时添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateImage。

create-instance-connect-endpoint

以下代码示例演示了如何使用 create-instance-connect-endpoint。

Amazon CLI

创建 EC2 实例连接端点

以下 create-instance-connect-endpoint 示例在指定子网中创建 EC2 实例连接端点。

```
aws ec2 create-instance-connect-endpoint \
    --region us-east-1 \
    --subnet-id subnet-0123456789example
```

输出：

```
{
    "VpcId": "vpc-0123abcd",
    "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
    "AvailabilityZone": "us-east-1a",
    "NetworkInterfaceIds": [
        "eni-0123abcd"
    ],
    "PreserveClientIp": true,
    "Tags": [],
    "FipsDnsName": "eice-0123456789example.0123abcd.fips.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
    "StateMessage": "",
    "State": "create-complete",
    "DnsName": "eice-0123456789example.0123abcd.ec2-instance-connect-endpoint.us-
east-1.amazonaws.com",
    "SubnetId": "subnet-0123abcd",
    "OwnerId": "111111111111",
    "SecurityGroupIds": [
        "sg-0123abcd"
    ],
    "InstanceConnectEndpointId": "eice-0123456789example",
    "CreatedAt": "2023-04-07T15:43:53.000Z"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的创建 EC2 实例连接端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstanceConnectEndpoint。

create-instance-event-window

以下代码示例演示了如何使用 create-instance-event-window。

Amazon CLI

示例 1：创建有时间范围的事件窗口

以下 create-instance-event-window 示例将创建有时间范围的事件窗口。您不能同时指定 cron-expression 参数。

```
aws ec2 create-instance-event-window \
    --region us-east-1 \
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8 \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
```

```
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 2：创建有 cron 表达式的事件窗口

以下 `create-instance-event-window` 示例将创建有 cron 表达式的事件窗口。您不能同时指定 `time-range` 参数。

```
aws ec2 create-instance-event-window \
    --region us-east-1 \
    --cron-expression "* 21-23 * * 2,3" \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstanceEventWindow。

create-instance-export-task

以下代码示例演示了如何使用 `create-instance-export-task`。

Amazon CLI

导出实例

此示例命令将创建一个任务，用于将实例 i-1234567890abcdef0 导出到 Amazon S3 存储桶 myexportbucket。

命令:

```
aws ec2 create-instance-export-task --description "RHEL5 instance" --
instance-id i-1234567890abcdef0 --target-environment vmware --export-to-s3-
task DiskImageFormat=vmdk,ContainerFormat=ova,S3Bucket=myexportbucket,S3Prefix=RHEL5
```

输出：

```
{
    "ExportTask": {
        "State": "active",
        "InstanceExportDetails": {
            "InstanceId": "i-1234567890abcdef0",
            "TargetEnvironment": "vmware"
        },
        "ExportToS3Task": {
            "S3Bucket": "myexportbucket",
            "S3Key": "RHEL5export-i-fh8sjjsq.ova",
            "DiskImageFormat": "vmdk",
            "ContainerFormat": "ova"
        },
        "Description": "RHEL5 instance",
        "ExportTaskId": "export-i-fh8sjjsq"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstanceExportTask。

create-internet-gateway

以下代码示例演示了如何使用 create-internet-gateway。

Amazon CLI

创建互联网网关

以下 create-internet-gateway 示例将创建带有标签 Name=my-igw 的互联网网关。

```
aws ec2 create-internet-gateway \
    --tag-specifications ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-igw}]
```

输出：

```
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-0d0fb496b3994d755",
        "OwnerId": "123456789012",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-igw"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的互联网网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInternetGateway。

create-ipam-pool

以下代码示例演示了如何使用 create-ipam-pool。

Amazon CLI

创建 IPAM 池

以下 create-ipam-pool 示例将创建 IPAM 池。

（Linux）：

```
aws ec2 create-ipam-pool \
    --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \
    --address-family ipv4 \
```

```
    --auto-import \
    --allocation-min-netmask-length 16 \
    --allocation-max-netmask-length 26 \
    --allocation-default-netmask-length 24 \
    --allocation-resource-tags "Key=Environment,Value=Preprod" \
    --tag-specifications 'ResourceType=ipam-pool,Tags=[{Key=Name,Value="Preprod
 pool"}]'
```

（Windows）：

```
aws ec2 create-ipam-pool ^
    --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^
    --address-family ipv4 ^
    --auto-import ^
    --allocation-min-netmask-length 16 ^
    --allocation-max-netmask-length 26 ^
    --allocation-default-netmask-length 24 ^
    --allocation-resource-tags "Key=Environment,Value=Preprod" ^
    --tag-specifications ResourceType=ipam-pool,Tags=[{Key=Name,Value="Preprod
 pool"}]
```

输出：

```
{
    "IpamPool": {
        "OwnerId": "123456789012",
        "IpamPoolId": "ipam-pool-0533048da7d823723",
        "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-0533048da7d823723",
        "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
        "IpamScopeType": "private",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
        "IpamRegion": "us-east-1",
        "Locale": "None",
        "PoolDepth": 1,
        "State": "create-in-progress",
        "AutoImport": true,
        "AddressFamily": "ipv4",
        "AllocationMinNetmaskLength": 16,
        "AllocationMaxNetmaskLength": 26,
        "AllocationDefaultNetmaskLength": 24,
        "AllocationResourceTags": [
```

```
            {
                "Key": "Environment",
                "Value": "Preprod"
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "Preprod pool"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的计划 IP 地址预置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpamPool。

create-ipam-resource-discovery

以下代码示例演示了如何使用 create-ipam-resource-discovery。

Amazon CLI

创建资源发现

在此示例中，您是一名委派 IPAM 管理员，您希望创建资源发现并与另一个 Amazon 组织中的 IPAM 管理员共享资源发现，以便其他组织中的管理员可以管理和监控您的组织中资源的 IP 地址。

重要提示

此示例同时包括 --region 和 --operating-regions 选项，因为虽然它们是可选的，但必须以特定方式配置它们才能成功将资源发现与 IPAM 集成。* --operating-regions 必须与您希望 IPAM 发现的资源所在的区域匹配。如果存在您不希望 IPAM 管理 IP 地址的区域（例如出于合规性原因），请不要将其包括在内。* --region 必须与要与之关联的 IPAM 的主区域相匹配。您必须在创建 IPAM 的同一区域中创建资源发现。例如，如果要关联的 IPAM 是在 us-east-1 中创建的，请在请求中包含 --region us-east-1。如果您未指定 --region 和 --operating-regions 选项，则它们都默认为您运行命令的区域。

在此示例中，我们与之集成的 IPAM 的运营区域包括 us-west-1、us-west-2 和 ap-south-1。创建资源发现时，我们希望 IPAM 发现 us-west-1 和 us-west-2（而不是

ap-south-1）中的资源 IP 地址。因此，我们在请求中仅包含 --operating-regions RegionName='us-west-1' RegionName='us-west-2'。

以下 create-ipam-resource-discovery 示例将创建 IPAM 资源发现。

```
aws ec2 create-ipam-resource-discovery \
    --description 'Example-resource-discovery' \
    --tag-specifications 'ResourceType=ipam-resource-discovery,Tags=[{Key=cost-center,Value=cc123}]' \
    --operating-regions RegionName='us-west-1' RegionName='us-west-2' \
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscovery":{
        "OwnerId": "149977607591",
        "IpamResourceDiscoveryId": "ipam-res-disco-0257046d8aa78b8bc",
        "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-discovery/ipam-res-disco-0257046d8aa78b8bc",
        "IpamResourceDiscoveryRegion": "us-east-1",
        "Description": "'Example-resource-discovery'",
        "OperatingRegions":[
            {"RegionName": "us-west-1"},
            {"RegionName": "us-west-2"},
            {"RegionName": "us-east-1"}
        ],
        "IsDefault": false,
        "State": "create-in-progress",
        "Tags": [
            {
                "Key": "cost-center",
                "Value": "cc123"
            }
        ]
}
```

创建资源发现后，您可能希望与其他 IPAM 委派管理员共享它，您可以使用 create-resource-share 执行此操作。有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpamResourceDiscovery。

create-ipam-scope

以下代码示例演示了如何使用 create-ipam-scope。

Amazon CLI

创建 IPAM 范围

以下 create-ipam-scope 示例将创建 IPAM 范围。

（Linux）：

```
aws ec2 create-ipam-scope \
    --ipam-id ipam-08440e7a3acde3908 \
    --description "Example description" \
    --tag-specifications 'ResourceType=ipam-scope,Tags=[{Key=Name,Value="Example
 name value"}]'
```

（Windows）：

```
aws ec2 create-ipam-scope ^
    --ipam-id ipam-08440e7a3acde3908 ^
    --description "Example description" ^
    --tag-specifications ResourceType=ipam-scope,Tags=[{Key=Name,Value="Example name
 value"}]
```

输出：

```
{
    "IpamScope": {
        "OwnerId": "123456789012",
        "IpamScopeId": "ipam-scope-01c1ebab2b63bd7e4",
        "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-01c1ebab2b63bd7e4",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
        "IpamRegion": "us-east-1",
        "IpamScopeType": "private",
        "IsDefault": false,
        "Description": "Example description",
        "PoolCount": 0,
        "State": "create-in-progress",
        "Tags": [
            {
```

```
                "Key": "Name",
                "Value": "Example name value"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的创建额外范围。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpamScope。

create-ipam

以下代码示例演示了如何使用 create-ipam。

Amazon CLI

创建 IPAM

以下 create-ipam 示例将创建 IPAM。

（Linux）：

```
aws ec2 create-ipam \
    --description "Example description" \
    --operating-regions "RegionName=us-east-2" "RegionName=us-west-1" \
    --tag-specifications 'ResourceType=ipam,Tags=[{Key=Name,Value=ExampleIPAM}]'
```

（Windows）：

```
aws ec2 create-ipam ^
    --description "Example description" ^
    --operating-regions "RegionName=us-east-2" "RegionName=us-west-1" ^
    --tag-specifications ResourceType=ipam,Tags=[{Key=Name,Value=ExampleIPAM}]
```

输出：

```
{
    "Ipam": {
        "OwnerId": "123456789012",
        "IpamId": "ipam-036486dfa6af58ee0",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-036486dfa6af58ee0",
        "IpamRegion": "us-east-1",
```

```
        "PublicDefaultScopeId": "ipam-scope-071b8042b0195c183",
        "PrivateDefaultScopeId": "ipam-scope-0807405dece705a30",
        "ScopeCount": 2,
        "OperatingRegions": [
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-west-1"
            },
            {
                "RegionName": "us-east-1"
            }
        ],
        "State": "create-in-progress",
        "Tags": [
            {
                "Key": "Name",
                "Value": "ExampleIPAM"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的创建 IPAM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpam。

create-key-pair

以下代码示例演示了如何使用 create-key-pair。

Amazon CLI

创建密钥对

本示例将创建一个名为 MyKeyPair 的密钥对。

命令:

```
aws ec2 create-key-pair --key-name MyKeyPair
```

输出是私有密钥和密钥指纹的 ASCII 版本。需要将密钥保存到文件中。

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的“使用密钥对”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKeyPair。

create-launch-template-version

以下代码示例演示了如何使用 create-launch-template-version。

Amazon CLI

创建启动模板版本

此示例根据启动模板的版本 1 创建新的启动模板版本并指定不同的 AMI ID。

命令:

```
aws ec2 create-launch-template-version --launch-template-id lt-0abcd290751193123
 --version-description WebVersion2 --source-version 1 --launch-template-data
 '{"ImageId":"ami-c998b6b2"}'
```

输出：

```
{
  "LaunchTemplateVersion": {
      "VersionDescription": "WebVersion2",
      "LaunchTemplateId": "lt-0abcd290751193123",
      "LaunchTemplateName": "WebServers",
      "VersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "LaunchTemplateData": {
          "ImageId": "ami-c998b6b2",
          "InstanceType": "t2.micro",
          "NetworkInterfaces": [
              {
                  "Ipv6Addresses": [
                      {
                          "Ipv6Address": "2001:db8:1234:1a00::123"
                      }
                  ],
                  "DeviceIndex": 0,
                  "SubnetId": "subnet-7b16de0c",
                  "AssociatePublicIpAddress": true
              }
```

```
            ]
        },
        "DefaultVersion": false,
        "CreateTime": "2017-12-01T13:35:46.000Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLaunchTemplateVersion。

create-launch-template

以下代码示例演示了如何使用 create-launch-template。

Amazon CLI

示例 1：创建启动模板

以下 create-launch-template 示例创建一个启动模板，该模板指定启动实例的子网，为实例分配公有 IP 地址和 IPv6 地址，并为实例创建标签。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
    --version-description WebVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"AssociatePublicIpAddress":true,"DeviceIndex":0,"Ipv6AddressCount":1,"SubnetId":"subnet-7b
[{"ResourceType":"instance","Tags":[{"Key":"purpose","Value":"webserver"}]}]}'
```

输出：

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-01238c059e3466abc",
        "LaunchTemplateName": "TemplateForWebServer",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-01-27T09:13:24.000Z"
    }
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“从启动模板启动实例”。有关引用 JSON 格式参数的信息，请参阅《Amazon 命令行界面用户指南》中的“引用字符串”。

示例 2：为 Amazon EC2 Auto Scaling 创建启动模板

以下 create-launch-template 示例创建一个具有多个标签和块设备映射的启动模板，以在实例启动时指定一个附加 EBS 卷。为 Groups 指定一个值，该值与 VPC 的安全组对应，自动扩缩组会将实例启动到该安全组中。指定 VPC 和子网作为自动扩缩组的属性。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForAutoScaling \
    --version-description AutoScalingVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-7c227019,sg-903004f8"],"DeleteOnTermination":true}],"ImageId":"ami-
b42209de","InstanceType":"m4.large","TagSpecifications":
[{"ResourceType":"instance","Tags":[{"Key":"environment","Value":"production"},
{"Key":"purpose","Value":"webserver"}]},{"ResourceType":"volume","Tags":
[{"Key":"environment","Value":"production"},{"Key":"cost-
center","Value":"cc123"}]}],"BlockDeviceMappings":[{"DeviceName":"/dev/sda1","Ebs":
{"VolumeSize":100}}]}' --region us-east-1
```

输出：

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0123c79c33a54e0abc",
        "LaunchTemplateName": "TemplateForAutoScaling",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-04-30T18:16:06.000Z"
    }
}
```

有关更多信息，请参阅《Amazon EC2 Auto Scaling 用户指南》中的“为自动扩缩组创建启动模板”。有关引用 JSON 格式参数的信息，请参阅《Amazon 命令行界面用户指南》中的“引用字符串”。

示例 3：创建指定对 EBS 卷进行加密的启动模板

以下 create-launch-template 示例创建一个启动模板，其中包括根据未加密的快照创建的加密 EBS 卷。同时还会在创建过程中标记卷。如果默认情况下禁用了加密，则必须指定以下示例中所示的 "Encrypted" 选项。如果使用 "KmsKeyId" 选项来指定客户托管的 CMK，则即使默认情况下启用了加密，也必须指定 "Encrypted" 选项。

```
aws ec2 create-launch-template \
  --launch-template-name TemplateForEncryption \
  --launch-template-data file://config.json
```

的内容config.json：

```
{
    "BlockDeviceMappings":[
        {
            "DeviceName":"/dev/sda1",
            "Ebs":{
                "VolumeType":"gp2",
                "DeleteOnTermination":true,
                "SnapshotId":"snap-066877671789bd71b",
                "Encrypted":true,
                "KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/abcd1234-
a123-456a-a12b-a123b4cd56ef"
            }
        }
    ],
    "ImageId":"ami-00068cd7555f543d5",
    "InstanceType":"c5.large",
    "TagSpecifications":[
        {
            "ResourceType":"volume",
            "Tags":[
                {
                    "Key":"encrypted",
                    "Value":"yes"
                }
            ]
        }
    ]
}
```

输出：

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0d5bd51bcf8530abc",
        "LaunchTemplateName": "TemplateForEncryption",
```

```
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2020-01-07T19:08:36.000Z"
    }
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“从快照恢复 Amazon EBS 卷且默认情况下加密”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLaunchTemplate。

create-local-gateway-route-table-virtual-interface-group-association

以下代码示例演示了如何使用 create-local-gateway-route-table-virtual-interface-group-association。

Amazon CLI

将本地网关路由表与虚拟接口（VIF）组相关联

以下 create-local-gateway-route-table-virtual-interface-group-association 示例将在指定的本地网关路由表与 VIF 组之间创建关联。

```
aws ec2 create-local-gateway-route-table-virtual-interface-group-association \
    --local-gateway-route-table-id lgw-rtb-exampleidabcd1234 \
    --local-gateway-virtual-interface-group-id lgw-vif-grp-exampleid0123abcd
```

输出：

```
{
    "LocalGatewayRouteTableVirtualInterfaceGroupAssociation": {
        "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-grp-
assoc-exampleid12345678",
        "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-exampleid0123abcd",
        "LocalGatewayId": "lgw-exampleid11223344",
        "LocalGatewayRouteTableId": "lgw-rtb-exampleidabcd1234",
        "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-
gateway-route-table/lgw-rtb-exampleidabcd1234",
        "OwnerId": "111122223333",
        "State": "pending",
        "Tags": []
    }
```

```
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的 VIF 组关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation。

create-local-gateway-route-table-vpc-association

以下代码示例演示了如何使用 create-local-gateway-route-table-vpc-association。

Amazon CLI

将 VPC 与路由表相关联

以下 create-local-gateway-route-table-vpc-association 示例将指定的 VPC 与指定的本地网关路由表相关联。

```
aws ec2 create-local-gateway-route-table-vpc-association \
    --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE \
    --vpc-id vpc-07ef66ac71EXAMPLE
```

输出：

```
{
    "LocalGatewayRouteTableVpcAssociation": {
        "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-0ee765bcc8EXAMPLE",
        "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
        "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
        "VpcId": "vpc-07ef66ac71EXAMPLE",
        "State": "associated"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLocalGatewayRouteTableVpcAssociation。

create-local-gateway-route-table

以下代码示例演示了如何使用 create-local-gateway-route-table。

Amazon CLI

创建本地网关路由表

以下 create-local-gateway-route-table 示例使用直接 VPC 路由模式创建本地网关路由表。

```
aws ec2 create-local-gateway-route-table \
    --local-gateway-id lgw-1a2b3c4d5e6f7g8h9 \
    --mode direct-vpc-routing
```

输出：

```
{
    "LocalGatewayRouteTable": {
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
        "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-
gateway-route-table/lgw-rtb-abcdefg1234567890",
        "LocalGatewayId": "lgw-1a2b3c4d5e6f7g8h9",
        "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/
op-021345abcdef67890",
        "OwnerId": "111122223333",
        "State": "pending",
        "Tags": [],
        "Mode": "direct-vpc-routing"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的本地网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLocalGatewayRouteTable。

create-local-gateway-route

以下代码示例演示了如何使用 create-local-gateway-route。

Amazon CLI

为本地网关路由表创建静态路由

以下 create-local-gateway-route 示例将在指定的本地网关路由表中创建指定的路由。

```
aws ec2 create-local-gateway-route \
    --destination-cidr-block 0.0.0.0/0 \
    --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "0.0.0.0/0",
        "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
        "Type": "static",
        "State": "deleted",
        "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLocalGatewayRoute。

create-managed-prefix-list

以下代码示例演示了如何使用 create-managed-prefix-list。

Amazon CLI

创建前缀列表

以下 create-managed-prefix-list 示例将创建最多可包含 10 个条目的 IPv4 前缀列表，并在前缀列表中创建 2 个条目。

```
aws ec2 create-managed-prefix-list \
    --address-family IPv4 \
    --max-entries 10 \
    --entries Cidr=10.0.0.0/16,Description=vpc-a Cidr=10.2.0.0/16,Description=vpc-b
 \
    --prefix-list-name vpc-cidrs
```

输出：

```
{
    "PrefixList": {
```

```
        "PrefixListId": "pl-0123456abcabcabc1",
        "AddressFamily": "IPv4",
        "State": "create-in-progress",
        "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
        "PrefixListName": "vpc-cidrs",
        "MaxEntries": 10,
        "Version": 1,
        "Tags": [],
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateManagedPrefixList。

`create-nat-gateway`

以下代码示例演示了如何使用 `create-nat-gateway`。

Amazon CLI

示例 1：创建公共 NAT 网关

以下 `create-nat-gateway` 示例在指定子网中创建一个公共 NAT 网关，并将弹性 IP 地址与指定的分配 ID 相关联。在创建公共 NAT 网关时，您必须关联弹性 IP 地址。

```
aws ec2 create-nat-gateway \
    --subnet-id subnet-0250c25a1fEXAMPLE \
    --allocation-id eipalloc-09ad461b0dEXAMPLE
```

输出：

```
{
    "NatGateway": {
        "CreateTime": "2021-12-01T22:22:38.000Z",
        "NatGatewayAddresses": [
            {
                "AllocationId": "eipalloc-09ad461b0dEXAMPLE"
            }
        ],
```

```
        "NatGatewayId": "nat-0c61bf8a12EXAMPLE",
        "State": "pending",
        "SubnetId": "subnet-0250c25a1fEXAMPLE",
        "VpcId": "vpc-0a60eb65b4EXAMPLE",
        "ConnectivityType": "public"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

示例 2：创建私有 NAT 网关

以下 `create-nat-gateway` 示例将在指定子网中创建私有 NAT 网关。私有 NAT 网关没有关联的弹性 IP 地址。

```
aws ec2 create-nat-gateway \
    --subnet-id subnet-0250c25a1fEXAMPLE \
    --connectivity-type private
```

输出：

```
{
    "NatGateway": {
        "CreateTime": "2021-12-01T22:26:00.000Z",
        "NatGatewayAddresses": [
            {}
        ],
        "NatGatewayId": "nat-011b568379EXAMPLE",
        "State": "pending",
        "SubnetId": "subnet-0250c25a1fEXAMPLE",
        "VpcId": "vpc-0a60eb65b4EXAMPLE",
        "ConnectivityType": "private"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNatGateway。

create-network-acl-entry

以下代码示例演示了如何使用 `create-network-acl-entry`。

Amazon CLI

创建网络 ACL 条目

此示例将为指定的网络 ACL 创建一个条目。该规则允许来自 UDP 端口 53（DNS）上任何 IPv4 地址（0.0.0.0/0）的入口流量进入任何关联子网。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100 --protocol udp --port-range From=53,To=53 --cidr-block 0.0.0.0/0 --rule-action allow
```

此示例将为指定的网络 ACL 创建一个规则，该规则允许来自 TCP 端口 80（HTTP）上任何 IPv6 地址（::/0）的入口流量。

命令:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 120 --protocol tcp --port-range From=80,To=80 --ipv6-cidr-block ::/0 --rule-action allow
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkAclEntry。

create-network-acl

以下代码示例演示了如何使用 create-network-acl。

Amazon CLI

创建网络 ACL

此示例将为指定的 VPC 创建网络 ACL。

命令:

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

输出：

```
{
    "NetworkAcl": {
        "Associations": [],
        "NetworkAclId": "acl-5fb85d36",
        "VpcId": "vpc-a01106c2",
        "Tags": [],
        "Entries": [
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": true,
                "RuleAction": "deny"
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": false,
                "RuleAction": "deny"
            }
        ],
        "IsDefault": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkAcl。

create-network-insights-access-scope

以下代码示例演示了如何使用 create-network-insights-access-scope。

Amazon CLI

创建网络访问范围

以下 create-network-insights-access-scope 示例将创建网络访问范围。

```
aws ec2 create-network-insights-access-scope \
    --cli-input-json file://access-scope-file.json
```

的内容access-scope-file.json：

```
{
    "MatchPaths": [
        {
            "Source": {
                "ResourceStatement": {
                    "Resources": [
                        "vpc-abcd12e3"
                    ]
                }
            }
         }
    ],
    "ExcludePaths": [
        {
            "Source": {
                "ResourceStatement": {
                    "ResourceTypes": [
                        "AWS::EC2::InternetGateway"
                    ]
                }
            }
        }
    ]
}
```

输出：

```
{
    "NetworkInsightsAccessScope": {
        "NetworkInsightsAccessScopeId": "nis-123456789abc01234",
        "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope/nis-123456789abc01234",
        "CreatedDate": "2022-01-25T19:20:28.796000+00:00",
        "UpdatedDate": "2022-01-25T19:20:28.797000+00:00"
    },
    "NetworkInsightsAccessScopeContent": {
        "NetworkInsightsAccessScopeId": "nis-123456789abc01234",
        "MatchPaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "Resources": [
                            "vpc-abcd12e3"
```

```
                        ]
                    }
                }
            }
        ],
        "ExcludePaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "ResourceTypes": [
                            "AWS::EC2::InternetGateway"
                        ]
                    }
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkInsightsAccessScope。

create-network-insights-path

以下代码示例演示了如何使用 create-network-insights-path。

Amazon CLI

创建路径

以下 create-network-insights-path 示例将创建一个路径。源是指定的互联网网关，而目标是指定的 EC2 实例。要确定是否可以使用指定的协议和端口到达目标，请使用 start-network-insights-analysis 命令分析路径。

```
aws ec2 create-network-insights-path \
    --source igw-0797cccdc9d73b0e5 \
    --destination i-0495d385ad28331c7 \
    --destination-port 22 \
    --protocol TCP
```

输出：

```
{
    "NetworkInsightsPaths": {
        "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
        "NetworkInsightsPathArn": "arn:aws:ec2:us-east-1:123456789012:network-
insights-path/nip-0b26f224f1d131fa8",
        "CreatedDate": "2021-01-20T22:43:46.933Z",
        "Source": "igw-0797cccdc9d73b0e5",
        "Destination": "i-0495d385ad28331c7",
        "Protocol": "tcp"
    }
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkInsightsPath。

create-network-interface-permission

以下代码示例演示了如何使用 create-network-interface-permission。

Amazon CLI

创建网络接口权限

此示例将向账户 123456789012 授予将网络接口 eni-1a2b3c4d 连接到实例的权限。

命令:

```
aws ec2 create-network-interface-permission --network-interface-id eni-1a2b3c4d --
aws-account-id 123456789012 --permission INSTANCE-ATTACH
```

输出：

```
{
  "InterfacePermission": {
      "PermissionState": {
          "State": "GRANTED"
      },
      "NetworkInterfacePermissionId": "eni-perm-06fd19020ede149ea",
      "NetworkInterfaceId": "eni-1a2b3c4d",
      "Permission": "INSTANCE-ATTACH",
```

```
        "AwsAccountId": "123456789012"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkInterfacePermission。

create-network-interface

以下代码示例演示了如何使用 create-network-interface。

Amazon CLI

示例 1：为网络接口指定 IPv4 地址

以下 create-network-interface 示例将使用指定的主 IPv4 地址为指定子网创建网络接口。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my network interface" \
    --groups sg-09dfba7ed20cda78b \
    --private-ip-address 10.0.8.17
```

输出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:6a:0f:9a:49:37",
        "NetworkInterfaceId": "eni-0492b355f0cf3b3f8",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.17",
```

```
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-17.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.17"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

示例 2：使用 IPv4 地址和 IPv6 地址创建网络接口

以下 `create-network-interface` 示例将使用由 Amazon EC2 选择的 IPv4 地址和 IPv6 地址为指定子网创建网络接口。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my dual stack network interface" \
    --ipv6-address-count 1 \
    --groups sg-09dfba7ed20cda78b
```

输出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my dual stack network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [
```

```
            {
                "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7",
                "IsPrimaryIpv6": false
            }
        ],
        "MacAddress": "06:b8:68:d2:b2:2d",
        "NetworkInterfaceId": "eni-05da417453f9a84bf",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.18",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.18"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b",
        "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
    }
}
```

示例 3：使用连接跟踪配置选项创建网络接口

以下 `create-network-interface` 示例将创建网络接口并配置空闲连接跟踪超时。

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --groups sg-02e57dbcfe0331c1b \
    --connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60
```

输出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "ConnectionTrackingConfiguration": {
```

```
            "TcpEstablishedTimeout": 86400,
            "UdpTimeout": 60
        },
        "Description": "",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:4c:53:de:6d:91",
        "NetworkInterfaceId": "eni-0c133586e08903d0b",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.94",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.94"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

示例 4：创建 Elastic Fabric Adapter

以下 `create-network-interface` 示例将创建 EFA。

```
aws ec2 create-network-interface \
    --interface-type efa \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my efa" \
    --groups sg-02e57dbcfe0331c1b
```

输出：

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my efa",
        "Groups": [
            {
                "GroupName": "my-efa-sg",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "efa",
        "Ipv6Addresses": [],
        "MacAddress": "06:d7:a4:f7:4d:57",
        "NetworkInterfaceId": "eni-034acc2885e862b65",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.180",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.180"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性网络接口。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNetworkInterface。

create-placement-group

以下代码示例演示了如何使用 create-placement-group。

Amazon CLI

创建置放群组

此示例命令将使用指定名称创建置放群组。

命令:

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

创建分区置放群组

此示例命令将创建一个名为 HDFS-Group-A 的分区置放群组，其中包含 5 个分区。

命令:

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --
partition-count 5
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePlacementGroup。

create-replace-root-volume-task

以下代码示例演示了如何使用 create-replace-root-volume-task。

Amazon CLI

示例 1：将根卷还原到其初始启动状态

以下 create-replace-root-volume-task 示例将实例 i-0123456789abcdefa 的根卷还原到其初始启动状态。

```
aws ec2 create-replace-root-volume-task \
    --instance-id i-0123456789abcdefa
```

输出：

```
{
    "ReplaceRootVolumeTask":
    {
        "InstanceId": "i-0123456789abcdefa",
            "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",
```

```
            "TaskState": "pending",
            "StartTime": "2022-03-14T15:06:38Z",
            "Tags": []
    }
}
```

示例 2：将根卷还原到特定快照

以下 `create-replace-root-volume-task` 示例将实例 i-0123456789abcdefa 的根卷还原到快照 snap-0abcdef1234567890。

```
aws ec2 create-replace-root-volume-task \
    --instance-id i-0123456789abcdefa \
    --snapshot-id  snap-0abcdef1234567890
```

输出：

```
{
    "ReplaceRootVolumeTask":
    {
        "InstanceId": "i-0123456789abcdefa",
        "ReplaceRootVolumeTaskId": "replacevol-0555566667777abcd",
        "TaskState": "pending",
        "StartTime": "2022-03-14T15:16:28Z",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的替换根卷。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplaceRootVolumeTask。

create-reserved-instances-listing

以下代码示例演示了如何使用 `create-reserved-instances-listing`。

Amazon CLI

在预留实例 Marketplace 中列出预留实例

以下 `create-reserved-instances-listing` 示例在预留实例 Marketplace 中为指定的预留实例创建列表。

```
aws ec2 create-reserved-instances-listing \
    --reserved-instances-id 5ec28771-05ff-4b9b-aa31-9e57dexample \
    --instance-count 3 \
    --price-schedules CurrencyCode=USD,Price=25.50 \
    --client-token 550e8400-e29b-41d4-a716-446655440000
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReservedInstancesListing。

create-restore-image-task

以下代码示例演示了如何使用 create-restore-image-task。

Amazon CLI

从 S3 存储桶还原 AMI

以下 create-restore-image-task 示例从 S3 存储桶还原 AMI。使用 describe-store-image-tasks 输出中的 S3ObjectKey `` and ``Bucket 值，指定 AMI 的对象键以及将 AMI 复制到的 S3 存储桶的名称，并为还原的 AMI 指定名称。名称对该账户在该区域中的 AMI 必须唯一。还原的 AMI 将收到一个新 AMI ID。

```
aws ec2 create-restore-image-task \
    --object-key ami-1234567890abcdef0.bin \
    --bucket my-ami-bucket \
    --name 'New AMI Name'
```

输出：

```
{
    "ImageId": "ami-0eab20fe36f83e1a8"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 S3 存储和还原 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRestoreImageTask。

create-route-table

以下代码示例演示了如何使用 create-route-table。

Amazon CLI

创建路由表

本示例为指定的 VPC 创建路由表。

命令:

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

输出：

```
{
    "RouteTable": {
        "Associations": [],
        "RouteTableId": "rtb-22574640",
        "VpcId": "vpc-a01106c2",
        "PropagatingVgws": [],
        "Tags": [],
        "Routes": [
            {
                "GatewayId": "local",
                "DestinationCidrBlock": "10.0.0.0/16",
                "State": "active"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRouteTable。

create-route

以下代码示例演示了如何使用 create-route。

Amazon CLI

创建路由

此示例将为指定的路由表创建路由。该路由匹配所有 IPv4 流量 (0.0.0.0/0)，并将其路由到指定的互联网网关。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-
block 0.0.0.0/0 --gateway-id igw-c0a643a9
```

此示例命令在路由表 rtb-g8ff4ea2 中创建路由。该路由匹配 IPv4 CIDR 块 10.0.0.0/16 的流量，并将其路由到 VPC 对等连接 pcx-111aaa22。此路由可将流量定向到 VPC 对等连接中的对等 VPC。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-
block 10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

此示例将在指定的路由表中创建一个匹配所有 IPv6 流量 (::/0) 的路由，并将其路由到指定的仅出口互联网网关。

命令:

```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-
block ::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoute。

create-security-group

以下代码示例演示了如何使用 create-security-group。

Amazon CLI

为 EC2-Classic 创建安全组

本示例将创建一个名为 MySecurityGroup 的安全组。

命令:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group"
```

输出：

```
{
    "GroupId": "sg-903004f8"
}
```

为 EC2-VPC 创建安全组

本示例为指定的 VPC 创建名为 MySecurityGroup 的安全组。

命令:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group" --vpc-id vpc-1a2b3c4d
```

输出：

```
{
    "GroupId": "sg-903004f8"
}
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的“使用安全组”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSecurityGroup。

create-snapshot

以下代码示例演示了如何使用 create-snapshot。

Amazon CLI

创建快照

此示例命令将创建卷 ID 为 vol-1234567890abcdef0 的卷的快照，并提供简短描述以标识快照。

命令:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is my
 root volume snapshot"
```

输出：

```
{
    "Description": "This is my root volume snapshot",
    "Tags": [],
    "Encrypted": false,
    "VolumeId": "vol-1234567890abcdef0",
    "State": "pending",
    "VolumeSize": 8,
    "StartTime": "2018-02-28T21:06:01.000Z",
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-066877671789bd71b"
}
```

创建具有标签的快照

此示例命令将创建快照并应用两个标签：purpose=prod 和 costcenter=123。

命令:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description 'Prod
 backup' --tag-specifications 'ResourceType=snapshot,Tags=[{Key=purpose,Value=prod},
{Key=costcenter,Value=123}]'
```

输出：

```
{
    "Description": "Prod backup",
    "Tags": [
        {
            "Value": "prod",
            "Key": "purpose"
        },
        {
            "Value": "123",
            "Key": "costcenter"
        }
     ],
     "Encrypted": false,
     "VolumeId": "vol-1234567890abcdef0",
     "State": "pending",
     "VolumeSize": 8,
     "StartTime": "2018-02-28T21:06:06.000Z",
```

```
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-09ed24a70bc19bbe4"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshot。

create-snapshots

以下代码示例演示了如何使用 create-snapshots。

Amazon CLI

示例 1：创建多卷快照

以下 create-snapshots 示例将创建连接到指定实例的所有卷的快照。

```
aws ec2 create-snapshots \
    --instance-specification InstanceId=i-1234567890abcdef0 \
    --description "This is snapshot of a volume from my-instance"
```

输出：

```
{
    "Snapshots": [
        {
            "Description": "This is a snapshot of a volume from my-instance",
            "Tags": [],
            "Encrypted": false,
            "VolumeId": "vol-0a01d2d5a34697479",
            "State": "pending",
            "VolumeSize": 16,
            "StartTime": "2019-08-05T16:58:19.000Z",
            "Progress": "",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-07f30e3909aa0045e"
        },
        {
            "Description": "This is a snapshot of a volume from my-instance",
            "Tags": [],
            "Encrypted": false,
            "VolumeId": "vol-02d0d4947008cb1a2",
```

```
            "State": "pending",
            "VolumeSize": 20,
            "StartTime": "2019-08-05T16:58:19.000Z",
            "Progress": "",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-0ec20b602264aad48"
        },
        ...
    ]
}
```

示例 2：使用源卷中的标签创建多卷快照

以下 create-snapshots 示例将创建连接到指定实例的所有卷的快照，并将标签从每个卷复制到其对应的快照中。

```
aws ec2 create-snapshots \
    --instance-specification InstanceId=i-1234567890abcdef0 \
    --copy-tags-from-source volume \
    --description "This is snapshot of a volume from my-instance"
```

输出：

```
{
    "Snapshots": [
        {
            "Description": "This is a snapshot of a volume from my-instance",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-volume"
                }
            ],
            "Encrypted": false,
            "VolumeId": "vol-02d0d4947008cb1a2",
            "State": "pending",
            "VolumeSize": 20,
            "StartTime": "2019-08-05T16:53:04.000Z",
            "Progress": "",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-053bfaeb821a458dd"
        }
        ...
```

```
    ]
}
```

示例 3：创建不包括根卷的多卷快照

以下 create-snapshots 示例将创建连接到指定实例的所有卷（根卷除外）的快照。

```
aws ec2 create-snapshots \
    --instance-specification InstanceId=i-1234567890abcdef0,ExcludeBootVolume=true
```

有关输出示例，请参阅示例 1。

示例 4：创建多卷快照并添加标签

以下 create-snapshots 示例将创建连接到指定实例的所有卷的快照，并向每个快照添加两个标签。

```
aws ec2 create-snapshots \
    --instance-specification InstanceId=i-1234567890abcdef0 \
    --tag-specifications 'ResourceType=snapshot,Tags=[{Key=Name,Value=backup},
{Key=costcenter,Value=123}]'
```

有关输出示例，请参阅示例 1。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshots。

create-spot-datafeed-subscription

以下代码示例演示了如何使用 create-spot-datafeed-subscription。

Amazon CLI

创建竞价型实例数据源

以下 create-spot-datafeed-subscription 示例将创建竞价型实例数据源。

```
aws ec2 create-spot-datafeed-subscription \
    --bucket amzn-s3-demo-bucket \
    --prefix spot-data-feed
```

输出：

```
{
    "SpotDatafeedSubscription": {
        "Bucket": "amzn-s3-demo-bucket",
        "OwnerId": "123456789012",
        "Prefix": "spot-data-feed",
        "State": "Active"
    }
}
```

数据源存储在您指定的 Amazon S3 存储桶中。此数据源的文件名采用以下格式。

```
amzn-s3-demo-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-
HH.n.abcd1234.gz
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的竞价型实例数据源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSpotDatafeedSubscription。

create-store-image-task

以下代码示例演示了如何使用 create-store-image-task。

Amazon CLI

将 AMI 存储在 S3 存储桶中

以下 create-store-image-task 示例将 AMI 存储在 S3 存储桶中。指定 AMI 的 ID 以及要在其中存储 AMI 的 S3 存储桶的名称。

```
aws ec2 create-store-image-task \
  --image-id ami-1234567890abcdef0 \
  --bucket my-ami-bucket
```

输出：

```
{
    "ObjectKey": "ami-1234567890abcdef0.bin"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 S3 存储和还原 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStoreImageTask。

create-subnet-cidr-reservation

以下代码示例演示了如何使用 create-subnet-cidr-reservation。

Amazon CLI

创建子网 CIDR 预留

以下 create-subnet-cidr-reservation 示例将为指定的子网和 CIDR 范围创建子网 CIDR 预留。

```
aws ec2 create-subnet-cidr-reservation \
    --subnet-id subnet-03c51e2eEXAMPLE \
    --reservation-type prefix \
    --cidr 10.1.0.20/26
```

输出：

```
{
    "SubnetCidrReservation": {
        "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",
        "SubnetId": "subnet-03c51e2e6cEXAMPLE",
        "Cidr": "10.1.0.16/28",
        "ReservationType": "prefix",
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的子网 CIDR 预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubnetCidrReservation。

create-subnet

以下代码示例演示了如何使用 create-subnet。

Amazon CLI

示例 1：创建仅具有 IPv4 CIDR 块的子网

以下 create-subnet 示例在指定的 VPC 中创建具有指定 IPv4 CIDR 块的子网。

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-subnet}]
```

输出：

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0e99b93155EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0e99b93155EXAMPLE"
    }
}
```

示例 2：创建同时具有 IPv4 和 IPv6 CIDR 块的子网

以下 create-subnet 示例在指定的 VPC 中，创建同时具有指定 IPv4 和 IPv6 CIDR 块的子网。

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --ipv6-cidr-block 2600:1f16:cfe:3660::/64 \
```

```
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-ipv6-
subnet}]
```

输出：

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0736441d38EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-06c5f904499fcc623",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-ipv6-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0736441d38EXAMPLE"
    }
}
```

示例 3：创建仅具有 IPv6 CIDR 块的子网

以下 `create-subnet` 示例在指定的 VPC 中创建具有指定 IPv6 CIDR 块的子网。

```
aws ec2 create-subnet \
```

```
    --vpc-id vpc-081ec835f3EXAMPLE \
    --ipv6-native \
    --ipv6-cidr-block 2600:1f16:115:200::/64 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv6-only-
subnet}]
```

输出：

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 0,
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-03f720e7deEXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": true,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-01ef639edde556709",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv6-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-03f720e7deEXAMPLE"
    }
}
```

有关更多信息，请参阅 Amazon VPC 用户指南中的 VPC 和子网。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubnet。

create-tags

以下代码示例演示了如何使用 create-tags。

Amazon CLI

示例 1：将标签添加到资源

以下 create-tags 示例将标签 Stack=production 添加到指定的映像，或者覆盖 AMI 的现有标签（其中标签键为 Stack）。

```
aws ec2 create-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=production
```

此命令不生成任何输出

示例 2：将标签添加到多个资源

以下 create-tags 示例为 AMI 和实例添加（或覆盖）两个标签。其中一个标签有一个键（webserver），但没有值（值设置为空字符串）。另一个标签有一个键（stack）和一个值（Production）。

```
aws ec2 create-tags \
    --resources ami-1a2b3c4d i-1234567890abcdef0 \
    --tags Key=webserver,Value=   Key=stack,Value=Production
```

此命令不生成任何输出

示例 3：添加包含特殊字符的标签

以下 create-tags 示例为实例添加标签 [Group]=test。方括号（[和]）是特殊字符，必须对其进行转义。以下示例还使用适用于每个环境的行延续字符。

如果使用的是 Windows，请用双引号（"）将具有特殊字符的元素引起来，然后在每个双引号字符前面添加反斜杠（\），如下所示。

```
aws ec2 create-tags ^
    --resources i-1234567890abcdef0 ^
    --tags Key=\"[Group]\",Value=test
```

如果使用的是 Windows PowerShell，请用双引号（"）将具有特殊字符的元素引起来，在每个双引号字符前面添加反斜杠（\），然后用单引号（'）将整个键和值结构引起来，如下所示。

```
aws ec2 create-tags `
    --resources i-1234567890abcdef0 `
    --tags 'Key=\"[Group]\",Value=test'
```

如果使用的是 Linux 或 OS X，请使用双引号（"）将具有特殊字符的元素引起来，然后使用单引号（'）将整个键和值结构引起来，如下所示。

```
aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags 'Key="[Group]",Value=test'
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的标记 Amazon EC2 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTags。

create-traffic-mirror-filter-rule

以下代码示例演示了如何使用 create-traffic-mirror-filter-rule。

Amazon CLI

为传入 TCP 流量创建筛选规则

以下 create-traffic-mirror-filter-rule 示例将创建一个规则，您可以使用该规则镜像所有传入 TCP 流量。在运行此命令之前，请使用 create-traffic-mirror-filter 创建流量镜像筛选条件。

```
aws ec2 create-traffic-mirror-filter-rule \
    --description 'TCP Rule' \
    --destination-cidr-block 0.0.0.0/0  \
    --protocol 6 \
    --rule-action accept \
    --rule-number 1 \
    --source-cidr-block 0.0.0.0/0 \
    --traffic-direction ingress \
    --traffic-mirror-filter-id tmf-04812ff784b25ae67
```

输出：

```
{
    "TrafficMirrorFilterRule": {
        "DestinationCidrBlock": "0.0.0.0/0",
        "TrafficMirrorFilterId": "tmf-04812ff784b25ae67",
        "TrafficMirrorFilterRuleId": "tmfr-02d20d996673f3732",
        "SourceCidrBlock": "0.0.0.0/0",
        "TrafficDirection": "ingress",
        "Description": "TCP Rule",
        "RuleNumber": 1,
        "RuleAction": "accept",
        "Protocol": 6
    },
    "ClientToken": "4752b573-40a6-4eac-a8a4-a72058761219"
}
```

有关更多信息，请参阅《Traffic Mirroring Guide》中的 Create a traffic mirror filter。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTrafficMirrorFilterRule。

create-traffic-mirror-filter

以下代码示例演示了如何使用 create-traffic-mirror-filter。

Amazon CLI

创建流量镜像筛选条件

以下 create-traffic-mirror-filter 示例将创建流量镜像筛选条件。创建筛选条件后，使用 create-traffic-mirror-filter-rule 添加规则。

```
aws ec2 create-traffic-mirror-filter \
    --description 'TCP Filter'
```

输出：

```
{
    "ClientToken": "28908518-100b-4987-8233-8c744EXAMPLE",
    "TrafficMirrorFilter": {
        "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
        "Description": "TCP Filter",
        "EgressFilterRules": [],
        "IngressFilterRules": [],
```

```
        "Tags": [],
        "NetworkServices": []
    }
}
```

有关更多信息，请参阅《Traffic Mirroring Guide》中的 Create a traffic mirror filter。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTrafficMirrorFilter。

create-traffic-mirror-session

以下代码示例演示了如何使用 create-traffic-mirror-session。

Amazon CLI

创建流量镜像会话

以下 create-traffic-mirror-session 命令为 25 字节的数据包的指定源和目标创建流量镜像会话。

```
aws ec2 create-traffic-mirror-session \
    --description 'example session' \
    --traffic-mirror-target-id tmt-07f75d8feeEXAMPLE \
    --network-interface-id eni-070203f901EXAMPLE \
    --session-number 1  \
    --packet-length 25 \
    --traffic-mirror-filter-id tmf-04812ff784EXAMPLE
```

输出：

```
{
    "TrafficMirrorSession": {
        "TrafficMirrorSessionId": "tms-08a33b1214EXAMPLE",
        "TrafficMirrorTargetId": "tmt-07f75d8feeEXAMPLE",
        "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
        "NetworkInterfaceId": "eni-070203f901EXAMPLE",
        "OwnerId": "111122223333",
        "PacketLength": 25,
        "SessionNumber": 1,
        "VirtualNetworkId": 7159709,
        "Description": "example session",
        "Tags": []
    },
```

```
    "ClientToken": "5236cffc-ee13-4a32-bb5b-388d9da09d96"
}
```

有关更多信息，请参阅《Traffic Mirroring Guide》中的 Create a traffic mirror session。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTrafficMirrorSession。

create-traffic-mirror-target

以下代码示例演示了如何使用 create-traffic-mirror-target。

Amazon CLI

创建网络负载均衡器流量镜像目标

以下 create-traffic-mirror-target 示例将创建网络负载均衡器流量镜像目标。

```
aws ec2 create-traffic-mirror-target \
    --description 'Example Network Load Balancer Target' \
    --network-load-balancer-arn arn:aws:elasticloadbalancing:us-
east-1:111122223333:loadbalancer/net/NLB/7cdec873EXAMPLE
```

输出：

```
{
    "TrafficMirrorTarget": {
        "Type": "network-load-balancer",
        "Tags": [],
        "Description": "Example Network Load Balancer Target",
        "OwnerId": "111122223333",
        "NetworkLoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:724145273726:loadbalancer/net/NLB/7cdec873EXAMPLE",
        "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE"
    },
    "ClientToken": "d5c090f5-8a0f-49c7-8281-72c796a21f72"
}
```

创建网络流量镜像目标

以下 create-traffic-mirror-target 示例将创建一个网络接口流量镜像目标。

```
aws ec2 create-traffic-mirror-target \
```

```
    --description 'Network interface target' \
    --network-interface-id eni-eni-01f6f631eEXAMPLE
```

输出：

```
{
    "ClientToken": "5289a345-0358-4e62-93d5-47ef3061d65e",
    "TrafficMirrorTarget": {
        "Description": "Network interface target",
        "NetworkInterfaceId": "eni-01f6f631eEXAMPLE",
        "TrafficMirrorTargetId": "tmt-02dcdbe2abEXAMPLE",
        "OwnerId": "111122223333",
        "Type": "network-interface",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Traffic Mirroring Guide》中的 Create a traffic mirror target。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTrafficMirrorTarget。

create-transit-gateway-connect-peer

以下代码示例演示了如何使用 create-transit-gateway-connect-peer。

Amazon CLI

创建 Transit Gateway Connect 对等节点

以下 create-transit-gateway-connect-peer 示例将创建一个 Connect 对等节点。

```
aws ec2 create-transit-gateway-connect-peer \
    --transit-gateway-attachment-id tgw-attach-0f0927767cEXAMPLE \
    --peer-address 172.31.1.11 \
    --inside-cidr-blocks 169.254.6.0/29
```

输出：

```
{
    "TransitGatewayConnectPeer": {
        "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",
```

```
        "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",
        "State": "pending",
        "CreationTime": "2021-10-13T03:35:17.000Z",
        "ConnectPeerConfiguration": {
            "TransitGatewayAddress": "10.0.0.234",
            "PeerAddress": "172.31.1.11",
            "InsideCidrBlocks": [
                "169.254.6.0/29"
            ],
            "Protocol": "gre",
            "BgpConfigurations": [
                {
                    "TransitGatewayAsn": 64512,
                    "PeerAsn": 64512,
                    "TransitGatewayAddress": "169.254.6.2",
                    "PeerAddress": "169.254.6.1",
                    "BgpStatus": "down"
                },
                {
                    "TransitGatewayAsn": 64512,
                    "PeerAsn": 64512,
                    "TransitGatewayAddress": "169.254.6.3",
                    "PeerAddress": "169.254.6.1",
                    "BgpStatus": "down"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayConnectPeer。

`create-transit-gateway-connect`

以下代码示例演示了如何使用 `create-transit-gateway-connect`。

Amazon CLI

创建 Transit Gateway Connect 连接

以下 create-transit-gateway-connect 示例使用“gre”协议为指定连接创建一个 Connect 连接。

```
aws ec2 create-transit-gateway-connect \
    --transport-transit-gateway-attachment-id tgw-attach-0a89069f57EXAMPLE \
    --options "Protocol=gre"
```

输出：

```
{
    "TransitGatewayConnect": {
        "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",
        "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
        "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
        "State": "pending",
        "CreationTime": "2021-03-09T19:59:17+00:00",
        "Options": {
            "Protocol": "gre"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayConnect。

create-transit-gateway-multicast-domain

以下代码示例演示了如何使用 create-transit-gateway-multicast-domain。

Amazon CLI

示例 1：创建 IGMP 组播域

以下 create-transit-gateway-multicast-domain 示例将为指定的中转网关创建组播域。禁用静态源后，位于与组播域关联的子网中的任何实例都可以发送组播流量。如果至少有一个成员使用 IGMP 协议，则必须启用 IGMPv2 支持。

```
aws ec2 create-transit-gateway-multicast-domain \
    --transit-gateway-id tgw-0bf0bffefaEXAMPLE \
```

```
    --options StaticSourcesSupport=disable,Igmpv2Support=enable
```

输出：

```
{
    "TransitGatewayMulticastDomain": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c9e29e2a7EXAMPLE",
        "TransitGatewayId": "tgw-0bf0bffefaEXAMPLE",
        "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
west-2:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-0c9e29e2a7EXAMPLE",
        "OwnerId": "123456789012",
        "Options": {
            "Igmpv2Support": "enable",
            "StaticSourcesSupport": "disable",
            "AutoAcceptSharedAssociations": "disable"
        },
        "State": "pending",
        "CreationTime": "2021-09-29T22:17:13.000Z"
    }
}
```

示例 2：创建静态组播域

以下 create-transit-gateway-multicast-domain 示例将为指定的中转网关创建组播域。启用静态源后，必须静态添加源。

```
aws ec2 create-transit-gateway-multicast-domain \
    --transit-gateway-id tgw-0bf0bffefaEXAMPLE \
    --options StaticSourcesSupport=enable,Igmpv2Support=disable
```

输出：

```
{
    "TransitGatewayMulticastDomain": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-000fb24d04EXAMPLE",
        "TransitGatewayId": "tgw-0bf0bffefaEXAMPLE",
        "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
west-2:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-000fb24d04EXAMPLE",
        "OwnerId": "123456789012",
        "Options": {
```

```
            "Igmpv2Support": "disable",
            "StaticSourcesSupport": "enable",
            "AutoAcceptSharedAssociations": "disable"
        },
        "State": "pending",
        "CreationTime": "2021-09-29T22:20:19.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的管理组播域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayMulticastDomain。

create-transit-gateway-peering-attachment

以下代码示例演示了如何使用 create-transit-gateway-peering-attachment。

Amazon CLI

创建中转网关对等连接

以下 create-transit-gateway-peering-attachment 示例将在两个指定的中转网关之间创建对等连接请求。

```
aws ec2 create-transit-gateway-peering-attachment \
    --transit-gateway-id tgw-123abc05e04123abc \
    --peer-transit-gateway-id tgw-11223344aabbcc112 \
    --peer-account-id 123456789012 \
    --peer-region us-east-2
```

输出：

```
{
    "TransitGatewayPeeringAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
        "RequesterTgwInfo": {
            "TransitGatewayId": "tgw-123abc05e04123abc",
            "OwnerId": "123456789012",
            "Region": "us-west-2"
        },
        "AccepterTgwInfo": {
```

```
            "TransitGatewayId": "tgw-11223344aabbcc112",
            "OwnerId": "123456789012",
            "Region": "us-east-2"
        },
        "State": "initiatingRequest",
        "CreationTime": "2019-12-09T11:38:05.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关对等连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayPeeringAttachment。

create-transit-gateway-policy-table

以下代码示例演示了如何使用 create-transit-gateway-policy-table。

Amazon CLI

创建中转网关策略表

以下 create-transit-gateway-policy-table 示例将为指定的中转网关创建中转网关策略表。

```
aws ec2 create-transit-gateway-policy-table \
    --transit-gateway-id tgw-067f8505c18f0bd6e
```

输出：

```
{
    "TransitGatewayPolicyTable": {
        "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
        "TransitGatewayId": "tgw-067f8505c18f0bd6e",
        "State": "pending",
        "CreationTime": "2023-11-28T16:36:43+00:00"
    }
}
```

有关更多信息，请参阅《中转网关用户指南》中的中转网关策略表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayPolicyTable。

create-transit-gateway-prefix-list-reference

以下代码示例演示了如何使用 create-transit-gateway-prefix-list-reference。

Amazon CLI

创建对前缀列表的引用

以下 create-transit-gateway-prefix-list-reference 示例在指定的中转网关路由表中创建对指定前缀列表的引用。

```
aws ec2 create-transit-gateway-prefix-list-reference \
    --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
    --prefix-list-id pl-1111122222222333 \
    --transit-gateway-attachment-id tgw-attach-aaaaaabbbbbb11111
```

输出：

```
{
    "TransitGatewayPrefixListReference": {
        "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
        "PrefixListId": "pl-1111122222222333",
        "PrefixListOwnerId": "123456789012",
        "State": "pending",
        "Blackhole": false,
        "TransitGatewayAttachment": {
            "TransitGatewayAttachmentId": "tgw-attach-aaaaaabbbbbb11111",
            "ResourceType": "vpc",
            "ResourceId": "vpc-112233445566aabbc"
        }
    }
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Create a prefix list reference。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayPrefixListReference。

create-transit-gateway-route-table

以下代码示例演示了如何使用 create-transit-gateway-route-table。

Amazon CLI

创建中转网关路由表

以下 create-transit-gateway-route-table 示例将为指定的中转网关创建路由表。

```
aws ec2 create-transit-gateway-route-table \
    --transit-gateway-id tgw-0262a0e521EXAMPLE
```

输出：

```
{
    "TransitGatewayRouteTable": {
        "TransitGatewayRouteTableId": "tgw-rtb-0960981be7EXAMPLE",
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
        "State": "pending",
        "DefaultAssociationRouteTable": false,
        "DefaultPropagationRouteTable": false,
        "CreationTime": "2019-07-10T19:01:46.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的创建中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayRouteTable。

create-transit-gateway-route

以下代码示例演示了如何使用 create-transit-gateway-route。

Amazon CLI

创建中转网关路由

以下 create-transit-gateway-route 示例将为指定的路由表创建具有指定目标的路由。

```
aws ec2 create-transit-gateway-route \
    --destination-cidr-block 10.0.2.0/24 \
    --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE \
    --transit-gateway-attachment-id tgw-attach-0b5968d3b6EXAMPLE
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "10.0.2.0/24",
        "TransitGatewayAttachments": [
            {
                "ResourceId": "vpc-0065acced4EXAMPLE",
                "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
                "ResourceType": "vpc"
            }
        ],
        "Type": "static",
        "State": "active"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayRoute。

create-transit-gateway-vpc-attachment

以下代码示例演示了如何使用 create-transit-gateway-vpc-attachment。

Amazon CLI

示例 1：将中转网关与 VPC 相关联

以下 create-transit-gateway-vpc-attachment 示例将创建指定 VPC 的中转网关连接。

```
aws ec2 create-transit-gateway-vpc-attachment \
    --transit-gateway-id tgw-0262a0e521EXAMPLE \
    --vpc-id vpc-07e8ffd50f49335df \
    --subnet-id subnet-0752213d59EXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
```

```
        "VpcId": "vpc-07e8ffd50fEXAMPLE",
        "VpcOwnerId": "111122223333",
        "State": "pending",
        "SubnetIds": [
            "subnet-0752213d59EXAMPLE"
        ],
        "CreationTime": "2019-07-10T17:33:46.000Z",
        "Options": {
            "DnsSupport": "enable",
            "Ipv6Support": "disable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的创建 VPC 的中转网关连接。

示例 2：将中转网关与 VPC 中的多个子网相关联

以下 `create-transit-gateway-vpc-attachment` 示例将创建指定 VPC 和子网的中转网关连接。

```
aws ec2 create-transit-gateway-vpc-attachment \
    --transit-gateway-id tgw-02f776b1a7EXAMPLE  \
    --vpc-id vpc-3EXAMPLE \
    --subnet-ids "subnet-dEXAMPLE" "subnet-6EXAMPLE"
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0e141e0bebEXAMPLE",
        "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
        "VpcId": "vpc-3EXAMPLE",
        "VpcOwnerId": "111122223333",
        "State": "pending",
        "SubnetIds": [
            "subnet-6EXAMPLE",
            "subnet-dEXAMPLE"
        ],
        "CreationTime": "2019-12-17T20:07:52.000Z",
        "Options": {
            "DnsSupport": "enable",
```

```
            "Ipv6Support": "disable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的创建 VPC 的中转网关连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGatewayVpcAttachment。

create-transit-gateway

以下代码示例演示了如何使用 create-transit-gateway。

Amazon CLI

创建中转网关

以下 create-transit-gateway 示例将创建一个中转网关。

```
aws ec2 create-transit-gateway \
    --description MyTGW \
    --
options AmazonSideAsn=64516,AutoAcceptSharedAttachments=enable,DefaultRouteTableAssociation=
```

输出：

```
{
    "TransitGateway": {
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
        "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-gateway/
tgw-0262a0e521EXAMPLE",
        "State": "pending",
        "OwnerId": "111122223333",
        "Description": "MyTGW",
        "CreationTime": "2019-07-10T14:02:12.000Z",
        "Options": {
            "AmazonSideAsn": 64516,
            "AutoAcceptSharedAttachments": "enable",
            "DefaultRouteTableAssociation": "enable",
            "AssociationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
            "DefaultRouteTablePropagation": "enable",
```

```
            "PropagationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
            "VpnEcmpSupport": "enable",
            "DnsSupport": "enable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的创建中转网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTransitGateway。

create-verified-access-endpoint

以下代码示例演示了如何使用 create-verified-access-endpoint。

Amazon CLI

创建验证访问端点

以下 create-verified-access-endpoint 示例将为指定的已验证访问组创建已验证访问端点。指定的网络接口和安全组必须属于同一 VPC。

```
aws ec2 create-verified-access-endpoint \
    --verified-access-group-id vagr-0dbe967baf14b7235 \
    --endpoint-type network-interface \
    --attachment-type vpc \
    --domain-certificate-arn arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE \
    --application-domain example.com \
    --endpoint-domain-prefix my-ava-app \
    --security-group-ids sg-004915970c4c8f13a \
    --network-interface-
options NetworkInterfaceId=eni-0aec70418c8d87a0f,Protocol=https,Port=443 \
    --tag-specifications ResourceType=verified-access-
endpoint,Tags=[{Key=Name,Value=my-va-endpoint}]
```

输出：

```
{
    "VerifiedAccessEndpoint": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
```

```
        "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
        "ApplicationDomain": "example.com",
        "EndpointType": "network-interface",
        "AttachmentType": "vpc",
        "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
        "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
        "SecurityGroupIds": [
            "sg-004915970c4c8f13a"
        ],
        "NetworkInterfaceOptions": {
            "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
            "Protocol": "https",
            "Port": 443
        },
        "Status": {
            "Code": "pending"
        },
        "Description": "",
        "CreationTime": "2023-08-25T20:54:43",
        "LastUpdatedTime": "2023-08-25T20:54:43",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-va-endpoint"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVerifiedAccessEndpoint。

`create-verified-access-group`

以下代码示例演示了如何使用 `create-verified-access-group`。

Amazon CLI

创建已验证访问组

以下 create-verified-access-group 示例将为指定的已验证访问实例创建一个已验证访问组。

```
aws ec2 create-verified-access-group \
    --verified-access-instance-id vai-0ce000c0b7643abea \
    --tag-specifications ResourceType=verified-access-
group,Tags=[{Key=Name,Value=my-va-group}]
```

输出：

```
{
    "VerifiedAccessGroup": {
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "",
        "Owner": "123456789012",
        "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
        "CreationTime": "2023-08-25T19:55:19",
        "LastUpdatedTime": "2023-08-25T19:55:19",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-va-group"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVerifiedAccessGroup。

create-verified-access-instance

以下代码示例演示了如何使用 create-verified-access-instance。

Amazon CLI

创建已验证访问实例

以下 create-verified-access-instance 示例将创建一个带有名称标签的已验证访问实例。

```
aws ec2 create-verified-access-instance \
    --tag-specifications ResourceType=verified-access-
instance,Tags=[{Key=Name,Value=my-va-instance}]
```

输出：

```
{
    "VerifiedAccessInstance": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "",
        "VerifiedAccessTrustProviders": [],
        "CreationTime": "2023-08-25T18:27:56",
        "LastUpdatedTime": "2023-08-25T18:27:56",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-va-instance"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVerifiedAccessInstance。

create-verified-access-trust-provider

以下代码示例演示了如何使用 create-verified-access-trust-provider。

Amazon CLI

创建已验证访问信任提供商

以下 create-verified-access-trust-provider 示例使用 Amazon Identity Center 设置已验证访问信任提供商。

```
aws ec2 create-verified-access-trust-provider \
    --trust-provider-type user \
    --user-trust-provider-type iam-identity-center \
    --policy-reference-name idc \
```

```
    --tag-specifications ResourceType=verified-access-trust-
provider,Tags=[{Key=Name,Value=my-va-trust-provider}]
```

输出：

```
{
    "VerifiedAccessTrustProvider": {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "Description": "",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center",
        "PolicyReferenceName": "idc",
        "CreationTime": "2023-08-25T18:40:36",
        "LastUpdatedTime": "2023-08-25T18:40:36",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-va-trust-provider"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问的信任提供商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVerifiedAccessTrustProvider。

create-volume

以下代码示例演示了如何使用 create-volume。

Amazon CLI

创建空通用型 SSD（gp2）卷

以下 create-volume 示例将在指定的可用区中创建一个 80 GiB 的通用型 SSD（gp2）卷。请注意，当前区域必须为 us-east-1，或者您可以添加 --region 参数来为命令指定区域。

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
```

```
    --availability-zone us-east-1a
```

输出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
}
```

如果您未指定卷类型，则默认卷类型为 gp2。

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

示例 2：从快照创建预调配 IOPS SSD（io1）卷

以下 `create-volume` 示例将使用指定的快照在指定的可用区中创建具有 1000 预调配 IOPS 的预调配 IOPS SSD（io1）卷。

```
aws ec2 create-volume \
    --volume-type io1 \
    --iops 1000 \
    --snapshot-id snap-066877671789bd71b \
    --availability-zone us-east-1a
```

输出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "io1",
```

```
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 1000,
    "SnapshotId": "snap-066877671789bd71b",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 500
}
```

示例 3：创建加密卷

以下 create-volume 示例使用 EBS 加密的默认 CMK 创建加密卷。如果默认情况下禁用了加密，则必须按以下方式指定 --encrypted 参数。

```
aws ec2 create-volume \
    --size 80 \
    --encrypted \
    --availability-zone us-east-1a
```

输出：

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": true,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
}
```

如果默认情况下启用了加密，则即使没有 --encrypted 参数，以下示例命令会创建一个加密卷。

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

如果您使用 --kms-key-id 参数指定客户托管的 CMK，则即使默认情况下已启用加密，也必须指定 --encrypted 参数。

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
    --encrypted \
    --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \
    --availability-zone us-east-1a
```

示例 4：创建带有标签的卷

以下 `create-volume` 示例将创建一个卷并添加两个标签。

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=purpose,Value=production},
{Key=cost-center,Value=cc123}]'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVolume。

create-vpc-endpoint-connection-notification

以下代码示例演示了如何使用 `create-vpc-endpoint-connection-notification`。

Amazon CLI

创建端点连接通知

此示例将为特定端点服务创建通知，当接口端点已连接到您的服务以及您的服务已接受端点时，该通知会提醒您。

命令:

```
aws ec2 create-vpc-endpoint-connection-notification --connection-notification-
arn arn:aws:sns:us-east-2:123456789012:VpceNotification --connection-
events Connect Accept --service-id vpce-svc-1237881c0d25a3abc
```

输出：

```
{
```

```
   "ConnectionNotification": {
       "ConnectionNotificationState": "Enabled",
       "ConnectionNotificationType": "Topic",
       "ServiceId": "vpce-svc-1237881c0d25a3abc",
       "ConnectionEvents": [
           "Accept",
           "Connect"
       ],
       "ConnectionNotificationId": "vpce-nfn-008776de7e03f5abc",
       "ConnectionNotificationArn": "arn:aws:sns:us-
east-2:123456789012:VpceNotification"
   }
 }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcEndpointConnectionNotification。

create-vpc-endpoint-service-configuration

以下代码示例演示了如何使用 create-vpc-endpoint-service-configuration。

Amazon CLI

示例 1：为接口端点创建端点服务配置

以下 create-vpc-endpoint-service-configuration 示例将使用网络负载均衡器 nlb-vpce 创建 VPC 端点服务配置。此示例还指定必须接受通过接口端点连接到服务的请求。

```
aws ec2 create-vpc-endpoint-service-configuration \
    --network-load-balancer-arns arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/nlb-vpce/e94221227f1ba532 \
    --acceptance-required
```

输出：

```
{
   "ServiceConfiguration": {
       "ServiceType": [
           {
               "ServiceType": "Interface"
           }
```

```
        ],
        "NetworkLoadBalancerArns": [
            "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/net/
nlb-vpce/e94221227f1ba532"
        ],
        "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-03d5ebb7d9579a2b3",
        "ServiceState": "Available",
        "ServiceId": "vpce-svc-03d5ebb7d9579a2b3",
        "AcceptanceRequired": true,
        "AvailabilityZones": [
            "us-east-1d"
        ],
        "BaseEndpointDnsNames": [
            "vpce-svc-03d5ebb7d9579a2b3.us-east-1.vpce.amazonaws.com"
        ]
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Create an endpoint service。

示例 2：为网关负载均衡器端点创建端点服务配置

以下 create-vpc-endpoint-service-configuration 示例将使用网关负载均衡器 GWLBService 创建 VPC 端点服务配置。系统会自动接受通过网关负载均衡器端点连接到服务的请求。

```
aws ec2 create-vpc-endpoint-service-configuration \
    --gateway-load-balancer-arns arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/gwy/GWLBService/123123123123abcc \
    --no-acceptance-required
```

输出：

```
{
    "ServiceConfiguration": {
        "ServiceType": [
            {
                "ServiceType": "GatewayLoadBalancer"
            }
        ],
        "ServiceId": "vpce-svc-123123a1c43abc123",
        "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",
```

```
        "ServiceState": "Available",
        "AvailabilityZones": [
            "us-east-1d"
        ],
        "AcceptanceRequired": false,
        "ManagesVpcEndpoints": false,
        "GatewayLoadBalancerArns": [
            "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/gwy/
GWLBService/123123123123abcc"
        ]
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Create a Gateway Load Balancer endpoint service。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcEndpointServiceConfiguration。

create-vpc-endpoint

以下代码示例演示了如何使用 create-vpc-endpoint。

Amazon CLI

示例 1：创建网关端点

以下 create-vpc-endpoint 示例将在 us-east-1 区域中的 VPC vpc-1a2b3c4d 和 Amazon S3 之间创建一个网关 VPC 端点，并将路由表 rtb-11aa22bb 与该端点相关联。

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --service-name com.amazonaws.us-east-1.s3 \
    --route-table-ids rtb-11aa22bb
```

输出：

```
{
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"\",
\"Effect\":\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",
        "VpcId": "vpc-1a2b3c4d",
```

```
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "RouteTableIds": [
            "rtb-11aa22bb"
        ],
        "VpcEndpointId": "vpc-1a2b3c4d",
        "CreationTimestamp": "2015-05-15T09:40:50Z"
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Create a gateway endpoint。

示例 2：创建接口端点

以下 `create-vpc-endpoint` 示例将在 `us-east-1` 区域中的 VPC `vpc-1a2b3c4d` 和 Amazon S3 之间创建一个接口 VPC 端点。该命令在将子网 `subnet-1a2b3c4d` 中创建端点，将其与安全组 `sg-1a2b3c4d` 相关联，并添加一个键为“Service”且值为“S3”的标签。

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --vpc-endpoint-type Interface \
    --service-name com.amazonaws.us-east-1.s3 \
    --subnet-ids subnet-7b16de0c \
    --security-group-id sg-1a2b3c4d \
    --tag-specifications ResourceType=vpc-endpoint,Tags=[{Key=service,Value=S3}]
```

输出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-1a2b3c4d5e6f1a2b3",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-1a2b3c4d",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-1a2b3c4d"
        ],
        "Groups": [
            {
                "GroupId": "sg-1a2b3c4d",
```

```
                "GroupName": "default"
            }
        ],
        "PrivateDnsEnabled": false,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-0b16f0581c8ac6877"
        ],
        "DnsEntries": [
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg-us-east-1c.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            }
        ],
        "CreationTimestamp": "2021-03-05T14:46:16.030000+00:00",
        "Tags": [
            {
                "Key": "service",
                "Value": "S3"
            }
        ],
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Create an interface VPC endpoint。

示例 3：创建网关负载均衡器端点

以下 create-vpc-endpoint 示例将在 VPC vpc-111122223333aabbc 和使用网关负载均衡器配置的服务之间创建一个网关负载均衡器端点。

```
aws ec2 create-vpc-endpoint \
    --service-name com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123 \
    --vpc-endpoint-type GatewayLoadBalancer \
    --vpc-id vpc-111122223333aabbc \
```

```
    --subnet-ids subnet-0011aabbcc2233445
```

输出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-aabbaabbaabbaabba",
        "VpcEndpointType": "GatewayLoadBalancer",
        "VpcId": "vpc-111122223333aabbc",
        "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",
        "State": "pending",
        "SubnetIds": [
            "subnet-0011aabbcc2233445"
        ],
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-01010120203030405"
        ],
        "CreationTimestamp": "2020-11-11T08:06:03.522Z",
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Gateway Load Balancer endpoints。

示例 4：创建资源端点

以下 `create-vpc-endpoint` 示例创建了一个资源端点。

```
aws ec2 create-vpc-endpoint \
    --vpc-endpoint-type Resource \
    --vpc-id vpc-111122223333aabbc \
    --subnet-ids subnet-0011aabbcc2233445 \
    --resource-configuration-arn arn:aws:vpc-lattice-us-
east-1:123456789012:resourceconfiguration/rcfg-0123abcde98765432
```

输出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-00939a7ed9EXAMPLE",
```

```
        "VpcEndpointType": "Resource",
        "VpcId": "vpc-111122223333aabbc",
        "State": "Pending",
        "SubnetIds": [
            "subnet-0011aabbcc2233445"
        ],
        "Groups": [
            {
                "GroupId": "sg-03e2f15fbfc09b000",
                "GroupName": "default"
            }
        ],
        "IpAddressType": "IPV4",
        "PrivateDnsEnabled": false,
        "CreationTimestamp": "2025-02-06T23:38:49.525000+00:00",
        "Tags": [],
        "OwnerId": "123456789012",
        "ResourceConfigurationArn": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-0123abcde98765432"
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Resource endpoints。

示例 5：创建服务网络端点

以下 `create-vpc-endpoint` 示例创建了一个服务网络端点。

```
aws ec2 create-vpc-endpoint \
    --vpc-endpoint-type ServiceNetwork \
    --vpc-id vpc-111122223333aabbc \
    --subnet-ids subnet-0011aabbcc2233445 \
    --service-network-arn arn:aws:vpc-lattice:us-east-1:123456789012:servicenetwork/
sn-0101abcd5432abcd0 \
    --security-group-ids sg-0123456789012abcd
```

输出：

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-0f00567fa8EXAMPLE",
        "VpcEndpointType": "ServiceNetwork",
        "VpcId": "vpc-111122223333aabbc",
```

```
        "State": "Pending",
        "SubnetIds": [
            "subnet-0011aabbcc2233445"
        ],
        "Groups": [
            {
                "GroupId": "sg-0123456789012abcd",
                "GroupName": "my-security-group"
            }
        ],
        "IpAddressType": "IPV4",
        "PrivateDnsEnabled": false,
        "CreationTimestamp": "2025-02-06T23:44:20.449000+00:00",
        "Tags": [],
        "OwnerId": "123456789012",
        "ServiceNetworkArn": "arn:aws:vpc-lattice:us-
east-1:123456789012:servicenetwork/sn-0101abcd5432abcd0"
    }
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Service network endpoints。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcEndpoint。

`create-vpc-peering-connection`

以下代码示例演示了如何使用 `create-vpc-peering-connection`。

Amazon CLI

在 VPC 之间创建 VPC 对等连接

此示例在您的 VPC vpc-1a2b3c4d 和 vpc-11122233 之间请求对等连接。

命令:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-
id vpc-11122233
```

输出：

```
{
```

```
    "VpcPeeringConnection": {
        "Status": {
            "Message": "Initiating Request to 444455556666",
            "Code": "initiating-request"
        },
        "Tags": [],
        "RequesterVpcInfo": {
            "OwnerId": "444455556666",
            "VpcId": "vpc-1a2b3c4d",
            "CidrBlock": "10.0.0.0/28"
        },
        "VpcPeeringConnectionId": "pcx-111aaa111",
        "ExpirationTime": "2014-04-02T16:13:36.000Z",
        "AccepterVpcInfo": {
            "OwnerId": "444455556666",
            "VpcId": "vpc-11122233"
        }
    }
}
```

与其他账户中的 VPC 创建 VPC 对等连接

此示例在您的 VPC（vpc-1a2b3c4d）和属于 Amazon 账户 123456789012 的 VPC（vpc-11122233）之间请求对等连接。

命令:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-
id vpc-11122233 --peer-owner-id 123456789012
```

创建与不同区域内的 VPC 之间的 VPC 对等连接

此示例在当前区域中的 VPC（vpc-1a2b3c4d）与您在 us-west-2 区域中的账户中的 VPC（vpc-11122233）之间请求对等连接。

命令:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-
id vpc-11122233 --peer-region us-west-2
```

此示例在当前区域中的 VPC（vpc-1a2b3c4d）和 us-west-2 区域中属于 Amazon 账户 123456789012 的 VPC（vpc-11122233）之间请求对等连接。

命令:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-
id vpc-11122233 --peer-owner-id 123456789012 --peer-region us-west-2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcPeeringConnection。

create-vpc

以下代码示例演示了如何使用 create-vpc。

Amazon CLI

示例 1：创建 VPC

以下 create-vpc 示例创建具有指定 IPv4 CIDR 块和名称标签的 VPC。

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

输出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-5EXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0a60eb65b4EXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-07501b79ecEXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
```

```
        "Tags": [
            {
                "Key": "Name",
                "Value": MyVpc"
            }
        ]
    }
}
```

示例 2：创建具有专用租赁的 VPC

以下 `create-vpc` 示例创建具有指定 IPv4 CIDR 块和专用租赁的 VPC。

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --instance-tenancy dedicated
```

输出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-19edf471",
        "State": "pending",
        "VpcId": "vpc-0a53287fa4EXAMPLE",
        "OwnerId": "111122223333",
        "InstanceTenancy": "dedicated",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-00b24cc1c2EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

示例 3：创建具有 IPv6 CIDR 块的 VPC

以下 create-vpc 示例创建具有 Amazon 提供的 IPv6 CIDR 块的 VPC。

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --amazon-provided-ipv6-cidr-block
```

输出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-dEXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0fc5e3406bEXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-068432c60bEXAMPLE",
                "Ipv6CidrBlock": "",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                },
                "Ipv6Pool": "Amazon",
                "NetworkBorderGroup": "us-west-2"
            }
        ],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0669f8f9f5EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

示例 4：从 IPAM 池中创建具有 CIDR 的 VPC

以下 create-vpc 示例从 Amazon VPC IP 地址管理器（IPAM）池，创建具有 CIDR 的 VPC。

Linux 和 macOS：

```
aws ec2 create-vpc \
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 \
    --tag-specifications ResourceType=vpc,Tags='[{Key=Environment,Value="Preprod"},
{Key=Owner,Value="Build Team"}]'
```

Windows:

```
aws ec2 create-vpc ^
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 ^
    --tag-specifications ResourceType=vpc,Tags=[{Key=Environment,Value="Preprod"},
{Key=Owner,Value="Build Team"}]
```

输出：

```
{
    "Vpc": {
        "CidrBlock": "10.0.1.0/24",
        "DhcpOptionsId": "dopt-2afccf50",
        "State": "pending",
        "VpcId": "vpc-010e1791024eb0af9",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0a77de1d803226d4b",
                "CidrBlock": "10.0.1.0/24",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Environment",
                "Value": "Preprod"
            },
            {
                "Key": "Owner",
```

```
                "Value": "Build Team"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的创建使用 IPAM 池 CIDR 的 VPC。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpc。

create-vpn-connection-route

以下代码示例演示了如何使用 create-vpn-connection-route。

Amazon CLI

为 VPN 连接创建静态路由

此示例为指定的 VPN 连接创建静态路由。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --destination-cidr-block 11.12.0.0/16
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpnConnectionRoute。

create-vpn-connection

以下代码示例演示了如何使用 create-vpn-connection。

Amazon CLI

示例 1：创建使用动态路由的 VPN 连接

以下 create-vpn-connection 示例将在指定的虚拟私有网关和指定的客户网关之间创建 VPN 连接，并将标签应用于 VPN 连接。输出包括 XML 格式的客户网关设备的配置信息。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
```

```
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --tag-specification 'ResourceType=vpn-connection,Tags=[{Key=Name,Value=BGP-
VPN}]'
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "...configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
                {}
            ]
        },
        "Routes": [],
        "Tags": [
             {
                "Key": "Name",
                "Value": "BGP-VPN"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

示例 2：创建使用静态路由的 VPN 连接

以下 `create-vpn-connection` 示例将在指定的虚拟私有网关和指定的客户网关之间创建 VPN 连接。这些选项指定静态路由。输出包括 XML 格式的客户网关设备的配置信息。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options "{\"StaticRoutesOnly\":true}"
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": true,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
                {}
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

示例 3：创建 VPN 连接并指定您自己的内部 CIDR 和预共享密钥

以下 `create-vpn-connection` 示例将创建 VPN 连接，并为每个隧道指定内部 IP 地址 CIDR 块和自定义预共享密钥。将在 `CustomerGatewayConfiguration` 信息中返回指定值。

```
aws ec2 create-vpn-connection \
```

```
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options
 TunnelOptions='[{TunnelInsideCidr=169.254.12.0/30,PreSharedKey=ExamplePreSharedKey1},
{TunnelInsideCidr=169.254.13.0/30,PreSharedKey=ExamplePreSharedKey2}]'
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.3",
                    "TunnelInsideCidr": "169.254.12.0/30",
                    "PreSharedKey": "ExamplePreSharedKey1"
                },
                {
                    "OutsideIpAddress": "203.0.113.5",
                    "TunnelInsideCidr": "169.254.13.0/30",
                    "PreSharedKey": "ExamplePreSharedKey2"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

示例 4：创建支持 IPv6 流量的 VPN 连接

以下 create-vpn-connection 示例将创建一个 VPN 连接，该连接支持指定中转网关和指定客户网关之间的 IPv6 流量。两个隧道的隧道选项均指定 Amazon 必须启动 IKE 协商。

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --transit-gateway-id tgw-12312312312312312 \
    --customer-gateway-id cgw-001122334455aabbc \
    --options TunnelInsideIpVersion=ipv6,TunnelOptions=[{StartupAction=start},
{StartupAction=start}]
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-11111111122222222",
        "TransitGatewayId": "tgw-12312312312312312",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv6NetworkCidr": "::/0",
            "RemoteIpv6NetworkCidr": "::/0",
            "TunnelInsideIpVersion": "ipv6",
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.3",
                    "StartupAction": "start"
                },
                {
                    "OutsideIpAddress": "203.0.113.5",
                    "StartupAction": "start"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpnConnection。

`create-vpn-gateway`

以下代码示例演示了如何使用 `create-vpn-gateway`。

Amazon CLI

创建虚拟私有网关

此示例创建虚拟私有网关。

命令:

```
aws ec2 create-vpn-gateway --type ipsec.1
```

输出：

```
{
    "VpnGateway": {
        "AmazonSideAsn": 64512,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

创建具有特定 Amazon 端 ASN 的虚拟私有网关

此示例创建一个虚拟私有网关，并为 BGP 会话的 Amazon 端指定自治系统编号（ASN）。

命令:

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

输出：

```
{
```

```
    "VpnGateway": {
        "AmazonSideAsn": 65001,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpnGateway。

delete-carrier-gateway

以下代码示例演示了如何使用 delete-carrier-gateway。

Amazon CLI

删除运营商网关

以下 delete-carrier-gateway 示例删除指定的运营商网关。

```
aws ec2 delete-carrier-gateway \
    --carrier-gateway-id cagw-0465cdEXAMPLE1111
```

输出：

```
{
    "CarrierGateway": {
        "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",
        "VpcId": "vpc-0c529aEXAMPLE1111",
        "State": "deleting",
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的运营商网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCarrierGateway。

delete-client-vpn-endpoint

以下代码示例演示了如何使用 delete-client-vpn-endpoint。

Amazon CLI

删除 Client VPN 端点

以下 `delete-client-vpn-endpoint` 示例删除指定的 Client VPN 端点。

```
aws ec2 delete-client-vpn-endpoint \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "Status": {
        "Code": "deleting"
    }
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的 Client VPN 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClientVpnEndpoint。

delete-client-vpn-route

以下代码示例演示了如何使用 `delete-client-vpn-route`。

Amazon CLI

删除 Client VPN 端点的路由

以下 `delete-client-vpn-route` 示例将删除 Client VPN 端点指定子网的 `0.0.0.0/0` 路由。

```
aws ec2 delete-client-vpn-route \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --destination-cidr-block 0.0.0.0/0 \
    --target-vpc-subnet-id subnet-0123456789abcabca
```

输出：

```
{
    "Status": {
        "Code": "deleting"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClientVpnRoute。

delete-coip-cidr

以下代码示例演示了如何使用 delete-coip-cidr。

Amazon CLI

删除客户拥有的 IP（CoIP）地址的范围

以下 delete-coip-cidr 示例将在指定的 CoIP 池中删除指定范围的 CoIP 地址。

```
aws ec2 delete-coip-cidr \
    --cidr 14.0.0.0/24 \
    --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

输出：

```
{
    "CoipCidr": {
        "Cidr": "14.0.0.0/24",
        "CoipPoolId": "ipv4pool-coip-1234567890abcdefg",
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCoipCidr。

delete-coip-pool

以下代码示例演示了如何使用 delete-coip-pool。

Amazon CLI

删除客户拥有的 IP（CoIP）地址池

以下 delete-coip-pool 示例将删除 CoIP 地址的 CoIP 池。

```
aws ec2 delete-coip-pool \
    --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

输出：

```
{
    "CoipPool": {
        "PoolId": "ipv4pool-coip-1234567890abcdefg",
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
        "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-
coip-1234567890abcdefg"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCoipPool。

delete-customer-gateway

以下代码示例演示了如何使用 delete-customer-gateway。

Amazon CLI

删除客户网关

此示例删除指定的客户网关。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCustomerGateway。

delete-dhcp-options

以下代码示例演示了如何使用 delete-dhcp-options。

Amazon CLI

删除 DHCP 选项集

此示例删除指定的 DHCP 选项集。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDhcpOptions。

delete-egress-only-internet-gateway

以下代码示例演示了如何使用 delete-egress-only-internet-gateway。

Amazon CLI

删除仅出口互联网网关

此示例删除指定的仅出口互联网网关。

命令:

```
aws ec2 delete-egress-only-internet-gateway --egress-only-internet-gateway-
id eigw-01eadbd45ecd7943f
```

输出：

```
{
  "ReturnCode": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEgressOnlyInternetGateway。

delete-fleets

以下代码示例演示了如何使用 delete-fleets。

Amazon CLI

示例 1：删除 EC2 实例集并终止关联的实例

以下 delete-fleets 示例删除指定的 EC2 实例集并终止关联的按需型实例和竞价型实例。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
    --terminate-instances
```

输出：

```
{
    "SuccessfulFleetDeletions": [
        {
            "CurrentFleetState": "deleted_terminating",
            "PreviousFleetState": "active",
            "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
        }
    ],
    "UnsuccessfulFleetDeletions": []
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的删除 EC2 实例集。

示例 2：删除 EC2 实例集而不终止关联的实例

以下 delete-fleets 示例删除指定的 EC2 实例集而不终止关联的按需型实例和竞价型实例。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
    --no-terminate-instances
```

输出：

```
{
    "SuccessfulFleetDeletions": [
        {
            "CurrentFleetState": "deleted_running",
            "PreviousFleetState": "active",
            "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
        }
    ],
    "UnsuccessfulFleetDeletions": []
```

```
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的删除 EC2 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFleets。

delete-flow-logs

以下代码示例演示了如何使用 delete-flow-logs。

Amazon CLI

删除流日志

以下 delete-flow-logs 示例删除指定流日志。

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

输出：

```
{
    "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFlowLogs。

delete-fpga-image

以下代码示例演示了如何使用 delete-fpga-image。

Amazon CLI

删除 Amazon FPGA 映像

此示例将删除指定 AFI。

命令:

```
aws ec2 delete-fpga-image --fpga-image-id afi-06b12350a123fbabc
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFpgaImage。

delete-instance-connect-endpoint

以下代码示例演示了如何使用 delete-instance-connect-endpoint。

Amazon CLI

删除 EC2 实例连接端点

以下 delete-instance-connect-endpoint 示例删除指定的 EC2 实例连接端点。

```
aws ec2 delete-instance-connect-endpoint \
    --instance-connect-endpoint-id eice-03f5e49b83924bbc7
```

输出：

```
{
    "InstanceConnectEndpoint": {
        "OwnerId": "111111111111",
        "InstanceConnectEndpointId": "eice-0123456789example",
        "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
        "State": "delete-in-progress",
        "StateMessage": "",
        "NetworkInterfaceIds": [],
        "VpcId": "vpc-0123abcd",
        "AvailabilityZone": "us-east-1d",
        "CreatedAt": "2023-02-07T12:05:37+00:00",
        "SubnetId": "subnet-0123abcd"
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的删除 EC2 实例连接端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInstanceConnectEndpoint。

delete-instance-event-window

以下代码示例演示了如何使用 delete-instance-event-window。

Amazon CLI

示例 1：删除事件窗口

以下 delete-instance-event-window 示例将删除事件窗口。

```
aws ec2 delete-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890
```

输出：

```
{
    "InstanceEventWindowState": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "State": "deleting"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 2：强制删除事件窗口

如果事件窗口当前与目标相关联，则以下 delete-instance-event-window 示例将强制删除该事件窗口。

```
aws ec2 delete-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --force-delete
```

输出：

```
{
    "InstanceEventWindowState": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "State": "deleting"
```

```
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInstanceEventWindow。

`delete-internet-gateway`

以下代码示例演示了如何使用 `delete-internet-gateway`。

Amazon CLI

删除互联网网关

以下 `delete-internet-gateway` 示例删除指定的互联网网关。

```
aws ec2 delete-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC 用户指南》中的互联网网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInternetGateway。

`delete-ipam-pool`

以下代码示例演示了如何使用 `delete-ipam-pool`。

Amazon CLI

删除 IPAM 池

在此示例中，您是 IPAM 委派管理员，想要删除不再需要的 IPAM 池，但该池已预置 CIDR。如果池已预置 CIDR，则除非您使用 `--cascade` 选项，否则无法删除该池，因此您将使用 `--cascade`。

要完成此请求，请执行以下操作：

您需要 IPAM 池 ID，您可以通过 describe-ipam-pools 获取该 ID。`--region` 必须是 IPAM 主区域。

以下 delete-ipam-pool 示例将删除 Amazon 账户中的 IPAM 池。

```
aws ec2 delete-ipam-pool \
    --ipam-pool-id ipam-pool-050c886a3ca41cd5b \
    --cascade \
    --region us-east-1
```

输出：

```
{
    "IpamPool": {
        "OwnerId": "320805250157",
        "IpamPoolId": "ipam-pool-050c886a3ca41cd5b",
        "IpamPoolArn": "arn:aws:ec2::320805250157:ipam-pool/ipam-
pool-050c886a3ca41cd5b",
        "IpamScopeArn": "arn:aws:ec2::320805250157:ipam-scope/ipam-
scope-0a158dde35c51107b",
        "IpamScopeType": "private",
        "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
        "IpamRegion": "us-east-1",
        "Locale": "None",
        "PoolDepth": 1,
        "State": "delete-in-progress",
        "Description": "example",
        "AutoImport": false,
        "AddressFamily": "ipv4",
        "AllocationMinNetmaskLength": 0,
        "AllocationMaxNetmaskLength": 32
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的删除池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIpamPool。

delete-ipam-resource-discovery

以下代码示例演示了如何使用 delete-ipam-resource-discovery。

Amazon CLI

删除资源发现

在此示例中，您是 IPAM 委派管理员，想要删除您在将 IPAM 与组织外部账户集成的过程中为与其他 IPAM 管理员共享而创建的非默认资源发现。

要完成此请求，请执行以下操作：

--region 必须是您在其中创建资源发现的区域。如果是 "IsDefault": true，则无法删除默认资源发现。默认资源发现是在创建 IPAM 的账户中自动创建的资源发现。要删除默认资源发现，必须删除 IPAM。

以下 delete-ipam-resource-discovery 示例将删除资源发现。

```
aws ec2 delete-ipam-resource-discovery \
    --ipam-resource-discovery-id ipam-res-disco-0e39761475298ee0f \
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscovery": {
        "OwnerId": "149977607591",
        "IpamResourceDiscoveryId": "ipam-res-disco-0e39761475298ee0f",
        "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0e39761475298ee0f",
        "IpamResourceDiscoveryRegion": "us-east-1",
        "OperatingRegions": [
            {
                "RegionName": "us-east-1"
            }
        ],
        "IsDefault": false,
        "State": "delete-in-progress"
    }
}
```

有关资源发现的更多信息，请参阅《Amazon VPC IPAM 用户指南》中的使用资源发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIpamResourceDiscovery。

delete-ipam-scope

以下代码示例演示了如何使用 delete-ipam-scope。

Amazon CLI

删除 IPAM 范围

以下 delete-ipam-scope 示例将删除 IPAM。

```
aws ec2 delete-ipam-scope \
    --ipam-scope-id ipam-scope-01c1ebab2b63bd7e4
```

输出：

```
{
    "IpamScope": {
        "OwnerId": "123456789012",
        "IpamScopeId": "ipam-scope-01c1ebab2b63bd7e4",
        "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-01c1ebab2b63bd7e4",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
        "IpamRegion": "us-east-1",
        "IpamScopeType": "private",
        "IsDefault": false,
        "Description": "Example description",
        "PoolCount": 0,
        "State": "delete-in-progress"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的删除范围。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIpamScope。

delete-ipam

以下代码示例演示了如何使用 delete-ipam。

Amazon CLI

删除 IPAM

以下 delete-ipam 示例将删除 IPAM。

```
aws ec2 delete-ipam \
```

```
    --ipam-id ipam-036486dfa6af58ee0
```

输出：

```
{
    "Ipam": {
        "OwnerId": "123456789012",
        "IpamId": "ipam-036486dfa6af58ee0",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-036486dfa6af58ee0",
        "IpamRegion": "us-east-1",
        "PublicDefaultScopeId": "ipam-scope-071b8042b0195c183",
        "PrivateDefaultScopeId": "ipam-scope-0807405dece705a30",
        "ScopeCount": 2,
        "OperatingRegions": [
            {
                "RegionName": "us-east-1"
            },
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-west-1"
            }
        ],
        "State": "delete-in-progress"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的删除 IPAM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIpam。

delete-key-pair

以下代码示例演示了如何使用 delete-key-pair。

Amazon CLI

删除密钥对

以下 delete-key-pair 示例删除指定密钥对。

```
aws ec2 delete-key-pair \
```

```
    --key-name my-key-pair
```

输出：

```
{
    "Return": true,
    "KeyPairId": "key-03c8d3aceb53b507"
}
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的创建和删除密钥对。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteKeyPair。

delete-launch-template-versions

以下代码示例演示了如何使用 delete-launch-template-versions。

Amazon CLI

删除启动模板版本

此示例删除指定的启动模板版本。

命令:

```
aws ec2 delete-launch-template-versions --launch-template-id lt-0abcd290751193123 --
versions 1
```

输出：

```
{
  "UnsuccessfullyDeletedLaunchTemplateVersions": [],
  "SuccessfullyDeletedLaunchTemplateVersions": [
      {
          "LaunchTemplateName": "TestVersion",
          "VersionNumber": 1,
          "LaunchTemplateId": "lt-0abcd290751193123"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLaunchTemplateVersions。

delete-launch-template

以下代码示例演示了如何使用 delete-launch-template。

Amazon CLI

删除启动模板

此示例删除指定的启动模板。

命令:

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

输出：

```
{
  "LaunchTemplate": {
      "LatestVersionNumber": 2,
      "LaunchTemplateId": "lt-0abcd290751193123",
      "LaunchTemplateName": "TestTemplate",
      "DefaultVersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-23T16:46:25.000Z"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLaunchTemplate。

delete-local-gateway-route-table-virtual-interface-group-association

以下代码示例演示了如何使用 delete-local-gateway-route-table-virtual-interface-group-association。

Amazon CLI

取消本地网关路由表与虚拟接口（VIF）组的关联

以下 delete-local-gateway-route-table-virtual-interface-group-association 示例删除指定的本地网关路由表与 VIF 组之间的关联。

```
aws ec2 delete-local-gateway-route-table-virtual-interface-group-association \
```

```
    --local-gateway-route-table-virtual-interface-group-association-id lgw-vif-grp-assoc-exampleid12345678
```

输出：

```
{
    "LocalGatewayRouteTableVirtualInterfaceGroupAssociation": {
        "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-grp-assoc-exampleid12345678",
        "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-exampleid0123abcd",
        "LocalGatewayId": "lgw-exampleid11223344",
        "LocalGatewayRouteTableId": "lgw-rtb-exampleidabcd1234",
        "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-gateway-route-table/lgw-rtb-exampleidabcd1234",
        "OwnerId": "111122223333",
        "State": "disassociating",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的 VIF 组关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation。

delete-local-gateway-route-table-vpc-association

以下代码示例演示了如何使用 delete-local-gateway-route-table-vpc-association。

Amazon CLI

取消本地网关路由表与 VPC 的关联

以下 delete-local-gateway-route-table-vpc-association 示例删除指定的本地网关路由表与 VPC 之间的关联。

```
aws ec2 delete-local-gateway-route-table-vpc-association \
    --local-gateway-route-table-vpc-association-id vpc-example0123456789
```

输出：

```
{
```

```
    "LocalGatewayRouteTableVpcAssociation": {
        "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-abcd1234wxyz56789",
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
        "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:555555555555:local-
gateway-route-table/lgw-rtb-abcdefg1234567890",
        "LocalGatewayId": "lgw-exampleid01234567",
        "VpcId": "vpc-example0123456789",
        "OwnerId": "555555555555",
        "State": "disassociating"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLocalGatewayRouteTableVpcAssociation。

delete-local-gateway-route-table

以下代码示例演示了如何使用 delete-local-gateway-route-table。

Amazon CLI

删除本地网关路由表

以下 delete-local-gateway-route-table 示例使用直接 VPC 路由模式创建本地网关路由表。

```
aws ec2 delete-local-gateway-route-table \
    --local-gateway-route-table-id lgw-rtb-abcdefg1234567890
```

输出：

```
{
    "LocalGatewayRouteTable": {
        "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
        "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-
gateway-route-table/lgw-rtb-abcdefg1234567890",
        "LocalGatewayId": "lgw-1a2b3c4d5e6f7g8h9",
        "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/
op-021345abcdef67890",
        "OwnerId": "111122223333",
```

```
        "State": "deleting",
        "Tags": [],
        "Mode": "direct-vpc-routing"
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的本地网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLocalGatewayRouteTable。

delete-local-gateway-route

以下代码示例演示了如何使用 delete-local-gateway-route。

Amazon CLI

从本地网关路由表中删除路由

以下 delete-local-gateway-route 示例从指定的本地网关路由表中删除指定的路由。

```
aws ec2 delete-local-gateway-route \
    --destination-cidr-block 0.0.0.0/0 \
    --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "0.0.0.0/0",
        "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
        "Type": "static",
        "State": "deleted",
        "LocalGatewayRouteTableId": "lgw-rtb-059615ef7EXAMPLE"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLocalGatewayRoute。

delete-managed-prefix-list

以下代码示例演示了如何使用 delete-managed-prefix-list。

Amazon CLI

删除前缀列表

以下 delete-managed-prefix-list 示例删除指定的前缀列表。

```
aws ec2 delete-managed-prefix-list \
    --prefix-list-id pl-0123456abcabcabc1
```

输出：

```
{
    "PrefixList": {
        "PrefixListId": "pl-0123456abcabcabc1",
        "AddressFamily": "IPv4",
        "State": "delete-in-progress",
        "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
        "PrefixListName": "test",
        "MaxEntries": 10,
        "Version": 1,
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteManagedPrefixList。

delete-nat-gateway

以下代码示例演示了如何使用 delete-nat-gateway。

Amazon CLI

删除 NAT 网关

此示例删除 NAT 网关 nat-04ae55e711cec5680。

命令:

```
aws ec2 delete-nat-gateway --nat-gateway-id nat-04ae55e711cec5680
```

输出：

```
{
    "NatGatewayId": "nat-04ae55e711cec5680"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNatGateway。

delete-network-acl-entry

以下代码示例演示了如何使用 delete-network-acl-entry。

Amazon CLI

删除网络 ACL 条目

此示例从指定的网络 ACL 中删除编号为 100 的入口规则。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkAclEntry。

delete-network-acl

以下代码示例演示了如何使用 delete-network-acl。

Amazon CLI

删除网络 ACL

此示例删除指定的网络 ACL。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkAcl。

delete-network-insights-access-scope-analysis

以下代码示例演示了如何使用 delete-network-insights-access-scope-analysis。

Amazon CLI

删除网络访问范围分析

以下 delete-network-insights-access-scope-analysis 示例删除指定的网络访问范围分析。

```
aws ec2 delete-network-insights-access-scope-analysis \
    --network-insights-access-scope-analysis-id nisa-01234567891abcdef
```

输出：

```
{
    "NetworkInsightsAccessScopeAnalysisId": "nisa-01234567891abcdef
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInsightsAccessScopeAnalysis。

delete-network-insights-access-scope

以下代码示例演示了如何使用 delete-network-insights-access-scope。

Amazon CLI

删除网络访问范围

以下 delete-network-insights-access-scope 示例删除指定的网络访问范围。

```
aws ec2 delete-network-insights-access-scope \
    --network-insights-access-scope-id nis-123456789abc01234
```

输出：

```
{
```

```
    "NetworkInsightsAccessScopeId": "nis-123456789abc01234"
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInsightsAccessScope。

delete-network-insights-analysis

以下代码示例演示了如何使用 delete-network-insights-analysis。

Amazon CLI

删除路径分析

以下 delete-network-insights-analysis 示例删除指定的分析。

```
aws ec2 delete-network-insights-analysis \
    --network-insights-analysis-id nia-02207aa13eb480c7a
```

输出：

```
{
    "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a"
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInsightsAnalysis。

delete-network-insights-path

以下代码示例演示了如何使用 delete-network-insights-path。

Amazon CLI

删除路径

以下 delete-network-insights-path 示例删除指定的路径。在删除路径之前，必须先使用 delete-network-insights-analysis 命令删除其所有分析。

```
aws ec2 delete-network-insights-path \
    --network-insights-path-id nip-0b26f224f1d131fa8
```

输出：

```
{
    "NetworkInsightsPathId": "nip-0b26f224f1d131fa8"
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInsightsPath。

delete-network-interface-permission

以下代码示例演示了如何使用 delete-network-interface-permission。

Amazon CLI

删除网络接口权限

此示例删除指定的网络接口权限。

命令:

```
aws ec2 delete-network-interface-permission --network-interface-permission-id eni-perm-06fd19020ede149ea
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInterfacePermission。

delete-network-interface

以下代码示例演示了如何使用 delete-network-interface。

Amazon CLI

删除网络接口

此示例删除指定的网络接口。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNetworkInterface。

delete-placement-group

以下代码示例演示了如何使用 delete-placement-group。

Amazon CLI

删除置放群组

此示例命令将删除指定的置放群组。

命令:

```
aws ec2 delete-placement-group --group-name my-cluster
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePlacementGroup。

delete-queued-reserved-instances

以下代码示例演示了如何使用 delete-queued-reserved-instances。

Amazon CLI

删除排队的购买

以下 delete-queued-reserved-instances 示例删除已排队等候购买的指定预留实例。

```
aws ec2 delete-queued-reserved-instances \
    --reserved-instances-ids af9f760e-6f91-4559-85f7-4980eexample
```

输出：

```
{
    "SuccessfulQueuedPurchaseDeletions": [
        {
            "ReservedInstancesId": "af9f760e-6f91-4559-85f7-4980eexample"
        }
    ],
    "FailedQueuedPurchaseDeletions": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteQueuedReservedInstances。

delete-route-table

以下代码示例演示了如何使用 delete-route-table。

Amazon CLI

删除路由表

此示例删除指定的路由表。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRouteTable。

delete-route

以下代码示例演示了如何使用 delete-route。

Amazon CLI

删除路由

此示例从指定的路由表中删除指定的路由。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-
block 0.0.0.0/0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoute。

delete-security-group

以下代码示例演示了如何使用 delete-security-group。

Amazon CLI

[EC2-Classic] 删除安全组

此示例删除名为 MySecurityGroup 的安全组。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

[EC2-VPC] 删除安全组

本示例将删除 ID 为 sg-903004f8 的安全组。请注意，您不能通过名称引用 EC2-VPC 的安全组。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-security-group --group-id sg-903004f8
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的“使用安全组”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSecurityGroup。

delete-snapshot

以下代码示例演示了如何使用 delete-snapshot。

Amazon CLI

删除快照

本示例命令将删除快照 ID 为 snap-1234567890abcdef0 的快照。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSnapshot。

delete-spot-datafeed-subscription

以下代码示例演示了如何使用 delete-spot-datafeed-subscription。

Amazon CLI

取消竞价型实例数据源订阅

此示例命令将删除账户的竞价型数据源订阅。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-spot-datafeed-subscription
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSpotDatafeedSubscription。

delete-subnet-cidr-reservation

以下代码示例演示了如何使用 delete-subnet-cidr-reservation。

Amazon CLI

删除子网 CIDR 预留

以下 delete-subnet-cidr-reservation 示例删除指定的子网 CIDR 预留。

```
aws ec2 delete-subnet-cidr-reservation \
    --subnet-cidr-reservation-id scr-044f977c4eEXAMPLE
```

输出：

```
{
    "DeletedSubnetCidrReservation": {
        "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",
        "SubnetId": "subnet-03c51e2e6cEXAMPLE",
        "Cidr": "10.1.0.16/28",
        "ReservationType": "prefix",
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的子网 CIDR 预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubnetCidrReservation。

delete-subnet

以下代码示例演示了如何使用 delete-subnet。

Amazon CLI

删除子网

此示例删除指定的子网。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubnet。

delete-tags

以下代码示例演示了如何使用 delete-tags。

Amazon CLI

示例 1：从资源中删除标签

以下 delete-tags 示例从指定映像中删除标签 Stack=Test。当您同时指定值和键名称时，只有当标签的值与指定值匹配时，才会删除该标签。

```
aws ec2 delete-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=Test
```

可以选择为标签指定值。以下 delete-tags 示例从指定实例中删除键名称为 purpose 的标签（无论该标签的标签值如何）。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=purpose
```

如果您将空字符串指定为标签值，则仅当标签的值为空字符串时，才会删除标签。以下 delete-tags 示例将空字符串指定为要删除的标签的标签值。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=Name,Value=
```

示例 2：从多个资源中删除标签

以下 delete-tags 示例从实例和 AMI 中删除标签 ``Purpose=Test``。如上一个示例所示，您可以省略命令中的标签值。

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 ami-1234567890abcdef0 \
    --tags Key=Purpose
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTags。

delete-traffic-mirror-filter-rule

以下代码示例演示了如何使用 delete-traffic-mirror-filter-rule。

Amazon CLI

删除流量镜像筛选条件规则

以下 delete-traffic-mirror-filter-rule 示例删除指定的流量镜像筛选条件规则。

```
aws ec2 delete-traffic-mirror-filter-rule \
    --traffic-mirror-filter-rule-id tmfr-081f71283bEXAMPLE
```

输出：

```
{
    "TrafficMirrorFilterRuleId": "tmfr-081f71283bEXAMPLE"
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的修改流量镜像筛选条件规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTrafficMirrorFilterRule。

delete-traffic-mirror-filter

以下代码示例演示了如何使用 delete-traffic-mirror-filter。

Amazon CLI

删除流量镜像筛选条件

以下 delete-traffic-mirror-filter 示例删除指定的流量镜像筛选条件。

```
aws ec2 delete-traffic-mirror-filter \
    --traffic-mirror-filter-id tmf-0be0b25fcdEXAMPLE
```

输出：

```
{
    "TrafficMirrorFilterId": "tmf-0be0b25fcdEXAMPLE"
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的删除流量镜像筛选条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTrafficMirrorFilter。

delete-traffic-mirror-session

以下代码示例演示了如何使用 delete-traffic-mirror-session。

Amazon CLI

删除流量镜像会话

以下 delete-traffic-mirror-session 示例删除指定的流量镜像会话。

```
aws ec2 delete-traffic-mirror-session \
    --traffic-mirror-session-id tms-0af3141ce5EXAMPLE
```

输出：

```
{
    "TrafficMirrorSessionId": "tms-0af3141ce5EXAMPLE"
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的删除流量镜像会话。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTrafficMirrorSession。

delete-traffic-mirror-target

以下代码示例演示了如何使用 delete-traffic-mirror-target。

Amazon CLI

删除流量镜像目标

以下 delete-traffic-mirror-target 示例删除指定的流量镜像目标。

```
aws ec2 delete-traffic-mirror-target \
    --traffic-mirror-target-id tmt-060f48ce9EXAMPLE
```

输出：

```
{
    "TrafficMirrorTargetId": "tmt-060f48ce9EXAMPLE"
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的删除流量镜像目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTrafficMirrorTarget。

delete-transit-gateway-connect-peer

以下代码示例演示了如何使用 delete-transit-gateway-connect-peer。

Amazon CLI

删除 Transit Gateway Connect 对等

以下 delete-transit-gateway-connect-peer 示例删除指定的 Connect 对等。

```
aws ec2 delete-transit-gateway-connect-peer \
    --transit-gateway-connect-peer-id tgw-connect-peer-0666adbac4EXAMPLE
```

输出：

```
{
    "TransitGatewayConnectPeer": {
        "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",
        "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",
        "State": "deleting",
        "CreationTime": "2021-10-13T03:35:17.000Z",
        "ConnectPeerConfiguration": {
            "TransitGatewayAddress": "10.0.0.234",
            "PeerAddress": "172.31.1.11",
            "InsideCidrBlocks": [
                "169.254.6.0/29"
            ],
            "Protocol": "gre",
            "BgpConfigurations": [
                {
                    "TransitGatewayAsn": 64512,
                    "PeerAsn": 64512,
                    "TransitGatewayAddress": "169.254.6.2",
                    "PeerAddress": "169.254.6.1",
                    "BgpStatus": "down"
                },
                {
                    "TransitGatewayAsn": 64512,
                    "PeerAsn": 64512,
                    "TransitGatewayAddress": "169.254.6.3",
```

```
                    "PeerAddress": "169.254.6.1",
                    "BgpStatus": "down"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayConnectPeer。

`delete-transit-gateway-connect`

以下代码示例演示了如何使用 `delete-transit-gateway-connect`。

Amazon CLI

删除 Transit Gateway Connect 连接

以下 `delete-transit-gateway-connect` 示例删除指定的 Connect 连接。

```
aws ec2 delete-transit-gateway-connect \
    --transit-gateway-attachment-id tgw-attach-037012e5dcEXAMPLE
```

输出：

```
{
    "TransitGatewayConnect": {
        "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",
        "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
        "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
        "State": "deleting",
        "CreationTime": "2021-03-09T19:59:17+00:00",
        "Options": {
            "Protocol": "gre"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayConnect。

delete-transit-gateway-multicast-domain

以下代码示例演示了如何使用 delete-transit-gateway-multicast-domain。

Amazon CLI

删除中转网关组播域

以下 delete-transit-gateway-multicast-domain 示例删除指定的组播域。

```
aws ec2 delete-transit-gateway-multicast-domain \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

输出：

```
{
    "TransitGatewayMulticastDomain": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-02bb79002bEXAMPLE",
        "TransitGatewayId": "tgw-0d88d2d0d5EXAMPLE",
        "State": "deleting",
        "CreationTime": "2019-11-20T22:02:03.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的管理组播域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayMulticastDomain。

delete-transit-gateway-peering-attachment

以下代码示例演示了如何使用 delete-transit-gateway-peering-attachment。

Amazon CLI

删除中转网关对等连接

以下 delete-transit-gateway-peering-attachment 示例删除指定的中转网关对等连接。

```
aws ec2 delete-transit-gateway-peering-attachment \
    --transit-gateway-attachment-id tgw-attach-4455667788aabbccd
```

输出：

```
{
    "TransitGatewayPeeringAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
        "RequesterTgwInfo": {
            "TransitGatewayId": "tgw-123abc05e04123abc",
            "OwnerId": "123456789012",
            "Region": "us-west-2"
        },
        "AccepterTgwInfo": {
            "TransitGatewayId": "tgw-11223344aabbcc112",
            "OwnerId": "123456789012",
            "Region": "us-east-2"
        },
        "State": "deleting",
        "CreationTime": "2019-12-09T11:38:31.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关对等连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayPeeringAttachment。

delete-transit-gateway-policy-table

以下代码示例演示了如何使用 delete-transit-gateway-policy-table。

Amazon CLI

删除中转网关策略表

以下 delete-transit-gateway-policy-table 示例删除指定的中转网关策略表。

```
aws ec2 delete-transit-gateway-policy-table \
```

```
    --transit-gateway-policy-table-id tgw-ptb-0a16f134b78668a81
```

输出：

```
{
    "TransitGatewayPolicyTables": [
        {
            "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
            "TransitGatewayId": "tgw-067f8505c18f0bd6e",
            "State": "deleting",
            "CreationTime": "2023-11-28T16:36:43+00:00",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关用户指南》中的中转网关策略表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayPolicyTable。

delete-transit-gateway-prefix-list-reference

以下代码示例演示了如何使用 delete-transit-gateway-prefix-list-reference。

Amazon CLI

删除前缀列表引用

以下 delete-transit-gateway-prefix-list-reference 示例删除指定的前缀列表引用。

```
aws ec2 delete-transit-gateway-prefix-list-reference \
    --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
    --prefix-list-id pl-11111122222222333
```

输出：

```
{
    "TransitGatewayPrefixListReference": {
        "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
        "PrefixListId": "pl-11111122222222333",
        "PrefixListOwnerId": "123456789012",
        "State": "deleting",
```

```
        "Blackhole": false,
        "TransitGatewayAttachment": {
            "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",
            "ResourceType": "vpc",
            "ResourceId": "vpc-112233445566aabbc"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的前缀列表引用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayPrefixListReference。

delete-transit-gateway-route-table

以下代码示例演示了如何使用 delete-transit-gateway-route-table。

Amazon CLI

删除中转网关路由表

以下 delete-transit-gateway-route-table 示例删除指定的中转网关路由表。

```
aws ec2  delete-transit-gateway-route-table \
    --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE
```

输出：

```
{
    "TransitGatewayRouteTable": {
        "TransitGatewayRouteTableId": "tgw-rtb-0b6f6aaa01EXAMPLE",
        "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
        "State": "deleting",
        "DefaultAssociationRouteTable": false,
        "DefaultPropagationRouteTable": false,
        "CreationTime": "2019-07-17T20:27:26.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的删除中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayRouteTable。

delete-transit-gateway-route

以下代码示例演示了如何使用 delete-transit-gateway-route。

Amazon CLI

从路由表中删除 CIDR 块

以下 delete-transit-gateway-route 示例从指定的中转网关路由表中删除 CIDR 块。

```
aws ec2 delete-transit-gateway-route \
    --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE \
    --destination-cidr-block 10.0.2.0/24
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "10.0.2.0/24",
        "TransitGatewayAttachments": [
            {
                "ResourceId": "vpc-0065acced4EXAMPLE",
                "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
                "ResourceType": "vpc"
            }
        ],
        "Type": "static",
        "State": "deleted"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的删除静态路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayRoute。

delete-transit-gateway-vpc-attachment

以下代码示例演示了如何使用 delete-transit-gateway-vpc-attachment。

Amazon CLI

删除中转网关 VPC 连接

以下 delete-transit-gateway-vpc-attachment 示例删除指定的 VPC 连接。

```
aws ec2 delete-transit-gateway-vpc-attachment \
    --transit-gateway-attachment-id tgw-attach-0d2c54bdbEXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0d2c54bdb3EXAMPLE",
        "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
        "VpcId": "vpc-0065acced4f61c651",
        "VpcOwnerId": "111122223333",
        "State": "deleting",
        "CreationTime": "2019-07-17T16:04:27.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的删除 VPC 连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGatewayVpcAttachment。

delete-transit-gateway

以下代码示例演示了如何使用 delete-transit-gateway。

Amazon CLI

删除中转网关

以下 delete-transit-gateway 示例删除指定的中转网关。

```
aws ec2  delete-transit-gateway \
    --transit-gateway-id tgw-01f04542b2EXAMPLE
```

输出：

```
{
    "TransitGateway": {
```

```
        "TransitGatewayId": "tgw-01f04542b2EXAMPLE",
        "State": "deleting",
        "OwnerId": "123456789012",
        "Description": "Example Transit Gateway",
        "CreationTime": "2019-08-27T15:04:35.000Z",
        "Options": {
            "AmazonSideAsn": 64515,
            "AutoAcceptSharedAttachments": "disable",
            "DefaultRouteTableAssociation": "enable",
            "AssociationDefaultRouteTableId": "tgw-rtb-0ce7a6948fEXAMPLE",
            "DefaultRouteTablePropagation": "enable",
            "PropagationDefaultRouteTableId": "tgw-rtb-0ce7a6948fEXAMPLE",
            "VpnEcmpSupport": "enable",
            "DnsSupport": "enable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的删除中转网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTransitGateway。

delete-verified-access-endpoint

以下代码示例演示了如何使用 delete-verified-access-endpoint。

Amazon CLI

删除已验证访问端点

以下 delete-verified-access-endpoint 示例删除指定的已验证访问端点。

```
aws ec2 delete-verified-access-endpoint \
    --verified-access-endpoint-id vae-066fac616d4d546f2
```

输出：

```
{
    "VerifiedAccessEndpoint": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
        "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
```

```
        "ApplicationDomain": "example.com",
        "EndpointType": "network-interface",
        "AttachmentType": "vpc",
        "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
        "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
        "SecurityGroupIds": [
            "sg-004915970c4c8f13a"
        ],
        "NetworkInterfaceOptions": {
            "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
            "Protocol": "https",
            "Port": 443
        },
        "Status": {
            "Code": "deleting"
        },
        "Description": "Testing Verified Access",
        "CreationTime": "2023-08-25T20:54:43",
        "LastUpdatedTime": "2023-08-25T22:46:32"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVerifiedAccessEndpoint。

delete-verified-access-group

以下代码示例演示了如何使用 delete-verified-access-group。

Amazon CLI

删除已验证访问组

以下 delete-verified-access-group 示例删除指定的已验证访问组。

```
aws ec2 delete-verified-access-group \
    --verified-access-group-id vagr-0dbe967baf14b7235
```

输出：

```
{
    "VerifiedAccessGroup": {
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "Testing Verified Access",
        "Owner": "123456789012",
        "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
        "CreationTime": "2023-08-25T19:55:19",
        "LastUpdatedTime": "2023-08-25T22:49:03",
        "DeletionTime": "2023-08-26T00:58:31"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVerifiedAccessGroup。

delete-verified-access-instance

以下代码示例演示了如何使用 delete-verified-access-instance。

Amazon CLI

删除已验证访问实例

以下 delete-verified-access-instance 示例删除指定的已验证访问实例。

```
aws ec2 delete-verified-access-instance \
    --verified-access-instance-id vai-0ce000c0b7643abea
```

输出：

```
{
    "VerifiedAccessInstance": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "Testing Verified Access",
        "VerifiedAccessTrustProviders": [],
        "CreationTime": "2023-08-25T18:27:56",
        "LastUpdatedTime": "2023-08-26T01:00:18"
    }
```

```
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVerifiedAccessInstance。

delete-verified-access-trust-provider

以下代码示例演示了如何使用 delete-verified-access-trust-provider。

Amazon CLI

删除已验证访问信任提供商

以下 delete-verified-access-trust-provider 示例删除指定的已验证访问信任提供商。

```
aws ec2 delete-verified-access-trust-provider \
    --verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

输出：

```
{
    "VerifiedAccessTrustProvider": {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "Description": "Testing Verified Access",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center",
        "PolicyReferenceName": "idc",
        "CreationTime": "2023-08-25T18:40:36",
        "LastUpdatedTime": "2023-08-25T18:40:36"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问的信任提供商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVerifiedAccessTrustProvider。

delete-volume

以下代码示例演示了如何使用 delete-volume。

Amazon CLI

删除卷

此示例命令将删除卷 ID 为 vol-049df61146c4d7901 的可用卷。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVolume。

delete-vpc-endpoint-connection-notifications

以下代码示例演示了如何使用 delete-vpc-endpoint-connection-notifications。

Amazon CLI

删除端点连接通知

此示例删除指定的端点连接通知。

命令:

```
aws ec2 delete-vpc-endpoint-connection-notifications --connection-notification-
ids vpce-nfn-008776de7e03f5abc
```

输出：

```
{
  "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpcEndpointConnectionNotifications。

delete-vpc-endpoint-service-configurations

以下代码示例演示了如何使用 delete-vpc-endpoint-service-configurations。

Amazon CLI

删除端点服务配置

此示例删除指定的端点服务配置。

命令:

```
aws ec2 delete-vpc-endpoint-service-configurations --service-ids vpce-svc-03d5ebb7d9579a2b3
```

输出：

```
{
    "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpcEndpointServiceConfigurations。

delete-vpc-endpoints

以下代码示例演示了如何使用 delete-vpc-endpoints。

Amazon CLI

删除端点

此示例删除端点 vpce-aa22bb33 和 vpce-1a2b3c4d。如果命令部分成功或不成功，则会返回失败项目的列表。如果命令成功，则返回的列表为空。

命令:

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-aa22bb33 vpce-1a2b3c4d
```

输出：

```
{
  "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpcEndpoints。

delete-vpc-peering-connection

以下代码示例演示了如何使用 delete-vpc-peering-connection。

Amazon CLI

删除 VPC 对等连接

此示例删除指定的 VPC 对等连接。

命令:

```
aws ec2 delete-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

输出：

```
{
    "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpcPeeringConnection。

delete-vpc

以下代码示例演示了如何使用 delete-vpc。

Amazon CLI

删除 VPC

此示例删除指定的 VPC。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpc。

delete-vpn-connection-route

以下代码示例演示了如何使用 delete-vpn-connection-route。

Amazon CLI

从 VPN 连接中删除静态路由

此示例从指定的 VPN 连接中删除指定的静态路由。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --destination-cidr-block 11.12.0.0/16
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpnConnectionRoute。

delete-vpn-connection

以下代码示例演示了如何使用 delete-vpn-connection。

Amazon CLI

删除 VPN 连接

此示例删除指定的 VPN 连接。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpnConnection。

delete-vpn-gateway

以下代码示例演示了如何使用 delete-vpn-gateway。

Amazon CLI

删除虚拟私有网关

此示例删除指定的虚拟私有网关。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVpnGateway。

deprovision-byoip-cidr

以下代码示例演示了如何使用 deprovision-byoip-cidr。

Amazon CLI

删除 IP 地址范围以使其不再使用

以下示例删除指定的地址范围，使其不再用于 Amazon。

```
aws ec2 deprovision-byoip-cidr \
    --cidr 203.0.113.25/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "203.0.113.25/24",
        "State": "pending-deprovision"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprovisionByoipCidr。

deprovision-ipam-pool-cidr

以下代码示例演示了如何使用 deprovision-ipam-pool-cidr。

Amazon CLI

取消预置 IPAM 池 CIDR

以下示例 deprovision-ipam-pool-cidr 取消预置已配置到 IPAM 池的 CIDR。

（Linux）：

```
aws ec2 deprovision-ipam-pool-cidr \
    --ipam-pool-id ipam-pool-02ec043a19bbe5d08 \
    --cidr 11.0.0.0/16
```

（Windows）：

```
aws ec2 deprovision-ipam-pool-cidr ^
    --ipam-pool-id ipam-pool-02ec043a19bbe5d08 ^
    --cidr 11.0.0.0/16
```

输出：

```
{
    "IpamPoolCidr": {
        "Cidr": "11.0.0.0/16",
        "State": "pending-deprovision"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的取消预置池 CIDR。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprovisionIpamPoolCidr。

deregister-image

以下代码示例演示了如何使用 deregister-image。

Amazon CLI

取消注册 AMI

此示例取消注册指定的 AMI。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterImage。

deregister-instance-event-notification-attributes

以下代码示例演示了如何使用 deregister-instance-event-notification-attributes。

Amazon CLI

示例 1：从事件通知中删除所有标签

以下 deregister-instance-event-notification-attributes 示例删除 IncludeAllTagsOfInstance=true，其效果是将 IncludeAllTagsOfInstance 设置为 false。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute IncludeAllTagsOfInstance=true
```

输出：

```
{
    "InstanceTagAttribute": {
        "InstanceTagKeys": [],
        "IncludeAllTagsOfInstance": true
    }
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的实例的计划事件。

示例 2：从事件通知中删除特定标签

以下 deregister-instance-event-notification-attributes 示例从事件通知中包含的标签中删除指定标签。要描述事件通知中包含的剩余标签，请使用 describe-instance-event-notification-attributes。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute InstanceTagKeys="tag-key2"
```

输出：

```
{
    "InstanceTagAttribute": {
        "InstanceTagKeys": [
            "tag-key2"
```

```
        ],
        "IncludeAllTagsOfInstance": false
    }
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的实例的计划事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterInstanceEventNotificationAttributes。

deregister-transit-gateway-multicast-group-members

以下代码示例演示了如何使用 deregister-transit-gateway-multicast-group-members。

Amazon CLI

从多播组中取消注册组成员

此示例从中转网关多播组中取消注册指定的网络接口组成员。

```
aws ec2 deregister-transit-gateway-multicast-group-members \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE \
    --group-ip-address 224.0.1.0 \
    --network-interface-ids eni-0e246d3269EXAMPLE
```

输出：

```
{
    "DeregisteredMulticastGroupMembers": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef7EXAMPLE",
        "RegisteredNetworkInterfaceIds": [
            "eni-0e246d3269EXAMPLE"
        ],
        "GroupIpAddress": "224.0.1.0"
    }
}
```

有关更多信息，请参阅《Amazon 中转网关用户指南》中的从多播组中取消注册成员。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTransitGatewayMulticastGroupMembers。

deregister-transit-gateway-multicast-group-source

以下代码示例演示了如何使用 deregister-transit-gateway-multicast-group-source。

Amazon CLI

从中转网关多播组中取消注册源

此示例从多播组中取消注册指定的网络接口组源。

```
aws ec2 register-transit-gateway-multicast-group-sources \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
    --group-ip-address 224.0.1.0 \
    --network-interface-ids eni-07f290fc3c090cbae
```

输出：

```
{
    "DeregisteredMulticastGroupSources": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
        "DeregisteredNetworkInterfaceIds": [
            "eni-07f290fc3c090cbae"
        ],
        "GroupIpAddress": "224.0.1.0"
    }
}
```

有关更多信息，请参阅《Amazon 中转网关用户指南》中的从多播组中取消注册源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTransitGatewayMulticastGroupSource。

describe-account-attributes

以下代码示例演示了如何使用 describe-account-attributes。

Amazon CLI

描述 Amazon 账户的所有属性

此示例描述了您的 Amazon 账户的属性。

命令:

```
aws ec2 describe-account-attributes
```

输出：

```
{
    "AccountAttributes": [
        {
            "AttributeName": "vpc-max-security-groups-per-interface",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "max-instances",
            "AttributeValues": [
                {
                    "AttributeValue": "20"
                }
            ]
        },
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
            ]
        },
        {
            "AttributeName": "default-vpc",
            "AttributeValues": [
                {
                    "AttributeValue": "none"
                }
            ]
        },
```

```
        {
            "AttributeName": "max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "vpc-max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        }
    ]
}
```

描述 Amazon 账户的单个属性

此示例描述了 Amazon 账户的 `supported-platforms` 属性。

命令:

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

输出：

```
{
    "AccountAttributes": [
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
            ]
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAttributes。

describe-address-transfers

以下代码示例演示了如何使用 describe-address-transfers。

Amazon CLI

描述弹性 IP 地址转移

以下 describe-address-transfers 示例描述了指定弹性 IP 地址的弹性 IP 地址转移。

```
aws ec2 describe-address-transfers \
    --allocation-ids eipalloc-09ad461b0d03f6aaf
```

输出：

```
{
    "AddressTransfers": [
        {
            "PublicIp": "100.21.184.216",
            "AllocationId": "eipalloc-09ad461b0d03f6aaf",
            "TransferAccountId": "123456789012",
            "TransferOfferExpirationTimestamp": "2023-02-22T22:51:01.000Z",
            "AddressTransferStatus": "pending"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的转移弹性 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddressTransfers。

describe-addresses-attribute

以下代码示例演示了如何使用 describe-addresses-attribute。

Amazon CLI

查看与弹性 IP 地址关联的域名的属性

以下 describe-addresses-attribute 示例返回与弹性 IP 地址关联的域名的属性。

Linux：

```
aws ec2 describe-addresses-attribute \
    --allocation-ids eipalloc-abcdef01234567890 \
    --attribute domain-name
```

Windows:

```
aws ec2 describe-addresses-attribute ^
    --allocation-ids eipalloc-abcdef01234567890 ^
    --attribute domain-name
```

输出：

```
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.com."
        }
    ]
}
```

要查看弹性 IP 地址的属性，您必须先将域名与弹性 IP 地址相关联。有关更多信息，请参阅《Amazon EC2 用户指南》中的将反向 DNS 用于电子邮件应用程序，或《Amazon CLI 命令参考》中的 modify-address-attribute。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddressesAttribute。

describe-addresses

以下代码示例演示了如何使用 describe-addresses。

Amazon CLI

示例 1：检索所有弹性 IP 地址的详细信息

以下 describe addresses 示例显示有关弹性 IP 地址的详细信息。

```
aws ec2 describe-addresses
```

输出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "198.51.100.0",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        },
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
            "PublicIp": "203.0.113.0",
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

示例 2：检索 EC2-VPC 弹性 IP 地址的详细信息

以下 `describe-addresses` 示例显示，与 VPC 中的实例结合使用的弹性 IP 地址的详细信息。

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=vpc"
```

输出：

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
```

```
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
            "PublicIp": "203.0.113.0",
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

示例 3：检索由分配 ID 指定的弹性 IP 地址的详细信息

以下 `describe-addresses` 示例显示有关具有指定分配 ID 的弹性 IP 地址的详细信息，该地址与 EC2-VPC 中的实例相关联。

```
aws ec2 describe-addresses \
    --allocation-ids eipalloc-282d9641
```

输出：

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-1a2b3c4d",
            "AssociationId": "eipassoc-123abc12",
            "NetworkInterfaceOwnerId": "1234567891012",
            "PublicIp": "203.0.113.25",
            "AllocationId": "eipalloc-282d9641",
            "PrivateIpAddress": "10.251.50.12"
        }
    ]
}
```

示例 4：检索由 VPC 私有 IP 地址指定的弹性 IP 地址的详细信息

以下 `describe-addresses` 示例显示，与 EC2-VPC 中特定私有 IP 地址关联的弹性 IP 地址的详细信息。

```
aws ec2 describe-addresses \
```

```
    --filters "Name=private-ip-address,Values=10.251.50.12"
```

示例 5：检索 EC2-Classic 中有关弹性 IP 地址的详细信息

以下 describe-addresses 示例显示有关在 EC2-Classic 中使用的弹性 IP 地址的详细信息。

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=standard"
```

输出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        }
    ]
}
```

示例 6：检索由公有 IP 地址指定的弹性 IP 地址详细信息

以下 describe-addresses 示例显示有关具有值 203.0.110.25 的弹性 IP 地址的详细信息，该地址与 EC2-Classic 中的实例相关联。

```
aws ec2 describe-addresses \
    --public-ips 203.0.110.25
```

输出：

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddresses。

`describe-aggregate-id-format`

以下代码示例演示了如何使用 `describe-aggregate-id-format`。

Amazon CLI

描述区域中所有资源类型的较长 ID 格式设置

以下 `describe-aggregate-id-format` 示例描述了当前区域的整体长 ID 格式状态。`Deadline` 值表示这些资源从短 ID 格式永久切换到长 ID 格式的截止时间已过期。`UseLongIdsAggregated` 值表示所有 IAM 用户和 IAM 角色均配置为对所有资源类型使用长 ID 格式。

```
aws ec2 describe-aggregate-id-format
```

输出：

```
{
    "UseLongIdsAggregated": true,
    "Statuses": [
        {
            "Deadline": "2018-08-13T02:00:00.000Z",
            "Resource": "network-interface-attachment",
            "UseLongIds": true
        },
        {
            "Deadline": "2016-12-13T02:00:00.000Z",
            "Resource": "instance",
            "UseLongIds": true
        },
        {
            "Deadline": "2018-08-13T02:00:00.000Z",
            "Resource": "elastic-ip-association",
            "UseLongIds": true
        },
        ...
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAggregateIdFormat。

describe-availability-zones

以下代码示例演示了如何使用 describe-availability-zones。

Amazon CLI

描述可用区

以下示例 describe-availability-zones 显示了可供您使用的可用区详细信息。响应仅包括当前区域的可用区。在本示例中，将使用配置文件默认的 us-west-2（俄勒冈州）区域。

```
aws ec2 describe-availability-zones
```

输出：

```
{
    "AvailabilityZones": [
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2a",
            "ZoneId": "usw2-az1",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2b",
            "ZoneId": "usw2-az2",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
```

```
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2c",
            "ZoneId": "usw2-az3",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2d",
            "ZoneId": "usw2-az4",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opted-in",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2-lax-1a",
            "ZoneId": "usw2-lax1-az1",
            "GroupName": "us-west-2-lax-1",
            "NetworkBorderGroup": "us-west-2-lax-1"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAvailabilityZones。

describe-aws-network-performance-metric-subscription

以下代码示例演示了如何使用 `describe-aws-network-performance-metric-subscription`。

Amazon CLI

描述您的指标订阅

以下 describe-aws-network-performance-metric-subscriptions 示例描述了您的指标订阅。

```
aws ec2 describe-aws-network-performance-metric-subscriptions
```

输出：

```
{
    "Subscriptions": [
        {
            "Source": "us-east-1",
            "Destination": "eu-west-1",
            "Metric": "aggregate-latency",
            "Statistic": "p50",
            "Period": "five-minutes"
        }
    ]
}
```

有关更多信息，请参阅《基础结构性能用户指南》中的管理订阅。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAwsNetworkPerformanceMetricSubscription。

describe-aws-network-performance-metric-subscriptions

以下代码示例演示了如何使用 describe-aws-network-performance-metric-subscriptions。

Amazon CLI

描述您的指标订阅

以下 describe-aws-network-performance-metric-subscriptions 示例描述了您的指标订阅。

```
aws ec2 describe-aws-network-performance-metric-subscriptions
```

输出：

```
{
```

```
    "Subscriptions": [
        {
            "Source": "us-east-1",
            "Destination": "eu-west-1",
            "Metric": "aggregate-latency",
            "Statistic": "p50",
            "Period": "five-minutes"
        }
    ]
}
```

有关更多信息，请参阅《基础结构性能用户指南》中的管理订阅。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAwsNetworkPerformanceMetricSubscriptions。

describe-bundle-tasks

以下代码示例演示了如何使用 describe-bundle-tasks。

Amazon CLI

描述您的捆绑任务

此示例描述了您的所有捆绑任务。

命令:

```
aws ec2 describe-bundle-tasks
```

输出：

```
{
  "BundleTasks": [
    {
      "UpdateTime": "2015-09-15T13:26:54.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "Storage": {
        "S3": {
            "Prefix": "winami",
            "Bucket": "bundletasks"
        }
```

```
      },
      "State": "bundling",
      "StartTime": "2015-09-15T13:24:35.000Z",
      "Progress": "3%",
      "BundleId": "bun-2a4e041c"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBundleTasks。

describe-byoip-cidrs

以下代码示例演示了如何使用 describe-byoip-cidrs。

Amazon CLI

描述您的预置地址范围

以下 describe-byoip-cidrs 示例显示了有关您预置供 Amazon 使用的公有 IPv4 地址范围的详细信息。

```
aws ec2 describe-byoip-cidrs
```

输出：

```
{
    "ByoipCidrs": [
        {
            "Cidr": "203.0.113.25/24",
            "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",
            "State": "provisioned"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeByoipCidrs。

describe-capacity-reservation-fleets

以下代码示例演示了如何使用 describe-capacity-reservation-fleets。

Amazon CLI

查看容量预留实例集

以下 describe-capacity-reservation-fleets 示例列出了指定容量预留实例集的配置和容量信息。它还列出了有关实例集中各个容量预留的详细信息。

```
aws ec2 describe-capacity-reservation-fleets \
    --capacity-reservation-fleet-ids crf-abcdef01234567890
```

输出：

```
{
    "CapacityReservationFleets": [
        {
            "State": "active",
            "EndDate": "2022-12-31T23:59:59.000Z",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "CapacityReservationFleetId": "crf-abcdef01234567890",
            "Tenancy": "default",
            "InstanceTypeSpecifications": [
                {
                    "CapacityReservationId": "cr-1234567890abcdef0",
                    "AvailabilityZone": "us-east-1a",
                    "FulfilledCapacity": 5.0,
                    "Weight": 1.0,
                    "CreateDate": "2022-07-02T08:34:33.398Z",
                    "InstancePlatform": "Linux/UNIX",
                    "TotalInstanceCount": 5,
                    "Priority": 1,
                    "EbsOptimized": true,
                    "InstanceType": "m5.xlarge"
                }
            ],
            "TotalTargetCapacity": 5,
            "TotalFulfilledCapacity": 5.0,
            "CreateTime": "2022-07-02T08:34:33.397Z",
            "AllocationStrategy": "prioritized"
        }
    ]
}
```

有关容量预留实例集的更多信息，请参阅《Amazon EC2 用户指南》中的容量预留实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCapacityReservationFleets。

describe-capacity-reservations

以下代码示例演示了如何使用 describe-capacity-reservations。

Amazon CLI

示例 1：描述一个或多个容量预留

以下 describe-capacity-reservations 示例显示了有关当前 Amazon 区域中所有容量预留的详细信息。

```
aws ec2 describe-capacity-reservations
```

输出：

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
            "OwnerId": "123456789111",
            "CapacityReservationArn": "arn:aws:ec2:us-east-1:123456789111:capacity-
reservation/cr-1234abcd56EXAMPLE",
            "AvailabilityZoneId": "use1-az2",
            "InstanceType": "c5.large",
            "InstancePlatform": "Linux/UNIX",
            "AvailabilityZone": "us-east-1a",
            "Tenancy": "default",
            "TotalInstanceCount": 1,
            "AvailableInstanceCount": 1,
            "EbsOptimized": true,
            "EphemeralStorage": false,
            "State": "active",
            "StartDate": "2024-10-23T15:00:24+00:00",
            "EndDateType": "unlimited",
            "InstanceMatchCriteria": "open",
            "CreateDate": "2024-10-23T15:00:24+00:00",
            "Tags": [],
            "CapacityAllocations": []
```

```
        },
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
            "OwnerId": "123456789111",
            "CapacityReservationArn": "arn:aws:ec2:us-east-1:123456789111:capacity-
reservation/cr-abcdEXAMPLE9876ef",
            "AvailabilityZoneId": "use1-az2",
            "InstanceType": "c4.large",
            "InstancePlatform": "Linux/UNIX",
            "AvailabilityZone": "us-east-1a",
            "Tenancy": "default",
            "TotalInstanceCount": 1,
            "AvailableInstanceCount": 1,
            "EbsOptimized": true,
            "EphemeralStorage": false,
            "State": "cancelled",
            "StartDate": "2024-10-23T15:01:03+00:00",
            "EndDateType": "unlimited",
            "InstanceMatchCriteria": "open",
            "CreateDate": "2024-10-23T15:01:02+00:00",
            "Tags": [],
            "CapacityAllocations": []
        }
    ]
}
```

示例 2：描述一个或多个容量预留

以下 describe-capacity-reservations 示例显示有关指定容量预留的详细信息。

```
aws ec2 describe-capacity-reservations \
    --capacity-reservation-ids cr-1234abcd56EXAMPLE
```

输出：

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
            "OwnerId": "123456789111",
            "CapacityReservationArn": "arn:aws:ec2:us-east-1:123456789111:capacity-
reservation/cr-abcdEXAMPLE9876ef",
            "AvailabilityZoneId": "use1-az2",
```

```
            "InstanceType": "c4.large",
            "InstancePlatform": "Linux/UNIX",
            "AvailabilityZone": "us-east-1a",
            "Tenancy": "default",
            "TotalInstanceCount": 1,
            "AvailableInstanceCount": 1,
            "EbsOptimized": true,
            "EphemeralStorage": false,
            "State": "active",
            "StartDate": "2024-10-23T15:01:03+00:00",
            "EndDateType": "unlimited",
            "InstanceMatchCriteria": "open",
            "CreateDate": "2024-10-23T15:01:02+00:00",
            "Tags": [],
            "CapacityAllocations": []
        }
    ]
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的查看容量预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCapacityReservations。

describe-carrier-gateways

以下代码示例演示了如何使用 describe-carrier-gateways。

Amazon CLI

描述所有运营商网关

以下 describe-carrier-gateways 示例列出您的所有运营商网关。

```
aws ec2 describe-carrier-gateways
```

输出：

```
{
    "CarrierGateways": [
        {
            "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",
            "VpcId": "vpc-0c529aEXAMPLE",
```

```
            "State": "available",
            "OwnerId": "123456789012",
            "Tags": [
                {

                    "Key": "example",
                    "Value": "tag"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的“运营商网关”<https://docs.aws.amazon.com/vpc/latest/userguide/Carrier_Gateway.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCarrierGateways。

`describe-classic-link-instances`

以下代码示例演示了如何使用 `describe-classic-link-instances`。

Amazon CLI

描述链接的 EC2-Classic 实例

此示例列出了您的所有链接的 EC2-Classic 实例。

命令:

```
aws ec2 describe-classic-link-instances
```

输出：

```
{
        "Instances": [
                {
                        "InstanceId": "i-1234567890abcdef0",
                        "VpcId": "vpc-88888888",
                        "Groups": [
                                {
                                        "GroupId": "sg-11122233"
                                }
```

```
                        ],
                        "Tags": [
                                {
                                        "Value": "ClassicInstance",
                                        "Key": "Name"
                                }
                        ]
                },
                {
                        "InstanceId": "i-0598c7d356eba48d7",
                        "VpcId": "vpc-12312312",
                        "Groups": [
                                {
                                        "GroupId": "sg-aabbccdd"
                                }
                        ],
                        "Tags": [
                                {
                                        "Value": "ClassicInstance2",
                                        "Key": "Name"
                                }
                        ]
                }
        ]
}
```

此示例列出了所有链接的 EC2-Classic 实例，并筛选响应以仅包含链接到 VPC vpc-88888888 的实例。

命令:

```
aws ec2 describe-classic-link-instances --filter "Name=vpc-id,Values=vpc-88888888"
```

输出：

```
{
        "Instances": [
                {
                        "InstanceId": "i-1234567890abcdef0",
                        "VpcId": "vpc-88888888",
                        "Groups": [
                                {
                                        "GroupId": "sg-11122233"
```

```
                                }
                        ],
                        "Tags": [
                                {
                                        "Value": "ClassicInstance",
                                        "Key": "Name"
                                }
                        ]
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClassicLinkInstances。

describe-client-vpn-authorization-rules

以下代码示例演示了如何使用 describe-client-vpn-authorization-rules。

Amazon CLI

描述 Client VPN 端点的授权规则

以下 describe-client-vpn-authorization-rules 示例显示了有关指定 Client VPN 端点授权规则的详细信息。

```
aws ec2 describe-client-vpn-authorization-rules \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "AuthorizationRules": [
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "GroupId": "",
            "AccessAll": true,
            "DestinationCidr": "0.0.0.0/0",
            "Status": {
                "Code": "active"
            }
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的授权规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClientVpnAuthorizationRules。

describe-client-vpn-connections

以下代码示例演示了如何使用 describe-client-vpn-connections。

Amazon CLI

描述 Client VPN 端点的连接

以下 describe-client-vpn-connections 示例显示了有关客户端与指定 Client VPN 端点的连接的详细信息。

```
aws ec2 describe-client-vpn-connections \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "Connections": [
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "Timestamp": "2019-08-12 07:58:34",
            "ConnectionId": "cvpn-connection-0e03eb24267165acd",
            "ConnectionEstablishedTime": "2019-08-12 07:57:14",
            "IngressBytes": "32302",
            "EgressBytes": "5696",
            "IngressPackets": "332",
            "EgressPackets": "67",
            "ClientIp": "172.31.0.225",
            "CommonName": "client1.domain.tld",
            "Status": {
                "Code": "terminated"
            },
            "ConnectionEndTime": "2019-08-12 07:58:34"
        },
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
```

```
            "Timestamp": "2019-08-12 08:02:54",
            "ConnectionId": "cvpn-connection-00668867a40f18253",
            "ConnectionEstablishedTime": "2019-08-12 08:02:53",
            "IngressBytes": "2951",
            "EgressBytes": "2611",
            "IngressPackets": "9",
            "EgressPackets": "6",
            "ClientIp": "172.31.0.226",
            "CommonName": "client1.domain.tld",
            "Status": {
                "Code": "active"
            },
            "ConnectionEndTime": "-"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的客户端连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClientVpnConnections。

`describe-client-vpn-endpoints`

以下代码示例演示了如何使用 `describe-client-vpn-endpoints`。

Amazon CLI

描述 Client VPN 端点

以下 `describe-client-vpn-endpoints` 示例显示了有关您的所有 Client VPN 端点的详细信息。

```
aws ec2 describe-client-vpn-endpoints
```

输出：

```
{
    "ClientVpnEndpoints": [
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "Description": "Endpoint for Admin access",
            "Status": {
                "Code": "available"
```

```
            },
            "CreationTime": "2020-11-13T11:37:27",
            "DnsName": "*.cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-
south-1.amazonaws.com",
            "ClientCidrBlock": "172.31.0.0/16",
            "DnsServers": [
                "8.8.8.8"
            ],
            "SplitTunnel": false,
            "VpnProtocol": "openvpn",
            "TransportProtocol": "udp",
            "VpnPort": 443,
            "ServerCertificateArn": "arn:aws:acm:ap-
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "AuthenticationOptions": [
                {
                    "Type": "certificate-authentication",
                    "MutualAuthentication": {
                        "ClientRootCertificateChain": "arn:aws:acm:ap-
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"
                    }
                }
            ],
            "ConnectionLogOptions": {
                "Enabled": true,
                "CloudwatchLogGroup": "Client-vpn-connection-logs",
                "CloudwatchLogStream": "cvpn-endpoint-123456789123abcde-ap-
south-1-2020/11/13-FCD8HEMVaCcw"
            },
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Client VPN"
                }
            ],
            "SecurityGroupIds": [
                "sg-aabbcc11223344567"
            ],
            "VpcId": "vpc-a87f92c1",
            "SelfServicePortalUrl": "https://self-service.clientvpn.amazonaws.com/
endpoints/cvpn-endpoint-123456789123abcde",
            "ClientConnectOptions": {
                 "Enabled": false
            }
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的 Client VPN 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClientVpnEndpoints。

`describe-client-vpn-routes`

以下代码示例演示了如何使用 `describe-client-vpn-routes`。

Amazon CLI

描述 Client VPN 端点的路由

以下 `describe-client-vpn-routes` 示例显示了有关指定 Client VPN 端点的路由的详细信息。

```
aws ec2 describe-client-vpn-routes \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "Routes": [
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "DestinationCidr": "10.0.0.0/16",
            "TargetSubnet": "subnet-0123456789abcabca",
            "Type": "Nat",
            "Origin": "associate",
            "Status": {
                "Code": "active"
            },
            "Description": "Default Route"
        },
        {
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "DestinationCidr": "0.0.0.0/0",
            "TargetSubnet": "subnet-0123456789abcabca",
            "Type": "Nat",
```

```
            "Origin": "add-route",
            "Status": {
                "Code": "active"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClientVpnRoutes。

describe-client-vpn-target-networks

以下代码示例演示了如何使用 describe-client-vpn-target-networks。

Amazon CLI

描述 Client VPN 端点的目标网络

以下 describe-client-vpn-target-networks 示例显示了有关指定 Client VPN 端点的目标网络的详细信息。

```
aws ec2 describe-client-vpn-target-networks \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "ClientVpnTargetNetworks": [
        {
            "AssociationId": "cvpn-assoc-012e837060753dc3d",
            "VpcId": "vpc-11111222222333333",
            "TargetNetworkId": "subnet-0123456789abcabca",
            "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
            "Status": {
                "Code": "associating"
            },
            "SecurityGroups": [
                "sg-012345678910abcab"
            ]
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的目标网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClientVpnTargetNetworks。

describe-coip-pools

以下代码示例演示了如何使用 describe-coip-pools。

Amazon CLI

描述客户拥有的 IP 地址池

以下 describe-coip-pools 示例描述了 Amazon 账户中客户拥有的 IP 地址池。

```
aws ec2 describe-coip-pools
```

输出：

```
{
    "CoipPools": [
        {
            "PoolId": "ipv4pool-coip-123a45678bEXAMPLE",
            "PoolCidrs": [
                "0.0.0.0/0"
            ],
            "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
            "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-
coip-123a45678bEXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCoipPools。

describe-conversion-tasks

以下代码示例演示了如何使用 describe-conversion-tasks。

Amazon CLI

查看转换任务的状态

此示例返回 ID 为 import-i-ffvko9js 的转换任务的状态。

命令:

```
aws ec2 describe-conversion-tasks --conversion-task-ids import-i-ffvko9js
```

输出：

```
{
    "ConversionTasks": [
        {
            "ConversionTaskId": "import-i-ffvko9js",
            "ImportInstance": {
                "InstanceId": "i-1234567890abcdef0",
                "Volumes": [
                    {
                        "Volume": {
                            "Id": "vol-049df61146c4d7901",
                            "Size": 16
                        },
                        "Status": "completed",
                        "Image": {
                            "Size": 1300687360,
                            "ImportManifestUrl": "https://s3.amazonaws.com/
myimportbucket/411443cd-d620-4f1c-9d66-13144EXAMPLE/RHEL5.vmdkmanifest.xml?
AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=140EXAMPLE&Signature=XYNhznHNgCqsjDxL9wRL
%2FJvEXAMPLE",
                            "Format": "VMDK"
                        },
                        "BytesConverted": 1300682960,
                        "AvailabilityZone": "us-east-1d"
                    }
                ]
            },
            "ExpirationTime": "2014-05-14T22:06:23Z",
            "State": "completed"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConversionTasks。

describe-customer-gateways

以下代码示例演示了如何使用 describe-customer-gateways。

Amazon CLI

描述客户网关

此示例描述客户网关。

命令:

```
aws ec2 describe-customer-gateways
```

输出：

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-b4dc3961",
            "IpAddress": "203.0.113.12",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65000"
        },
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

描述特定的客户网关

此示例描述指定的客户网关。

命令:

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

输出：

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomerGateways。

describe-dhcp-options

以下代码示例演示了如何使用 describe-dhcp-options。

Amazon CLI

示例 1：描述 DHCP 选项

以下 describe-dhcp-options 示例检索有关 DHCP 选项的详细信息。

```
aws ec2 describe-dhcp-options
```

输出：

```
{
    "DhcpOptions": [
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
```

```
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
            ],
            "DhcpOptionsId": "dopt-19edf471",
            "OwnerId": "111122223333"
        },
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
            ],
            "DhcpOptionsId": "dopt-fEXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 DHCP 选项集。

示例 2：描述 DHCP 选项并筛选输出

以下 describe-dhcp-options 示例描述了 DHCP 选项，并使用筛选条件仅返回域名服务器具有 example.com 的 DHCP 选项。该示例使用 --query 参数在输出中仅显示配置信息和 ID。

```
aws ec2 describe-dhcp-options \
    --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
    --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"
```

输出：

```
[
    [
        [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
            },
            {
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "172.16.16.16"
                    }
                ]
            }
        ],
        "dopt-001122334455667ab"
    ]
]
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 DHCP 选项集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDhcpOptions。

describe-egress-only-internet-gateways

以下代码示例演示了如何使用 describe-egress-only-internet-gateways。

Amazon CLI

描述仅出口互联网网关

此示例描述仅出口互联网网关。

命令:

```
aws ec2 describe-egress-only-internet-gateways
```

输出：

```
{
  "EgressOnlyInternetGateways": [
      {
          "EgressOnlyInternetGatewayId": "eigw-015e0e244e24dfe8a",
          "Attachments": [
              {
                  "State": "attached",
                  "VpcId": "vpc-0c62a468"
              }
          ]
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEgressOnlyInternetGateways。

describe-elastic-gpus

以下代码示例演示了如何使用 describe-elastic-gpus。

Amazon CLI

描述弹性 GPU

命令:

```
aws ec2 describe-elastic-gpus --elastic-gpu-
ids egpu-12345678901234567890abcdefghijkl
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeElasticGpus。

describe-export-image-tasks

以下代码示例演示了如何使用 describe-export-image-tasks。

Amazon CLI

监控导出映像任务

以下 describe-export-image-tasks 示例检查指定导出映像任务的状态。在 Amazon S3 中生成的映像文件为 my-export-bucket/exports/export-ami-1234567890abcdef0.vmdk。

```
aws ec2 describe-export-image-tasks \
    --export-image-task-ids export-ami-1234567890abcdef0
```

正在进行的导出映像任务的输出。

```
{
    "ExportImageTasks": [
        {
            "ExportImageTaskId": "export-ami-1234567890abcdef0"
            "Progress": "21",
            "S3ExportLocation": {
                "S3Bucket": "my-export-bucket",
                "S3Prefix": "exports/"
            },
            "Status": "active",
            "StatusMessage": "updating"
        }
    ]
}
```

已完成的导出映像任务的输出。

```
{
    "ExportImageTasks": [
        {
            "ExportImageTaskId": "export-ami-1234567890abcdef0"
            "S3ExportLocation": {
                "S3Bucket": "my-export-bucket",
                "S3Prefix": "exports/"
```

```
            },
            "Status": "completed"
        }
    ]
}
```

有关更多信息，请参阅《VM Import/Export 用户指南》中的从 AMI 导出 VM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeExportImageTasks。

`describe-export-tasks`

以下代码示例演示了如何使用 `describe-export-tasks`。

Amazon CLI

列出有关实例导出任务的详细信息

此示例描述了 ID 为 export-i-fh8sjjsq 的导出任务。

命令:

```
aws ec2 describe-export-tasks --export-task-ids export-i-fh8sjjsq
```

输出：

```
{
    "ExportTasks": [
        {
            "State": "active",
            "InstanceExportDetails": {
                "InstanceId": "i-1234567890abcdef0",
                "TargetEnvironment": "vmware"
            },
            "ExportToS3Task": {
                "S3Bucket": "myexportbucket",
                "S3Key": "RHEL5export-i-fh8sjjsq.ova",
                "DiskImageFormat": "vmdk",
                "ContainerFormat": "ova"
            },
            "Description": "RHEL5 instance",
            "ExportTaskId": "export-i-fh8sjjsq"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeExportTasks。

`describe-fast-launch-images`

以下代码示例演示了如何使用 `describe-fast-launch-images`。

Amazon CLI

描述为加快启动速度而配置的 Windows AMI 的详细信息

以下 `describe-fast-launch-images` 示例描述了账户中为加快启动速度而配置的每个 AMI 的详细信息，包括资源类型、快照配置、启动模板详细信息、最大并行启动项数量、AMI 所有者 ID、快速启动配置的状态、状态发生更改的原因以及状态更改发生的时间。

```
aws ec2 describe-fast-launch-images
```

输出：

```
{
    "FastLaunchImages": [
        {
            "ImageId": "ami-01234567890abcedf",
            "ResourceType": "snapshot",
            "SnapshotConfiguration": {},
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-01234567890abcedf",
                "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-
a8c6215d-94e6-441b-9272-dbd1f87b07e2",
                "Version": "1"
            },
            "MaxParallelLaunches": 6,
            "OwnerId": "0123456789123",
            "State": "enabled",
            "StateTransitionReason": "Client.UserInitiated",
            "StateTransitionTime": "2022-01-27T22:20:06.552000+00:00"
        }
    ]
}
```

有关配置 Windows AMI 以加快启动速度的更多信息，请参阅《Amazon EC2 用户指南》中的配置 AMI 以加快启动速度。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFastLaunchImages。

describe-fast-snapshot-restores

以下代码示例演示了如何使用 describe-fast-snapshot-restores。

Amazon CLI

描述快速快照还原

以下 describe-fast-snapshot-restores 示例显示了状态为 disabled 的所有快速快照还原的详细信息。

```
aws ec2 describe-fast-snapshot-restores \
    --filters Name=state,Values=disabled
```

输出：

```
{
    "FastSnapshotRestores": [
        {
            "SnapshotId": "snap-1234567890abcdef0",
            "AvailabilityZone": "us-west-2c",
            "State": "disabled",
            "StateTransitionReason": "Client.UserInitiated - Lifecycle state
 transition",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.596Z",
            "OptimizingTime": "2020-01-25T23:58:25.573Z",
            "EnabledTime": "2020-01-25T23:59:29.852Z",
            "DisablingTime": "2020-01-26T00:40:56.069Z",
            "DisabledTime": "2020-01-26T00:41:27.390Z"
        }
    ]
}
```

以下 describe-fast-snapshot-restores 示例描述了所有快速快照还原。

```
aws ec2 describe-fast-snapshot-restores
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFastSnapshotRestores。

describe-fleet-history

以下代码示例演示了如何使用 describe-fleet-history。

Amazon CLI

描述 EC2 实例集的历史记录

以下 describe-fleet-history 示例返回指定 EC2 实例集从指定时间开始的历史记录。该输出适用于具有两个正在运行的实例的 EC2 实例集。

```
aws ec2 describe-fleet-history \
    --fleet-id fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
    --start-time 2020-09-01T00:00:00Z
```

输出：

```
{
    "HistoryRecords": [
        {
            "EventInformation": {
                "EventSubType": "submitted"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:05.000Z"
        },
        {
            "EventInformation": {
                "EventSubType": "active"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:15.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "t2.small, ami-07c8bc5c1ce9598c3, ...",
                "EventSubType": "progress"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
```

```
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-083a1c446e66085d2"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-090db02406cc3c2d6"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        }
    ],
    "LastEvaluatedTime": "2020-09-01T19:10:19.000Z",
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
    "StartTime": "2020-08-31T23:53:20.000Z"
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的管理 EC2 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetHistory。

describe-fleet-instances

以下代码示例演示了如何使用 describe-fleet-instances。

Amazon CLI

描述 EC2 实例集的正在运行的实例

以下 describe-fleet-instances 示例描述了指定 EC2 实例集的正在运行的实例。

```
aws ec2 describe-fleet-instances \
    --fleet-id 12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE
```

输出：

```
{
    "ActiveInstances": [
        {
            "InstanceId": "i-090db02406cc3c2d6",
            "InstanceType": "t2.small",
            "SpotInstanceRequestId": "sir-a43gtpfk",
            "InstanceHealth": "healthy"
        },
        {
            "InstanceId": "i-083a1c446e66085d2",
            "InstanceType": "t2.small",
            "SpotInstanceRequestId": "sir-iwcit2nj",
            "InstanceHealth": "healthy"
        }
    ],
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的管理 EC2 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetInstances。

describe-fleets

以下代码示例演示了如何使用 describe-fleets。

Amazon CLI

描述 EC2 实例集

以下 describe-fleets 示例描述了指定的 EC2 实例集。

```
aws ec2 describe-fleets \
    --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE
```

输出：

```
{
    "Fleets": [
        {
            "ActivityStatus": "pending_fulfillment",
```

```
            "CreateTime": "2020-09-01T18:26:05.000Z",
            "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
            "FleetState": "active",
            "ExcessCapacityTerminationPolicy": "termination",
            "FulfilledCapacity": 0.0,
            "FulfilledOnDemandCapacity": 0.0,
            "LaunchTemplateConfigs": [
                {
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateId": "lt-0e632f2855a979cd5",
                        "Version": "1"
                    }
                }
            ],
            "TargetCapacitySpecification": {
                "TotalTargetCapacity": 2,
                "OnDemandTargetCapacity": 0,
                "SpotTargetCapacity": 2,
                "DefaultTargetCapacityType": "spot"
            },
            "TerminateInstancesWithExpiration": false,
            "Type": "maintain",
            "ReplaceUnhealthyInstances": false,
            "SpotOptions": {
                "AllocationStrategy": "lowestPrice",
                "InstanceInterruptionBehavior": "terminate",
                "InstancePoolsToUseCount": 1
            },
            "OnDemandOptions": {
                "AllocationStrategy": "lowestPrice"
            }
        }
    ]
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的管理 EC2 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleets。

describe-flow-logs

以下代码示例演示了如何使用 describe-flow-logs。

Amazon CLI

示例 1：描述所有流日志

以下 describe-flow-logs 示例显示了所有流日志的详细信息。

```
aws ec2 describe-flow-logs
```

输出：

```
{
    "FlowLogs": [
        {
            "CreationTime": "2018-02-21T13:22:12.644Z",
            "DeliverLogsPermissionArn": "arn:aws:iam::123456789012:role/flow-logs-
role",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-aabbccdd112233445",
            "MaxAggregationInterval": 600,
            "FlowLogStatus": "ACTIVE",
            "LogGroupName": "FlowLogGroup",
            "ResourceId": "subnet-12345678901234567",
            "TrafficType": "ALL",
            "LogDestinationType": "cloud-watch-logs",
            "LogFormat": "${version} ${account-id} ${interface-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end}
 ${action} ${log-status}"
        },
        {
            "CreationTime": "2020-02-04T15:22:29.986Z",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-01234567890123456",
            "MaxAggregationInterval": 60,
            "FlowLogStatus": "ACTIVE",
            "ResourceId": "vpc-00112233445566778",
            "TrafficType": "ACCEPT",
            "LogDestinationType": "s3",
            "LogDestination": "arn:aws:s3:::my-flow-log-bucket/custom",
            "LogFormat": "${version} ${vpc-id} ${subnet-id} ${instance-id}
 ${interface-id} ${account-id} ${type} ${srcaddr} ${dstaddr} ${srcport} ${dstport}
 ${pkt-srcaddr} ${pkt-dstaddr} ${protocol} ${bytes} ${packets} ${start} ${end}
 ${action} ${tcp-flags} ${log-status}"
        }
```

```
    ]
}
```

示例 2：描述流日志的子集

以下 `describe-flow-logs` 示例使用筛选条件仅显示 Amazon CloudWatch Logs 中指定日志组中流日志的详细信息。

```
aws ec2 describe-flow-logs \
    --filter "Name=log-group-name,Values=MyFlowLogs"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFlowLogs。

`describe-fpga-image-attribute`

以下代码示例演示了如何使用 `describe-fpga-image-attribute`。

Amazon CLI

描述 Amazon FPGA 映像的属性

此示例描述了指定 AFI 的加载权限。

命令:

```
aws ec2 describe-fpga-image-attribute --fpga-image-id afi-0d123e123bfc85abc --
attribute loadPermission
```

输出：

```
{
  "FpgaImageAttribute": {
      "FpgaImageId": "afi-0d123e123bfc85abc",
      "LoadPermissions": [
          {
              "UserId": "123456789012"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFpgaImageAttribute。

describe-fpga-images

以下代码示例演示了如何使用 describe-fpga-images。

Amazon CLI

描述 Amazon FPGA 映像

此示例描述了账户 123456789012 拥有的 AFI。

命令:

```
aws ec2 describe-fpga-images --filters Name=owner-id,Values=123456789012
```

输出：

```
{
  "FpgaImages": [
      {
          "UpdateTime": "2017-12-22T12:09:14.000Z",
          "Name": "my-afi",
          "PciId": {
              "SubsystemVendorId": "0xfedd",
              "VendorId": "0x1d0f",
              "DeviceId": "0xf000",
              "SubsystemId": "0x1d51"
          },
          "FpgaImageGlobalId": "agfi-123cb27b5e84a0abc",
          "Public": false,
          "State": {
              "Code": "available"
          },
          "ShellVersion": "0x071417d3",
          "OwnerId": "123456789012",
          "FpgaImageId": "afi-0d123e123bfc85abc",
          "CreateTime": "2017-12-22T11:43:33.000Z",
          "Description": "my-afi"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFpgaImages。

describe-host-reservation-offerings

以下代码示例演示了如何使用 describe-host-reservation-offerings。

Amazon CLI

描述专属主机预留产品

此示例描述了可供购买的 M4 实例系列的专属主机预留。

命令:

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-family,Values=m4
```

输出：

```
{
  "OfferingSet": [
      {
          "HourlyPrice": "1.499",
          "OfferingId": "hro-03f707bf363b6b324",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      },
      {
          "HourlyPrice": "1.045",
          "OfferingId": "hro-0ef9181cabdef7a02",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 94608000
      },
      {
          "HourlyPrice": "0.714",
          "OfferingId": "hro-04567a15500b92a51",
          "InstanceFamily": "m4",
          "PaymentOption": "PartialUpfront",
          "UpfrontPrice": "6254.000",
          "Duration": 31536000
      },
      {
          "HourlyPrice": "0.484",
```

```
            "OfferingId": "hro-0d5d7a9d23ed7fbfe",
            "InstanceFamily": "m4",
            "PaymentOption": "PartialUpfront",
            "UpfrontPrice": "12720.000",
            "Duration": 94608000
        },
        {
            "HourlyPrice": "0.000",
            "OfferingId": "hro-05da4108ca998c2e5",
            "InstanceFamily": "m4",
            "PaymentOption": "AllUpfront",
            "UpfrontPrice": "23913.000",
            "Duration": 94608000
        },
        {
            "HourlyPrice": "0.000",
            "OfferingId": "hro-0a9f9be3b95a3dc8f",
            "InstanceFamily": "m4",
            "PaymentOption": "AllUpfront",
            "UpfrontPrice": "12257.000",
            "Duration": 31536000
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHostReservationOfferings。

describe-host-reservations

以下代码示例演示了如何使用 describe-host-reservations。

Amazon CLI

描述您账户中的专属主机预留

此示例描述了您账户中的专属主机预留。

命令:

```
aws ec2 describe-host-reservations
```

输出：

```
{
  "HostReservationSet": [
      {
          "Count": 1,
          "End": "2019-01-10T12:14:09Z",
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "OfferingId": "hro-03f707bf363b6b324",
          "PaymentOption": "NoUpfront",
          "State": "active",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
          ],
          "Start": "2018-01-10T12:14:09Z",
          "HostReservationId": "hr-0d418a3a4ffc669ae",
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHostReservations。

describe-hosts

以下代码示例演示了如何使用 describe-hosts。

Amazon CLI

查看有关专属主机的详细信息

以下 describe-hosts 示例显示了 Amazon 账户中 available 专属主机的详细信息。

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

输出：

```
{
    "Hosts": [
        {
            "HostId": "h-07879acf49EXAMPLE",
            "Tags": [
```

```
                {
                    "Value": "production",
                    "Key": "purpose"
                }
            ],
            "HostProperties": {
                "Cores": 48,
                "TotalVCpus": 96,
                "InstanceType": "m5.large",
                "Sockets": 2
            },
            "Instances": [],
            "State": "available",
            "AvailabilityZone": "eu-west-1a",
            "AvailableCapacity": {
                "AvailableInstanceCapacity": [
                    {
                        "AvailableCapacity": 48,
                        "InstanceType": "m5.large",
                        "TotalCapacity": 48
                    }
                ],
                "AvailableVCpus": 96
            },
            "HostRecovery": "on",
            "AllocationTime": "2019-08-19T08:57:44.000Z",
            "AutoPlacement": "off"
        }
    ]
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的查看专属主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHosts。

`describe-iam-instance-profile-associations`

以下代码示例演示了如何使用 `describe-iam-instance-profile-associations`。

Amazon CLI

描述 IAM 实例配置文件关联

本示例描述了所有 IAM 实例配置文件关联。

命令:

```
aws ec2 describe-iam-instance-profile-associations
```

输出：

```
{
  "IamInstanceProfileAssociations": [
      {
          "InstanceId": "i-09eb09efa73ec1dee",
          "State": "associated",
          "AssociationId": "iip-assoc-0db249b1f25fa24b8",
          "IamInstanceProfile": {
              "Id": "AIPAJVQN4F5WVLGCJDRGM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
          }
      },
      {
          "InstanceId": "i-0402909a2f4dffd14",
          "State": "associating",
          "AssociationId": "iip-assoc-0d1ec06278d29f44a",
          "IamInstanceProfile": {
              "Id": "AGJAJVQN4F5WVLGCJABCM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
          }
      }
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIamInstanceProfileAssociations。

describe-id-format

以下代码示例演示了如何使用 describe-id-format。

Amazon CLI

示例 1：描述资源的 ID 格式

以下 describe-id-format 示例描述了安全组的 ID 格式。

```
aws ec2 describe-id-format \
    --resource security-group
```

在以下示例输出中，Deadline 值表示此资源类型从短 ID 格式永久切换到长 ID 格式的截止时间已于 2018 年 8 月 15 日 00:00 UTC 到期。

```
{
    "Statuses": [
        {
            "Deadline": "2018-08-15T00:00:00.000Z",
            "Resource": "security-group",
            "UseLongIds": true
        }
    ]
}
```

示例 2：描述所有资源的 ID 格式

以下 describe-id-format 示例描述了所有资源类型的 ID 格式。所有曾支持短 ID 格式的资源类型均已切换为使用长 ID 格式。

```
aws ec2 describe-id-format
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIdFormat。

describe-identity-id-format

以下代码示例演示了如何使用 describe-identity-id-format。

Amazon CLI

描述 IAM 角色的 ID 格式

以下 describe-identity-id-format 示例描述了 Amazon 账户中由 IAM 角色 EC2Role 创建的实例所接收的 ID 格式。

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
    --resource instance
```

以下输出表明由此角色创建的实例会收到长 ID 格式的 ID。

```
{
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "instance",
            "UseLongIds": true
        }
    ]
}
```

描述 IAM 用户的 ID 格式

以下 `describe-identity-id-format` 示例描述了 Amazon 账户中由 IAM 用户 `AdminUser` 创建的快照所接收的 ID 格式。

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:user/AdminUser \
    --resource snapshot
```

该输出表明此用户创建的快照接收长 ID 格式的 ID。

```
{
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "snapshot",
            "UseLongIds": true
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIdentityIdFormat。

describe-image-attribute

以下代码示例演示了如何使用 `describe-image-attribute`。

Amazon CLI

描述 AMI 的启动权限

此示例描述了指定 AMI 的启动权限。

命令:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --
attribute launchPermission
```

输出：

```
{
    "LaunchPermissions": [
        {
            "UserId": "123456789012"
        }
    ],
    "ImageId": "ami-5731123e",
}
```

描述 AMI 的产品代码

此示例描述了指定 AMI 的产品代码。请注意，此 AMI 没有产品代码。

命令:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

输出：

```
{
    "ProductCodes": [],
    "ImageId": "ami-5731123e",
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImageAttribute。

describe-images

以下代码示例演示了如何使用 describe-images。

Amazon CLI

示例 1：描述 AMI

以下 describe-images 示例描述了指定区域中的指定 AMI。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-1234567890EXAMPLE
```

输出：

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "PlatformDetails": "Red Hat Enterprise Linux",
            "EnaSupport": true,
            "Hypervisor": "xen",
            "State": "available",
            "SriovNetSupport": "simple",
            "ImageId": "ami-1234567890EXAMPLE",
            "UsageOperation": "RunInstances:0010",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-
GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": true,
            "ImageType": "machine",
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的亚马逊机器映像（AMI）。

示例 2：根据筛选器描述 AMI

以下 `describe-images` 示例描述了由 Amazon 提供，并受 Amazon EBS 支持的 Windows AMI。

```
aws ec2 describe-images \
    --owners amazon \
    --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"
```

有关 `describe-images` 的输出示例，请参阅示例 1。

有关使用筛选器的其他示例，请参阅《Amazon EC2 用户指南》中的列出和筛选资源。

示例 3：根据标签描述 AMI

以下 `describe-images` 示例描述了带有标签 `Type=Custom` 的所有 AMI。示例使用 `--query` 参数仅显示 AMI ID。

```
aws ec2 describe-images \
    --filters "Name=tag:Type,Values=Custom" \
    --query 'Images[*].[ImageId]' \
    --output text
```

输出：

```
ami-1234567890EXAMPLE
ami-0abcdef1234567890
```

有关使用标签筛选器的更多示例，请参阅《Amazon EC2 用户指南》中的使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImages。

describe-import-image-tasks

以下代码示例演示了如何使用 `describe-import-image-tasks`。

Amazon CLI

监控导入映像任务

以下 describe-import-image-tasks 示例检查指定的导入映像任务的状态。

```
aws ec2 describe-import-image-tasks \
    --import-task-ids import-ami-1234567890abcdef0
```

正在进行的导入映像任务的输出。

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
            "Progress": "28",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
                    "Format": "ova",
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
                }
            ],
            "Status": "active",
            "StatusMessage": "converting"
        }
    ]
}
```

已完成的导入映像任务的输出。生成的 AMI 的 ID 由 ImageId 提供。

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
            "ImageId": "ami-1234567890abcdef0",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
```

```
                    "Format": "ova",
                    "SnapshotId": "snap-1234567890abcdef0"
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
                }
            ],
            "Status": "completed"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImportImageTasks。

describe-import-snapshot-tasks

以下代码示例演示了如何使用 describe-import-snapshot-tasks。

Amazon CLI

监控导入快照任务

以下 describe-import-snapshot-tasks 示例检查指定的导入快照任务的状态。

```
aws ec2 describe-import-snapshot-tasks \
    --import-task-ids import-snap-1234567890abcdef0
```

正在进行的导入快照任务的输出。

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "Progress": "42",
                "Status": "active",
```

```
                "StatusMessage": "downloading/converting",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
}
```

已完成的导入快照任务的输出。生成的快照的 ID 由 SnapshotId 提供。

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "SnapshotId": "snap-1234567890abcdef0"
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImportSnapshotTasks。

describe-instance-attribute

以下代码示例演示了如何使用 describe-instance-attribute。

Amazon CLI

描述实例类型

此示例描述了指定实例的实例类型。

命令:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --
attribute instanceType
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0"
    "InstanceType": {
        "Value": "t1.micro"
    }
}
```

描述 disableApiTermination 属性

此示例描述了指定实例的 disableApiTermination 属性。

命令:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --
attribute disableApiTermination
```

输出：

```
{
"InstanceId": "i-1234567890abcdef0"
    "DisableApiTermination": {
        "Value": "false"
    }
}
```

描述实例的块设备映射

此示例描述了指定实例的 blockDeviceMapping 属性。

命令:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --
attribute blockDeviceMapping
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0"
    "BlockDeviceMappings": [
        {
            "DeviceName": "/dev/sda1",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": true,
                "VolumeId": "vol-049df61146c4d7901",
                "AttachTime": "2013-05-17T22:42:34.000Z"
            }
        },
        {
            "DeviceName": "/dev/sdf",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": false,
                "VolumeId": "vol-049df61146c4d7901",
                "AttachTime": "2013-09-10T23:07:00.000Z"
            }
        }
    ],
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceAttribute。

describe-instance-connect-endpoints

以下代码示例演示了如何使用 describe-instance-connect-endpoints。

Amazon CLI

描述 EC2 实例连接端点

以下 describe-instance-connect-endpoints 示例描述指定的 EC2 实例连接端点。

```
aws ec2 describe-instance-connect-endpoints \
    --region us-east-1 \
    --instance-connect-endpoint-ids eice-0123456789example
```

输出：

```
{
    "InstanceConnectEndpoints": [
        {
            "OwnerId": "111111111111",
            "InstanceConnectEndpointId": "eice-0123456789example",
            "InstanceConnectEndpointArn": "arn:aws:ec2:us-
east-1:111111111111:instance-connect-endpoint/eice-0123456789example",
            "State": "create-complete",
            "StateMessage": "",
            "DnsName": "eice-0123456789example.b67b86ba.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
            "NetworkInterfaceIds": [
                "eni-0123456789example"
            ],
            "VpcId": "vpc-0123abcd",
            "AvailabilityZone": "us-east-1d",
            "CreatedAt": "2023-02-07T12:05:37+00:00",
            "SubnetId": "subnet-0123abcd",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的创建 EC2 实例连接端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceConnectEndpoints。

describe-instance-credit-specifications

以下代码示例演示了如何使用 describe-instance-credit-specifications。

Amazon CLI

描述一个或多个实例的 CPU 使用率的积分选项

以下 describe-instance-credit-specifications 示例描述了指定实例的 CPU 积分选项。

```
aws ec2 describe-instance-credit-specifications \
    --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "InstanceCreditSpecifications": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CpuCredits": "unlimited"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用可突增性能实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceCreditSpecifications。

describe-instance-event-notification-attributes

以下代码示例演示了如何使用 describe-instance-event-notification-attributes。

Amazon CLI

描述计划事件通知的标签

以下 describe-instance-event-notification-attributes 示例描述了要在计划事件通知中显示的标签。

```
aws ec2 describe-instance-event-notification-attributes
```

输出：

```
{
    "InstanceTagAttribute": {
        "InstanceTagKeys": [],
        "IncludeAllTagsOfInstance": true
    }
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的实例的计划事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceEventNotificationAttributes。

describe-instance-event-windows

以下代码示例演示了如何使用 describe-instance-event-windows。

Amazon CLI

示例 1：描述所有事件窗口

以下 describe-instance-event-windows 示例描述了指定区域中的所有事件窗口。

```
aws ec2 describe-instance-event-windows \
    --region us-east-1
```

输出：

```
{
    "InstanceEventWindows": [
        {
            "InstanceEventWindowId": "iew-0abcdef1234567890",
            "Name": "myEventWindowName",
            "CronExpression": "* 21-23 * * 2,3",
            "AssociationTarget": {
                "InstanceIds": [
                    "i-1234567890abcdef0",
                    "i-0598c7d356eba48d7"
                ],
                "Tags": [],
                "DedicatedHostIds": []
            },
            "State": "active",
            "Tags": []
        }

        ...

    ],
    "NextToken": "9d624e0c-388b-4862-a31e-a85c64fc1d4a"
}
```

示例 2：描述特定事件窗口

以下 describe-instance-event-windows 示例通过使用 instance-event-window 参数描述特定事件窗口来描述特定事件。

```
aws ec2 describe-instance-event-windows \
    --region us-east-1 \
    --instance-event-window-ids iew-0abcdef1234567890
```

输出：

```
{
    "InstanceEventWindows": [
        {
            "InstanceEventWindowId": "iew-0abcdef1234567890",
            "Name": "myEventWindowName",
            "CronExpression": "* 21-23 * * 2,3",
            "AssociationTarget": {
                "InstanceIds": [
                    "i-1234567890abcdef0",
                    "i-0598c7d356eba48d7"
                ],
                "Tags": [],
                "DedicatedHostIds": []
            },
            "State": "active",
            "Tags": []
        }
}
```

示例 3：描述匹配一个或多个筛选条件的事件窗口

以下 describe-instance-event-windows 示例使用 filter 参数描述了匹配一个或多个筛选条件的事件窗口。instance-id 筛选条件用于描述与指定实例关联的所有事件窗口。使用筛选器时，它会进行直接匹配。但是，instance-id 筛选器不同。如果没有与实例 ID 直接匹配，它会回退到与事件窗口的间接关联，例如实例的标签或专属主机 ID（如果实例为专属主机）。

```
aws ec2 describe-instance-event-windows \
    --region us-east-1 \
    --filters Name=instance-id,Values=i-1234567890abcdef0 \
    --max-results 100 \
```

```
    --next-token <next-token-value>
```

输出：

```
{
    "InstanceEventWindows": [
        {
            "InstanceEventWindowId": "iew-0dbc0adb66f235982",
            "TimeRanges": [
                {
                    "StartWeekDay": "sunday",
                    "StartHour": 2,
                    "EndWeekDay": "sunday",
                    "EndHour": 8
                }
            ],
            "Name": "myEventWindowName",
            "AssociationTarget": {
                "InstanceIds": [],
                "Tags": [],
                "DedicatedHostIds": [
                    "h-0140d9a7ecbd102dd"
                ]
            },
            "State": "active",
            "Tags": []
        }
    ]
}
```

在示例输出中，实例位于与事件窗口关联的专属主机上。

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》中的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceEventWindows。

describe-instance-image-metadata

以下代码示例演示了如何使用 describe-instance-image-metadata。

Amazon CLI

示例 1：描述所有实例的 AMI 元数据

以下 describe-instance-image-metadata 示例描述您的 Amazon 账户在指定区域中拥有的所有实例的 AMI 元数据。

```
aws ec2 describe-instance-image-metadata \
    --region us-east-1
```

输出：

```
{
    "InstanceImageMetadata": [
        {
            "InstanceId": "i-1234567890EXAMPLE",
            "InstanceType": "t2.micro",
            "LaunchTime": "2024-08-28T11:25:45+00:00",
            "AvailabilityZone": "us-east-1a",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "OwnerId": "123412341234",
            "Tags": [
                {
                    "Key": "MyTagName",
                    "Value": "my-tag-value"
                }
            ],
            "ImageMetadata": {
                "ImageId": "ami-0b752bf1df193a6c4",
                "Name": "al2023-ami-2023.5.20240819.0-kernel-6.1-x86_64",
                "OwnerId": "137112412989",
                "State": "available",
                "ImageOwnerAlias": "amazon",
                "CreationDate": "2023-01-25T17:20:40Z",
                "DeprecationTime": "2025-01-25T17:20:40Z",
                "IsPublic": true
            }
        }
    ],
    "NextToken": "...EXAMPLEwIAABAA2JHaFxLnEXAMPLE..."
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 中的亚马逊机器映像。

示例 2：描述指定实例的 AMI 元数据

以下 describe-instance-image-metadata 示例描述指定实例的 AMI 元数据。

```
aws ec2 describe-instance-image-metadata \
    --region us-east-1 \
    --instance-ids i-1234567890EXAMPLE i-0987654321EXAMPLE
```

输出：

```
{
    "InstanceImageMetadata": [
        {
            "InstanceId": "i-1234567890EXAMPLE",
            "InstanceType": "t2.micro",
            "LaunchTime": "2024-08-28T11:25:45+00:00",
            "AvailabilityZone": "us-east-1a",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "OwnerId": "123412341234",
            "Tags": [
                {
                    "Key": "MyTagName",
                    "Value": "my-tag-value"
                }
            ],
            "ImageMetadata": {
                "ImageId": "ami-0b752bf1df193a6c4",
                "Name": "al2023-ami-2023.5.20240819.0-kernel-6.1-x86_64",
                "OwnerId": "137112412989",
                "State": "available",
                "ImageOwnerAlias": "amazon",
                "CreationDate": "2023-01-25T17:20:40Z",
                "DeprecationTime": "2025-01-25T17:20:40Z",
                "IsPublic": true
            }
        },
        {
            "InstanceId": "i-0987654321EXAMPLE",
            "InstanceType": "t2.micro",
            "LaunchTime": "2024-08-28T11:25:45+00:00",
```

```
            "AvailabilityZone": "us-east-1a",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "OwnerId": "123412341234",
            "Tags": [
                {
                    "Key": "MyTagName",
                    "Value": "my-tag-value"
                }
            ],
            "ImageMetadata": {
                "ImageId": "ami-0b752bf1df193a6c4",
                "Name": "al2023-ami-2023.5.20240819.0-kernel-6.1-x86_64",
                "OwnerId": "137112412989",
                "State": "available",
                "ImageOwnerAlias": "amazon",
                "CreationDate": "2023-01-25T17:20:40Z",
                "DeprecationTime": "2025-01-25T17:20:40Z",
                "IsPublic": true
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 中的亚马逊机器映像。

示例 3：描述基于筛选条件的实例的 AMI 元数据

以下 `describe-instance-image-metadata` 示例描述 `us-east-1a` 可用区中 `t2.nano` 和 `t2.micro` 实例的 AMI 元数据。

```
aws ec2 describe-instance-image-metadata \
    --region us-east-1 \
    --filters Name=availability-zone,Values=us-east-1a Name=instance-
type,Values=t2.nano,t2.micro
```

输出：

```
{
    "InstanceImageMetadata": [
        {
```

```
            "InstanceId": "i-1234567890EXAMPLE",
            "InstanceType": "t2.micro",
            "LaunchTime": "2024-08-28T11:25:45+00:00",
            "AvailabilityZone": "us-east-1a",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "OwnerId": "123412341234",
            "Tags": [
                {
                    "Key": "MyTagName",
                    "Value": "my-tag-value"
                }
            ],
            "ImageMetadata": {
                "ImageId": "ami-0b752bf1df193a6c4",
                "Name": "al2023-ami-2023.5.20240819.0-kernel-6.1-x86_64",
                "OwnerId": "137112412989",
                "State": "available",
                "ImageOwnerAlias": "amazon",
                "CreationDate": "2023-01-25T17:20:40Z",
                "DeprecationTime": "2025-01-25T17:20:40Z",
                "IsPublic": true
            }
        },
        {
            "InstanceId": "i-0987654321EXAMPLE",
            "InstanceType": "t2.micro",
            "LaunchTime": "2024-08-28T11:25:45+00:00",
            "AvailabilityZone": "us-east-1a",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "OwnerId": "123412341234",
            "Tags": [
                {
                    "Key": "MyTagName",
                    "Value": "my-tag-value"
                }
            ],
            "ImageMetadata": {
                "ImageId": "ami-0b752bf1df193a6c4",
```

```
                "Name": "al2023-ami-2023.5.20240819.0-kernel-6.1-x86_64",
                "OwnerId": "137112412989",
                "State": "available",
                "ImageOwnerAlias": "amazon",
                "CreationDate": "2023-01-25T17:20:40Z",
                "DeprecationTime": "2025-01-25T17:20:40Z",
                "IsPublic": true
            }
        }
    ],
    "NextToken": "...EXAMPLEV7ixRYHwIAABAA2JHaFxLnDAzpatfEXAMPLE..."
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 中的亚马逊机器映像。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstanceImageMetadata。

describe-instance-status

以下代码示例演示了如何使用 describe-instance-status。

Amazon CLI

描述实例的状态

以下 describe-instance-status 示例描述了指定实例的当前状态。

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "InstanceStatuses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "AvailabilityZone": "us-east-1d",
            "SystemStatus": {
```

```
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            },
            "InstanceStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的监控实例状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceStatus。

describe-instance-topology

以下代码示例演示了如何使用 describe-instance-topology。

Amazon CLI

描述所有实例的实例拓扑

以下 describe-instance-topology 示例描述了与此命令支持的实例类型相匹配的所有实例的拓扑。

```
aws ec2 describe-instance-topology \
    --region us-west-2
```

输出：

```
{
    "Instances": [
```

```
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "my-ml-cpg",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "p4d.24xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
                "nn-1212121212example",
                "nn-1211122211example",
                "nn-1311133311example"
            ],
            "ZoneId": "usw2-az4",
            "AvailabilityZone": "us-west-2d"
        },
        {
            "InstanceId": "i-444444444example",
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-5434334334example",
                "nn-1235301234example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
```

```
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

有关更多信息（包括更多示例），请参阅《Amazon EC2 用户指南》中的 Amazon EC2 实例拓扑。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceTopology。

describe-instance-type-offerings

以下代码示例演示了如何使用 describe-instance-type-offerings。

Amazon CLI

示例 1：列出区域中提供的实例类型

以下 describe-instance-type-offerings 示例列出了配置为 Amazon CLI 的默认区域的区域中提供的实例类型。

```
aws ec2 describe-instance-type-offerings
```

要列出其他区域中提供的实例类型，请使用 --region 参数指定区域。

```
aws ec2 describe-instance-type-offerings \
    --region us-east-2
```

输出：

```
{
  "InstanceTypeOfferings": [
      {
          "InstanceType": "m5.2xlarge",
          "LocationType": "region",
          "Location": "us-east-2"
      },
      {
          "InstanceType": "t3.micro",
          "LocationType": "region",
          "Location": "us-east-2"
      },
```

```
        ...
    ]
}
```

示例 2：列出可用区中提供的实例类型

以下 describe-instance-type-offerings 示例列出了指定可用区中提供的实例类型。可用区必须位于指定区域。

```
aws ec2 describe-instance-type-offerings \
    --location-type availability-zone \
    --filters Name=location,Values=us-east-2a \
    --region us-east-2
```

示例 3：检查是否支持某个实例类型

以下 describe-instance-type-offerings 命令指示指定区域是否支持 c5.xlarge 实例类型。

```
aws ec2 describe-instance-type-offerings \
    --filters Name=instance-type,Values=c5.xlarge \
    --region us-east-2
```

以下 describe-instance-type-offerings 示例列出了指定区域支持的所有 C5 实例类型。

```
aws ec2 describe-instance-type-offerings \
    --filters Name=instance-type,Values=c5* \
    --query "InstanceTypeOfferings[].InstanceType" \
    --region us-east-2
```

输出：

```
[
    "c5d.12xlarge",
    "c5d.9xlarge",
    "c5n.xlarge",
    "c5.xlarge",
    "c5d.metal",
    "c5n.metal",
    "c5.large",
    "c5d.2xlarge",
    "c5n.4xlarge",
```

```
    "c5.2xlarge",
    "c5n.large",
    "c5n.9xlarge",
    "c5d.large",
    "c5.18xlarge",
    "c5d.18xlarge",
    "c5.12xlarge",
    "c5n.18xlarge",
    "c5.metal",
    "c5d.4xlarge",
    "c5.24xlarge",
    "c5d.xlarge",
    "c5n.2xlarge",
    "c5d.24xlarge",
    "c5.9xlarge",
    "c5.4xlarge"
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceTypeOfferings。

describe-instance-types

以下代码示例演示了如何使用 describe-instance-types。

Amazon CLI

示例 1：描述实例类型

以下 describe-instance-types 示例显示指定实例类型的详细信息。

```
aws ec2 describe-instance-types \
    --instance-types t2.micro
```

输出：

```
{
    "InstanceTypes": [
        {
            "InstanceType": "t2.micro",
            "CurrentGeneration": true,
            "FreeTierEligible": true,
            "SupportedUsageClasses": [
                "on-demand",
```

```
            "spot"
        ],
        "SupportedRootDeviceTypes": [
            "ebs"
        ],
        "BareMetal": false,
        "Hypervisor": "xen",
        "ProcessorInfo": {
            "SupportedArchitectures": [
                "i386",
                "x86_64"
            ],
            "SustainedClockSpeedInGhz": 2.5
        },
        "VCpuInfo": {
            "DefaultVCpus": 1,
            "DefaultCores": 1,
            "DefaultThreadsPerCore": 1,
            "ValidCores": [
                1
            ],
            "ValidThreadsPerCore": [
                1
            ]
        },
        "MemoryInfo": {
            "SizeInMiB": 1024
        },
        "InstanceStorageSupported": false,
        "EbsInfo": {
            "EbsOptimizedSupport": "unsupported",
            "EncryptionSupport": "supported"
        },
        "NetworkInfo": {
            "NetworkPerformance": "Low to Moderate",
            "MaximumNetworkInterfaces": 2,
            "Ipv4AddressesPerInterface": 2,
            "Ipv6AddressesPerInterface": 2,
            "Ipv6Supported": true,
            "EnaSupport": "unsupported"
        },
        "PlacementGroupInfo": {
            "SupportedStrategies": [
                "partition",
```

```
                    "spread"
                ]
            },
            "HibernationSupported": false,
            "BurstablePerformanceSupported": true,
            "DedicatedHostsSupported": false,
            "AutoRecoverySupported": true
        }
    ]
}
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的实例类型。

示例 2：筛选可用的实例类型

可以指定筛选器，将结果范围限定为具有特定特征的实例类型。以下 `describe-instance-types` 示例列出支持休眠的实例类型。

```
aws ec2 describe-instance-types \
    --filters Name=hibernation-supported,Values=true --query
 'InstanceTypes[*].InstanceType'
```

输出：

```
[
    "m5.8xlarge",
    "r3.large",
    "c3.8xlarge",
    "r5.large",
    "m4.4xlarge",
    "c4.large",
    "m5.xlarge",
    "m4.xlarge",
    "c3.large",
    "c4.8xlarge",
    "c4.4xlarge",
    "c5.xlarge",
    "c5.12xlarge",
    "r5.4xlarge",
    "c5.4xlarge"
]
```

有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的实例类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceTypes。

describe-instances

以下代码示例演示了如何使用 describe-instances。

Amazon CLI

示例 1：描述实例

以下 describe-instances 示例描述了指定的实例。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0abcdef1234567890",
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "t3.nano",
                    "KeyName": "my-key-pair",
                    "LaunchTime": "2022-11-15T10:48:59+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-2a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10-0-0-157",
                    "ProductCodes": [],
```

```
                    "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                    "PublicIpAddress": "34.253.223.13",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2022-11-15T10:49:00+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-02e6ccdca7de29cf2"
                            }
                        }
                    ],
                    "ClientToken": "1234abcd-1234-abcd-1234-d46a8903e9bc",
                    "EbsOptimized": true,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "IamInstanceProfile": {
                        "Arn": "arn:aws:iam::111111111111:instance-profile/
AmazonSSMRoleForInstancesQuickSetup",
                        "Id": "11111111111111111111"
                    },
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                "PublicIp": "34.253.223.13"
                            },
                            "Attachment": {
                                "AttachTime": "2022-11-15T10:48:59+00:00",
                                "AttachmentId": "eni-attach-1234567890abcdefg",
                                "DeleteOnTermination": true,
                                "DeviceIndex": 0,
```

```
                            "Status": "attached",
                            "NetworkCardIndex": 0
                        },
                        "Description": "",
                        "Groups": [
                            {
                                "GroupName": "launch-wizard-146",
                                "GroupId": "sg-1234567890abcdefg"
                            }
                        ],
                        "Ipv6Addresses": [],
                        "MacAddress": "00:11:22:33:44:55",
                        "NetworkInterfaceId": "eni-1234567890abcdefg",
                        "OwnerId": "104024344472",
                        "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                        "PrivateIpAddress": "10-0-0-157",
                        "PrivateIpAddresses": [
                            {
                                "Association": {
                                    "IpOwnerId": "amazon",
                                    "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                    "PublicIp": "34.253.223.13"
                                },
                                "Primary": true,
                                "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                                "PrivateIpAddress": "10-0-0-157"
                            }
                        ],
                        "SourceDestCheck": true,
                        "Status": "in-use",
                        "SubnetId": "subnet-1234567890abcdefg",
                        "VpcId": "vpc-1234567890abcdefg",
                        "InterfaceType": "interface"
                    }
                ],
                "RootDeviceName": "/dev/xvda",
                "RootDeviceType": "ebs",
                "SecurityGroups": [
                    {
                        "GroupName": "launch-wizard-146",
                        "GroupId": "sg-1234567890abcdefg"
```

```
                    }
                ],
                "SourceDestCheck": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": "my-instance"
                    }
                ],
                "VirtualizationType": "hvm",
                "CpuOptions": {
                    "CoreCount": 1,
                    "ThreadsPerCore": 2
                },
                "CapacityReservationSpecification": {
                    "CapacityReservationPreference": "open"
                },
                "HibernationOptions": {
                    "Configured": false
                },
                "MetadataOptions": {
                    "State": "applied",
                    "HttpTokens": "optional",
                    "HttpPutResponseHopLimit": 1,
                    "HttpEndpoint": "enabled",
                    "HttpProtocolIpv6": "disabled",
                    "InstanceMetadataTags": "enabled"
                },
                "EnclaveOptions": {
                    "Enabled": false
                },
                "PlatformDetails": "Linux/UNIX",
                "UsageOperation": "RunInstances",
                "UsageOperationUpdateTime": "2022-11-15T10:48:59+00:00",
                "PrivateDnsNameOptions": {
                    "HostnameType": "ip-name",
                    "EnableResourceNameDnsARecord": true,
                    "EnableResourceNameDnsAAAARecord": false
                },
                "MaintenanceOptions": {
                    "AutoRecovery": "default"
                }
            }
        ],
```

```
            "OwnerId": "111111111111",
            "ReservationId": "r-1234567890abcdefg"
        }
    ]
}
```

示例 2：筛选具有指定类型的实例

以下 describe-instances 示例使用筛选器，将结果范围限定为指定类型的实例。

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=m5.large
```

有关示例输出，请参阅示例 1。

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 CLI 列出和筛选。

示例 3：筛选具有指定类型和可用区的实例

以下 describe-instances 示例使用多个筛选器，将结果范围限定为同样位于指定可用区且具有指定类型的实例。

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-
zone,Values=us-east-2c
```

有关示例输出，请参阅示例 1。

示例 4：使用 JSON 文件筛选具有指定类型和可用区的实例

以下 describe-instances 示例使用 JSON 输入文件执行与上一个示例相同的筛选。当筛选器变得更加复杂时，可以更容易地在 JSON 文件中加以指定。

```
aws ec2 describe-instances \
    --filters file://filters.json
```

的内容filters.json：

```
[
    {
```

```
        "Name": "instance-type",
        "Values": ["t2.micro", "t3.micro"]
    },
    {
        "Name": "availability-zone",
        "Values": ["us-east-2c"]
    }
]
```

有关示例输出，请参阅示例 1。

示例 5：筛选具有指定 Owner 标签的实例

以下 `describe-instances` 示例使用标签筛选器，将结果范围限定为具有指定标签键（Owner）标签的实例，无论标签值如何。

```
aws ec2 describe-instances \
    --filters "Name=tag-key,Values=Owner"
```

有关示例输出，请参阅示例 1。

示例 6：筛选具有指定 my-team 标签值的实例

以下 `describe-instances` 示例使用标签筛选器，将结果范围限定为具有指定标签值（my-team）标签的实例，无论标签键如何。

```
aws ec2 describe-instances \
    --filters "Name=tag-value,Values=my-team"
```

有关示例输出，请参阅示例 1。

示例 7：筛选具有指定 Owner 标签和 my-team 值的实例

以下 `describe-instances` 示例使用标签筛选器，将结果范围限定为具有指定标签值（Owner=my-team）的实例。

```
aws ec2 describe-instances \
    --filters "Name=tag:Owner,Values=my-team"
```

有关示例输出，请参阅示例 1。

示例 8：仅显示所有实例的实例和子网 ID

以下 `describe-instances` 示例使用 `--query` 参数，以 JSON 格式仅显示所有实例的实例和子网 ID。

Linux 和 macOS：

```
aws ec2 describe-instances \
    --query 'Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}' \
    --output json
```

Windows:

```
aws ec2 describe-instances ^
    --query "Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}" ^
    --output json
```

输出：

```
[
    {
        "Instance": "i-057750d42936e468a",
        "Subnet": "subnet-069beee9b12030077"
    },
    {
        "Instance": "i-001efd250faaa6ffa",
        "Subnet": "subnet-0b715c6b7db68927a"
    },
    {
        "Instance": "i-027552a73f021f3bd",
        "Subnet": "subnet-0250c25a1f4e15235"
    }
    ...
]
```

示例 9：筛选指定类型的实例并仅显示其实例 ID

以下 `describe-instances` 示例使用筛选器，将结果范围限定为指定类型的实例，并使用 `--query` 参数仅显示实例 ID。

```
aws ec2 describe-instances \
```

```
    --filters "Name=instance-type,Values=t2.micro" \
    --query "Reservations[*].Instances[*].[InstanceId]" \
    --output text
```

输出：

```
i-031c0dc19de2fb70c
i-00d8bff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c
```

示例 10：筛选指定类型的实例，并仅显示其实例 ID、可用区和指定的标签值

以下 `describe-instances` 示例以表格格式显示实例的实例 ID、可用区和 `Name` 标签值，这些实例有一个名为 `tag-key` 的标签。

Linux 和 macOS：

```
aws ec2 describe-instances \
    --filters Name=tag-key,Values=Name \
    --query 'Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key==`Name`]|
[0].Value}' \
    --output table
```

Windows:

```
aws ec2 describe-instances ^
    --filters Name=tag-key,Values=Name ^
    --query "Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key=='Name']|
[0].Value}" ^
    --output table
```

输出：

```
-------------------------------------------------------------
|                    DescribeInstances                      |
```

```
+--------------+-----------------------+--------------------+
|      AZ      |       Instance        |        Name        |
+--------------+-----------------------+--------------------+
|  us-east-2b  |  i-057750d42936e468a  |  my-prod-server    |
|  us-east-2a  |  i-001efd250faaa6ffa  |  test-server-1     |
|  us-east-2a  |  i-027552a73f021f3bd  |  test-server-2     |
+--------------+-----------------------+--------------------+
```

示例 11：描述分区置放群组中的实例

以下 describe-instances 示例描述了指定的实例。输出包括实例的置放信息，其中包含实例的置放群组名称和分区编号。

```
aws ec2 describe-instances \
    --instance-ids i-0123a456700123456 \
    --query "Reservations[*].Instances[*].Placement"
```

输出：

```
[
    [
        {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 3,
            "Tenancy": "default"
        }

    ]
]
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的描述置放群组中的实例。

示例 12：筛选具有指定置放群组和分区编号的实例

以下 describe-instances 示例将结果筛选为仅具有指定置放群组和分区编号的实例。

```
aws ec2 describe-instances \
    --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-
partition-number,Values=7"
```

以下仅显示输出中的相关信息。

```
"Instances": [
    {
        "InstanceId": "i-0123a456700123456",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
        }
    },
    {
        "InstanceId": "i-9876a543210987654",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
        }
    ],
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的描述置放群组中的实例。

示例 13：筛选配置为允许从实例元数据访问标签的实例

以下 `describe-instances` 示例将结果筛选为，仅显示配置为允许从实例元数据访问实例标签的实例。

```
aws ec2 describe-instances \
    --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \
    --query "Reservations[*].Instances[*].InstanceId" \
    --output text
```

预期的输出如下所示。

```
i-1234567890abcdefg
i-abcdefg1234567890
i-11111111aaaaaaaaa
i-aaaaaaaa111111111
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用实例元数据中的实例标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstances。

`describe-internet-gateways`

以下代码示例演示了如何使用 `describe-internet-gateways`。

Amazon CLI

描述互联网网关

以下 `describe-internet-gateways` 示例描述了指定的互联网网关。

```
aws ec2 describe-internet-gateways \
    --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

输出：

```
{
    "InternetGateways": [
        {
            "Attachments": [
                {
                    "State": "available",
                    "VpcId": "vpc-0a60eb65b4EXAMPLE"
                }
            ],
            "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",
            "OwnerId": "123456789012",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-igw"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的互联网网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInternetGateways。

describe-ipam-pools

以下代码示例演示了如何使用 describe-ipam-pools。

Amazon CLI

查看 IPAM 池的详细信息

以下 describe-ipam-pools 示例显示了池的详细信息。

（Linux）：

```
aws ec2 describe-ipam-pools \
    --filters Name=owner-id,Values=123456789012 Name=ipam-scope-id,Values=ipam-scope-02fc38cd4c48e7d38
```

（Windows）：

```
aws ec2 describe-ipam-pools ^
    --filters Name=owner-id,Values=123456789012 Name=ipam-scope-id,Values=ipam-scope-02fc38cd4c48e7d38
```

输出：

```
{
    "IpamPools": [
        {
            "OwnerId": "123456789012",
            "IpamPoolId": "ipam-pool-02ec043a19bbe5d08",
            "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-pool-02ec043a19bbe5d08",
            "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-scope-02fc38cd4c48e7d38",
            "IpamScopeType": "private",
            "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
            "IpamRegion": "us-east-1",
            "Locale": "None",
            "PoolDepth": 1,
            "State": "create-complete",
            "AutoImport": true,
            "AddressFamily": "ipv4",
            "AllocationMinNetmaskLength": 16,
```

```
            "AllocationMaxNetmaskLength": 26,
            "AllocationDefaultNetmaskLength": 24,
            "AllocationResourceTags": [
                {
                    "Key": "Environment",
                    "Value": "Preprod"
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Preprod pool"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpamPools。

describe-ipam-resource-discoveries

以下代码示例演示了如何使用 describe-ipam-resource-discoveries。

Amazon CLI

示例 1：查看资源发现的完整详细信息

在此示例中，您是一名委派 IPAM 管理员，希望创建资源发现并与另一个 Amazon 组织中的 IPAM 管理员共享资源发现，以便管理员可以管理和监控组织中资源的 IP 地址。

在下列情况中，此示例可能很有用：

您尝试创建资源发现，但出现错误，提示您已达到 1 的上限。您意识到自己可能已经创建了资源发现，并希望在自己的账户中查看它。您在某个区域中拥有未被 IPAM 发现的资源。您希望查看为资源定义的 --operating-regions，并确保已将正确的区域添加为运营区域，以便可以发现其中的资源。

以下 describe-ipam-resource-discoveries 示例列出了 Amazon 账户中资源发现的详细信息。每个 Amazon 区域均可拥有一个资源发现。

```
aws ec2 describe-ipam-resource-discoveries \
```

```
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscoveries": [
        {
            "OwnerId": "149977607591",
            "IpamResourceDiscoveryId": "ipam-res-disco-0f8bdee9067137c0d",
            "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0f8bdee9067137c0d",
            "IpamResourceDiscoveryRegion": "us-east-1",
            "OperatingRegions": [
                {
                    "RegionName": "us-east-1"
                }
            ],
            "IsDefault": false,
            "State": "create-complete",
            "Tags": []
    }
]
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

示例 2：仅查看资源发现 ID

以下 `describe-ipam-resource-discoveries` 示例列出了 Amazon 账户中资源发现的 ID。每个 Amazon 区域均可拥有一个资源发现。

```
aws ec2 describe-ipam-resource-discoveries \
    --query "IpamResourceDiscoveries[*].IpamResourceDiscoveryId" \
    --output text
```

输出：

```
ipam-res-disco-0481e39b242860333
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpamResourceDiscoveries。

describe-ipam-resource-discovery-associations

以下代码示例演示了如何使用 describe-ipam-resource-discovery-associations。

Amazon CLI

查看与 IPAM 关联的所有资源发现

在此示例中，您是 IPAM 委派管理员，已将资源发现与您的 IPAM 相关联，以将其他账户与您的 IPAM 集成。您已经注意到，您的 IPAM 未按预期在资源发现的运营区域中发现资源。您需要检查资源发现的状态和状况，以确保创建它的账户仍然处于活动状态，并且资源发现仍在共享中。

--region 必须是 IPAM 的主区域。

以下 describe-ipam-resource-discovery-associations 示例列出了 Amazon 账户中的资源发现关联。

```
aws ec2 describe-ipam-resource-discovery-associations \
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscoveryAssociations": [
        {
            "OwnerId": "320805250157",
            "IpamResourceDiscoveryAssociationId": "ipam-res-disco-
assoc-05e6b45eca5bf5cf7",
            "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::320805250157:ipam-
resource-discovery-association/ipam-res-disco-assoc-05e6b45eca5bf5cf7",
            "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",
            "IpamId": "ipam-005f921c17ebd5107",
            "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
            "IpamRegion": "us-east-1",
            "IsDefault": true,
            "ResourceDiscoveryStatus": "active",
            "State": "associate-complete",
            "Tags": []
        },
```

```
        {
            "OwnerId": "149977607591",
            "IpamResourceDiscoveryAssociationId": "ipam-res-disco-
assoc-0dfd21ae189ab5f62",
            "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::149977607591:ipam-
resource-discovery-association/ipam-res-disco-assoc-0dfd21ae189ab5f62",
            "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
            "IpamId": "ipam-005f921c17ebd5107",
            "IpamArn": "arn:aws:ec2::149977607591:ipam/ipam-005f921c17ebd5107",
            "IpamRegion": "us-east-1",
            "IsDefault": false,
            "ResourceDiscoveryStatus": "active",
            "State": "create-complete",
            "Tags": []
        }
    ]
}
```

在此示例中，运行此命令后，您注意到您有一个非默认资源发现（`"IsDefault": false ``) that is ``"ResourceDiscoveryStatus": "not-found"` 和 `"State": "create-complete"`）。资源发现拥有者的账户已关闭。如果在另一种情况下，您注意到 `"ResourceDiscoveryStatus": "not-found"` 和 `"State": "associate-complete"`，则表示发生了以下情况之一：

资源发现已被资源发现拥有者删除。资源发现拥有者取消了资源发现的共享。

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpamResourceDiscoveryAssociations。

`describe-ipam-scopes`

以下代码示例演示了如何使用 `describe-ipam-scopes`。

Amazon CLI

查看 IPAM 范围的详细信息

以下 `describe-ipam-scopes` 示例显示了范围的详细信息。

```
aws ec2 describe-ipam-scopes \
```

```
    --filters Name=owner-id,Values=123456789012 Name=ipam-
id,Values=ipam-08440e7a3acde3908
```

输出：

```
{
    "IpamScopes": [
        {
            "OwnerId": "123456789012",
            "IpamScopeId": "ipam-scope-02fc38cd4c48e7d38",
            "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
            "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
            "IpamRegion": "us-east-1",
            "IpamScopeType": "private",
            "IsDefault": true,
            "PoolCount": 2,
            "State": "create-complete",
            "Tags": []
        },
        {
            "OwnerId": "123456789012",
            "IpamScopeId": "ipam-scope-0b9eed026396dbc16",
            "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-0b9eed026396dbc16",
            "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
            "IpamRegion": "us-east-1",
            "IpamScopeType": "public",
            "IsDefault": true,
            "PoolCount": 0,
            "State": "create-complete",
            "Tags": []
        },
        {
            "OwnerId": "123456789012",
            "IpamScopeId": "ipam-scope-0f1aff29486355c22",
            "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-0f1aff29486355c22",
            "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
            "IpamRegion": "us-east-1",
            "IpamScopeType": "private",
            "IsDefault": false,
            "Description": "Example description",
```

```
            "PoolCount": 0,
            "State": "create-complete",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Example name value"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpamScopes。

describe-ipams

以下代码示例演示了如何使用 describe-ipams。

Amazon CLI

查看 IPAM 的详细信息

以下 describe-ipams 示例显示了 IPAM 的详细信息。

```
aws ec2 describe-ipams \
    --filters Name=owner-id,Values=123456789012
```

输出：

```
{
    "Ipams": [
        {
            "OwnerId": "123456789012",
            "IpamId": "ipam-08440e7a3acde3908",
            "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
            "IpamRegion": "us-east-1",
            "PublicDefaultScopeId": "ipam-scope-0b9eed026396dbc16",
            "PrivateDefaultScopeId": "ipam-scope-02fc38cd4c48e7d38",
            "ScopeCount": 3,
            "OperatingRegions": [
                {
                    "RegionName": "us-east-1"
                },
```

```
                {
                    "RegionName": "us-east-2"
                },
                {
                    "RegionName": "us-west-1"
                }
            ],
            "State": "create-complete",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "ExampleIPAM"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpams。

describe-ipv6-pools

以下代码示例演示了如何使用 describe-ipv6-pools。

Amazon CLI

描述 IPv6 地址池

以下 describe-ipv6-pools 示例显示了所有 IPv6 地址池的详细信息。

```
aws ec2 describe-ipv6-pools
```

输出：

```
{
    "Ipv6Pools": [
        {
            "PoolId": "ipv6pool-ec2-012345abc12345abc",
            "PoolCidrBlocks": [
                {
                    "Cidr": "2001:db8:123::/48"
                }
            ],
```

```
            "Tags": [
                {
                    "Key": "pool-1",
                    "Value": "public"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIpv6Pools。

describe-key-pairs

以下代码示例演示了如何使用 describe-key-pairs。

Amazon CLI

显示密钥对

以下 describe-key-pairs 示例显示有关指定密钥对的信息。

```
aws ec2 describe-key-pairs \
    --key-names my-key-pair
```

输出：

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0b94643da6EXAMPLE",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
            "KeyName": "my-key-pair",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2022-05-27T21:51:16.000Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的描述公有密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeKeyPairs。

describe-launch-template-versions

以下代码示例演示了如何使用 describe-launch-template-versions。

Amazon CLI

描述启动模板版本

此示例描述了指定启动模板的版本。

命令:

```
aws ec2 describe-launch-template-versions --launch-template-id lt-068f72b72934aff71
```

输出：

```
{
  "LaunchTemplateVersions": [
      {
          "LaunchTemplateId": "lt-068f72b72934aff71",
          "LaunchTemplateName": "Webservers",
          "VersionNumber": 3,
          "CreatedBy": "arn:aws:iam::123456789102:root",
          "LaunchTemplateData": {
              "KeyName": "kp-us-east",
              "ImageId": "ami-6057e21a",
              "InstanceType": "t2.small",
              "NetworkInterfaces": [
                  {
                      "SubnetId": "subnet-7b16de0c",
                      "DeviceIndex": 0,
                      "Groups": [
                          "sg-7c227019"
                      ]
                  }
              ]
          },
          "DefaultVersion": false,
          "CreateTime": "2017-11-20T13:19:54.000Z"
      },
      {
```

```
            "LaunchTemplateId": "lt-068f72b72934aff71",
            "LaunchTemplateName": "Webservers",
            "VersionNumber": 2,
            "CreatedBy": "arn:aws:iam::123456789102:root",
            "LaunchTemplateData": {
                "KeyName": "kp-us-east",
                "ImageId": "ami-6057e21a",
                "InstanceType": "t2.medium",
                "NetworkInterfaces": [
                    {
                        "SubnetId": "subnet-1a2b3c4d",
                        "DeviceIndex": 0,
                        "Groups": [
                            "sg-7c227019"
                        ]
                    }
                ]
            },
            "DefaultVersion": false,
            "CreateTime": "2017-11-20T13:12:32.000Z"
        },
        {
            "LaunchTemplateId": "lt-068f72b72934aff71",
            "LaunchTemplateName": "Webservers",
            "VersionNumber": 1,
            "CreatedBy": "arn:aws:iam::123456789102:root",
            "LaunchTemplateData": {
                "UserData": "",
                "KeyName": "kp-us-east",
                "ImageId": "ami-aabbcc11",
                "InstanceType": "t2.medium",
                "NetworkInterfaces": [
                    {
                        "SubnetId": "subnet-7b16de0c",
                        "DeviceIndex": 0,
                        "DeleteOnTermination": false,
                        "Groups": [
                            "sg-7c227019"
                        ],
                        "AssociatePublicIpAddress": true
                    }
                ]
            },
            "DefaultVersion": true,
```

```
            "CreateTime": "2017-11-20T12:52:33.000Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLaunchTemplateVersions。

describe-launch-templates

以下代码示例演示了如何使用 describe-launch-templates。

Amazon CLI

描述启动模板

此示例描述了您的启动模板。

命令:

```
aws ec2 describe-launch-templates
```

输出：

```
{
    "LaunchTemplates": [
        {
            "LatestVersionNumber": 2,
            "LaunchTemplateId": "lt-0e06d290751193123",
            "LaunchTemplateName": "TemplateForWebServer",
            "DefaultVersionNumber": 2,
            "CreatedBy": "arn:aws:iam::123456789012:root",
            "CreateTime": "2017-11-27T09:30:23.000Z"
        },
        {
            "LatestVersionNumber": 6,
            "LaunchTemplateId": "lt-0c45b5e061ec98456",
            "LaunchTemplateName": "DBServersTemplate",
            "DefaultVersionNumber": 1,
            "CreatedBy": "arn:aws:iam::123456789012:root",
            "CreateTime": "2017-11-20T09:25:22.000Z"
        },
        {
```

```
            "LatestVersionNumber": 1,
            "LaunchTemplateId": "lt-0d47d774e8e52dabc",
            "LaunchTemplateName": "MyLaunchTemplate2",
            "DefaultVersionNumber": 1,
            "CreatedBy": "arn:aws:iam::123456789012:root",
            "CreateTime": "2017-11-02T12:06:21.000Z"
        },
        {
            "LatestVersionNumber": 3,
            "LaunchTemplateId": "lt-01e5f948eb4f589d6",
            "LaunchTemplateName": "testingtemplate2",
            "DefaultVersionNumber": 1,
            "CreatedBy": "arn:aws:sts::123456789012:assumed-role/AdminRole/
i-03ee35176e2e5aabc",
            "CreateTime": "2017-12-01T08:19:48.000Z"
        },
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLaunchTemplates。

describe-local-gateway-route-table-virtual-interface-group-associations

以下代码示例演示了如何使用 describe-local-gateway-route-table-virtual-interface-group-associations。

Amazon CLI

描述虚拟接口组和本地网关路由表之间的关联

以下 describe-local-gateway-route-table-virtual-interface-group-associations 示例描述了您 Amazon 账户中的虚拟接口组和本地网关路由表之间的关联。

```
aws ec2 describe-local-gateway-route-table-virtual-interface-group-associations
```

输出：

```
{
    "LocalGatewayRouteTableVirtualInterfaceGroupAssociations": [
        {
            "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-
grp-assoc-07145b276bEXAMPLE",
```

```
            "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
            "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
            "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
            "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:123456789012:local-
gateway-route-table/lgw-rtb-059615ef7dEXAMPLE",
            "OwnerId": "123456789012",
            "State": "associated",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用本地网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations。

describe-local-gateway-route-table-vpc-associations

以下代码示例演示了如何使用 describe-local-gateway-route-table-vpc-associations。

Amazon CLI

描述 VPC 和本地网关路由表之间的关联

以下 describe-local-gateway-route-table-vpc-associations 示例显示了有关 VPC 和本地网关路由表之间指定关联的信息。

```
aws ec2 describe-local-gateway-route-table-vpc-associations \
    --local-gateway-route-table-vpc-association-ids lgw-vpc-assoc-0e0f27af15EXAMPLE
```

输出：

```
{
    "LocalGatewayRouteTableVpcAssociation": {
        "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-0e0f27af1EXAMPLE",
        "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
        "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
        "VpcId": "vpc-0efe9bde08EXAMPLE",
        "State": "associated"
    }
}
```

有关更多信息，请参阅《Outposts 用户指南》中的本地网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGatewayRouteTableVpcAssociations。

describe-local-gateway-route-tables

以下代码示例演示了如何使用 describe-local-gateway-route-tables。

Amazon CLI

描述本地网关路由表

以下 describe-local-gateway-route-tables 示例显示了有关本地网关路由表的详细信息。

```
aws ec2 describe-local-gateway-route-tables
```

输出：

```
{
    "LocalGatewayRouteTables": [
        {
            "LocalGatewayRouteTableId": "lgw-rtb-059615ef7deEXAMPLE",
            "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
            "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/
op-0dc11b66edEXAMPLE",
            "State": "available"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGatewayRouteTables。

describe-local-gateway-virtual-interface-groups

以下代码示例演示了如何使用 describe-local-gateway-virtual-interface-groups。

Amazon CLI

描述本地网关虚拟接口组

以下 describe-local-gateway-virtual-interface-groups 示例描述了 Amazon 账户中的本地网关虚拟接口组。

```
aws ec2 describe-local-gateway-virtual-interface-groups
```

输出：

```
{
    "LocalGatewayVirtualInterfaceGroups": [
        {
            "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
            "LocalGatewayVirtualInterfaceIds": [
                "lgw-vif-01a23bc4d5EXAMPLE",
                "lgw-vif-543ab21012EXAMPLE"
            ],
            "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
            "OwnerId": "123456789012",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用本地网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGatewayVirtualInterfaceGroups。

describe-local-gateway-virtual-interfaces

以下代码示例演示了如何使用 describe-local-gateway-virtual-interfaces。

Amazon CLI

描述本地网关虚拟接口

以下 describe-local-gateway-virtual-interfaces 示例描述了 Amazon 账户中的本地网关虚拟接口。

```
aws ec2 describe-local-gateway-virtual-interfaces
```

输出：

```
{
    "LocalGatewayVirtualInterfaces": [
        {
            "LocalGatewayVirtualInterfaceId": "lgw-vif-01a23bc4d5EXAMPLE",
            "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
            "Vlan": 2410,
            "LocalAddress": "0.0.0.0/0",
            "PeerAddress": "0.0.0.0/0",
            "LocalBgpAsn": 65010,
            "PeerBgpAsn": 65000,
            "OwnerId": "123456789012",
            "Tags": []
        },
        {
            "LocalGatewayVirtualInterfaceId": "lgw-vif-543ab21012EXAMPLE",
            "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
            "Vlan": 2410,
            "LocalAddress": "0.0.0.0/0",
            "PeerAddress": "0.0.0.0/0",
            "LocalBgpAsn": 65010,
            "PeerBgpAsn": 65000,
            "OwnerId": "123456789012",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用本地网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGatewayVirtualInterfaces。

describe-local-gateways

以下代码示例演示了如何使用 describe-local-gateways。

Amazon CLI

描述本地网关

以下 describe-local-gateways 示例显示可供您使用的本地网关的详细信息。

```
aws ec2 describe-local-gateways
```

输出：

```
{
    "LocalGateways": [
        {
            "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
            "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0dc11b66ed59f995a",
            "OwnerId": "123456789012",
            "State": "available"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLocalGateways。

describe-locked-snapshots

以下代码示例演示了如何使用 describe-locked-snapshots。

Amazon CLI

描述快照的锁定状态

以下 describe-locked-snapshots 示例描述了指定快照的锁定状态。

```
aws ec2 describe-locked-snapshots \
    --snapshot-ids snap-0b5e733b4a8df6e0d
```

输出：

```
{
    "Snapshots": [
        {
            "OwnerId": "123456789012",
            "SnapshotId": "snap-0b5e733b4a8df6e0d",
            "LockState": "governance",
            "LockDuration": 365,
            "LockCreatedOn": "2024-05-05T00:56:06.208000+00:00",
            "LockDurationStartTime": "2024-05-05T00:56:06.208000+00:00",
```

```
            "LockExpiresOn": "2025-05-05T00:56:06.208000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的快照锁定。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLockedSnapshots。

describe-managed-prefix-lists

以下代码示例演示了如何使用 describe-managed-prefix-lists。

Amazon CLI

描述托管前缀列表

以下 describe-managed-prefix-lists 示例描述了 Amazon 账户 123456789012 拥有的前缀列表。

```
aws ec2 describe-managed-prefix-lists \
    --filters Name=owner-id,Values=123456789012
```

输出：

```
{
    "PrefixLists": [
        {
            "PrefixListId": "pl-11223344556677aab",
            "AddressFamily": "IPv6",
            "State": "create-complete",
            "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-11223344556677aab",
            "PrefixListName": "vpc-ipv6-cidrs",
            "MaxEntries": 25,
            "Version": 1,
            "Tags": [],
            "OwnerId": "123456789012"
        },
        {
            "PrefixListId": "pl-0123456abcabcabc1",
            "AddressFamily": "IPv4",
            "State": "active",
```

```
            "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
            "PrefixListName": "vpc-cidrs",
            "MaxEntries": 10,
            "Version": 1,
            "Tags": [],
            "OwnerId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeManagedPrefixLists。

`describe-moving-addresses`

以下代码示例演示了如何使用 `describe-moving-addresses`。

Amazon CLI

描述移动中的地址

此示例描述了所有移动中的弹性 IP 地址。

命令:

```
aws ec2 describe-moving-addresses
```

输出：

```
{
    "MovingAddressStatuses": [
        {
            "PublicIp": "198.51.100.0",
            "MoveStatus": "MovingToVpc"
        }
    ]
}
```

此示例描述了正在移动到 EC2-VPC 平台的所有地址。

命令:

```
aws ec2 describe-moving-addresses --filters Name=moving-status,Values=MovingToVpc
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMovingAddresses。

describe-nat-gateways

以下代码示例演示了如何使用 describe-nat-gateways。

Amazon CLI

示例 1：描述公共 NAT 网关

以下 describe-nat-gateways 示例描述了指定的公共 NAT 网关。

```
aws ec2 describe-nat-gateways \
    --nat-gateway-id nat-01234567890abcdef
```

输出：

```
{
    "NatGateways": [
        {
            "CreateTime": "2023-08-25T01:56:51.000Z",
            "NatGatewayAddresses": [
                {
                    "AllocationId": "eipalloc-0790180cd2EXAMPLE",
                    "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
                    "PrivateIp": "10.0.0.211",
                    "PublicIp": "54.85.121.213",
                    "AssociationId": "eipassoc-04d295cc9b8815b24",
                    "IsPrimary": true,
                    "Status": "succeeded"
                },
                {
                    "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
                    "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
                    "PrivateIp": "10.0.0.74",
                    "PublicIp": "3.211.231.218",
                    "AssociationId": "eipassoc-0f96bdca17EXAMPLE",
                    "IsPrimary": false,
                    "Status": "succeeded"
                }
```

```
            ],
            "NatGatewayId": "nat-01234567890abcdef",
            "State": "available",
            "SubnetId": "subnet-655eab5f08EXAMPLE",
            "VpcId": "vpc-098eb5ef58EXAMPLE",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "public-nat"
                }
            ],
            "ConnectivityType": "public"
        }
    ]
}
```

示例 2：描述私有 NAT 网关

以下 `describe-nat-gateways` 示例描述了指定的私有 NAT 网关。

```
aws ec2 describe-nat-gateways \
    --nat-gateway-id nat-1234567890abcdef0
```

输出：

```
{
    "NatGateways": [
        {
            "CreateTime": "2023-08-25T00:50:05.000Z",
            "NatGatewayAddresses": [
                {
                    "NetworkInterfaceId": "eni-0065a61b324d1897a",
                    "PrivateIp": "10.0.20.240",
                    "IsPrimary": true,
                    "Status": "succeeded"
                },
                {
                    "NetworkInterfaceId": "eni-0065a61b324d1897a",
                    "PrivateIp": "10.0.20.33",
                    "IsPrimary": false,
                    "Status": "succeeded"
                },
                {
```

```
                    "NetworkInterfaceId": "eni-0065a61b324d1897a",
                    "PrivateIp": "10.0.20.197",
                    "IsPrimary": false,
                    "Status": "succeeded"
                }
            ],
            "NatGatewayId": "nat-1234567890abcdef0",
            "State": "available",
            "SubnetId": "subnet-08fc749671EXAMPLE",
            "VpcId": "vpc-098eb5ef58EXAMPLE",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "private-nat"
                }
            ],
            "ConnectivityType": "private"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNatGateways。

describe-network-acls

以下代码示例演示了如何使用 describe-network-acls。

Amazon CLI

描述网络 ACL

以下 describe-network-acls 示例检索有关网络 ACL 的详细信息。

```
aws ec2 describe-network-acls
```

输出：

```
{
    "NetworkAcls": [
        {
            "Associations": [
```

```
                {
                    "NetworkAclAssociationId": "aclassoc-0c1679dc41EXAMPLE",
                    "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
                    "SubnetId": "subnet-0931fc2fa5EXAMPLE"
                }
            ],
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                }
            ],
            "IsDefault": true,
            "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [],
```

```
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "Egress": true,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 101
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "Egress": true,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32768
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "Egress": false,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 101
                },
                {
```

```
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "Egress": false,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32768
                }
            ],
            "IsDefault": true,
            "NetworkAclId": "acl-0e2a78e4e2EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-03914afb3eEXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的网络 ACL。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkAcls。

describe-network-insights-access-scope-analyses

以下代码示例演示了如何使用 describe-network-insights-access-scope-analyses。

Amazon CLI

描述 Network Insights 访问范围分析

以下 describe-network-insights-access-scope-analyses 示例描述了您 Amazon 账户中的访问范围分析。

```
aws ec2 describe-network-insights-access-scope-analyses \
    --region us-east-1
```

输出：

```
{
    "NetworkInsightsAccessScopeAnalyses": [
        {
            "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789111",
            "NetworkInsightsAccessScopeAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope-analysis/nisa-123456789111",
            "NetworkInsightsAccessScopeId": "nis-123456789222",
            "Status": "succeeded",
            "StartDate": "2022-01-25T19:45:36.842000+00:00",
            "FindingsFound": "true",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInsightsAccessScopeAnalyses。

describe-network-insights-access-scopes

以下代码示例演示了如何使用 describe-network-insights-access-scopes。

Amazon CLI

描述 Network Insights 访问范围

以下 describe-network-insights-access-scopes 示例描述了您 Amazon 账户中的访问范围分析。

```
aws ec2 describe-network-insights-access-scopes \
    --region us-east-1
```

输出：

```
{
    "NetworkInsightsAccessScopes": [
        {
            "NetworkInsightsAccessScopeId": "nis-123456789111",
            "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope/nis-123456789111",
```

```
            "CreatedDate": "2021-11-29T21:12:41.416000+00:00",
            "UpdatedDate": "2021-11-29T21:12:41.416000+00:00",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInsightsAccessScopes。

describe-network-insights-analyses

以下代码示例演示了如何使用 describe-network-insights-analyses。

Amazon CLI

查看路径分析的结果

以下 describe-network-insights-analyses 示例描述了指定的分析。在此示例中，源是互联网网关，目标是 EC2 实例，协议是 TCP。分析成功（Status 为 succeeded），路径无法访问（NetworkPathFound 为 false）。说明代码 ENI_SG_RULES_MISMATCH 表示实例的安全组不包含允许目标端口上的流量的规则。

```
aws ec2 describe-network-insights-analyses \
    --network-insights-analysis-ids nia-02207aa13eb480c7a
```

输出：

```
{
    "NetworkInsightsAnalyses": [
        {
            "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a",
            "NetworkInsightsAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-analysis/nia-02207aa13eb480c7a",
            "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
            "StartDate": "2021-01-20T22:58:37.495Z",
            "Status": "succeeded",
            "NetworkPathFound": false,
            "Explanations": [
                {
```

```
                    "Direction": "ingress",
                    "ExplanationCode": "ENI_SG_RULES_MISMATCH",
                    "NetworkInterface": {
                        "Id": "eni-0a25edef15a6cc08c",
                        "Arn": "arn:aws:ec2:us-east-1:123456789012:network-
interface/eni-0a25edef15a6cc08c"
                    },
                    "SecurityGroups": [
                        {
                            "Id": "sg-02f0d35a850ba727f",
                            "Arn": "arn:aws:ec2:us-east-1:123456789012:security-
group/sg-02f0d35a850ba727f"
                        }
                    ],
                    "Subnet": {
                        "Id": "subnet-004ff41eccb4d1194",
                        "Arn": "arn:aws:ec2:us-east-1:123456789012:subnet/
subnet-004ff41eccb4d1194"
                    },
                    "Vpc": {
                        "Id": "vpc-f1663d98ad28331c7",
                        "Arn": "arn:aws:ec2:us-east-1:123456789012:vpc/vpc-
f1663d98ad28331c7"
                    }
                }
            ],
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInsightsAnalyses。

`describe-network-insights-paths`

以下代码示例演示了如何使用 `describe-network-insights-paths`。

Amazon CLI

描述路径

以下 `describe-network-insights-paths` 示例描述了指定的路径。

```
aws ec2 describe-network-insights-paths \
    --network-insights-path-ids nip-0b26f224f1d131fa8
```

输出：

```
{
    "NetworkInsightsPaths": [
        {
            "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
            "NetworkInsightsPathArn": "arn:aws:ec2:us-east-1:123456789012:network-
insights-path/nip-0b26f224f1d131fa8",
            "CreatedDate": "2021-01-20T22:43:46.933Z",
            "Source": "igw-0797cccdc9d73b0e5",
            "Destination": "i-0495d385ad28331c7",
            "Protocol": "tcp"
        }
    ]
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInsightsPaths。

describe-network-interface-attribute

以下代码示例演示了如何使用 describe-network-interface-attribute。

Amazon CLI

描述网络接口的连接属性

此示例命令描述了指定网络接口的 attachment 属性。

命令:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --
attribute attachment
```

输出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
```

```
  "Attachment": {
      "Status": "attached",
      "DeviceIndex": 0,
      "AttachTime": "2015-05-21T20:02:20.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "DeleteOnTermination": true,
      "AttachmentId": "eni-attach-43348162",
      "InstanceOwnerId": "123456789012"
  }
}
```

描述网络接口的描述属性

此示例命令描述了指定网络接口的 description 属性。

命令:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --
attribute description
```

输出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Description": {
      "Value": "My description"
  }
}
```

描述网络接口的 groupSet 属性

此示例命令描述了指定网络接口的 groupSet 属性。

命令:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --
attribute groupSet
```

输出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Groups": [
```

```
        {
            "GroupName": "my-security-group",
            "GroupId": "sg-903004f8"
        }
    ]
}
```

描述网络接口的 sourceDestCheck 属性

此示例命令描述了指定网络接口的 sourceDestCheck 属性。

命令:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --
attribute sourceDestCheck
```

输出：

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "SourceDestCheck": {
      "Value": true
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInterfaceAttribute。

describe-network-interface-permissions

以下代码示例演示了如何使用 describe-network-interface-permissions。

Amazon CLI

描述网络接口权限

此示例描述了您的所有网络接口权限。

命令:

```
aws ec2 describe-network-interface-permissions
```

输出：

```
{
  "NetworkInterfacePermissions": [
      {
          "PermissionState": {
              "State": "GRANTED"
          },
          "NetworkInterfacePermissionId": "eni-perm-06fd19020ede149ea",
          "NetworkInterfaceId": "eni-b909511a",
          "Permission": "INSTANCE-ATTACH",
          "AwsAccountId": "123456789012"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInterfacePermissions。

`describe-network-interfaces`

以下代码示例演示了如何使用 `describe-network-interfaces`。

Amazon CLI

描述网络接口

此示例描述了您的所有网络接口。

命令:

```
aws ec2 describe-network-interfaces
```

输出：

```
{
  "NetworkInterfaces": [
      {
          "Status": "in-use",
          "MacAddress": "02:2f:8f:b0:cf:75",
          "SourceDestCheck": true,
          "VpcId": "vpc-a01106c2",
          "Description": "my network interface",
          "Association": {
```

```
            "PublicIp": "203.0.113.12",
            "AssociationId": "eipassoc-0fbb766a",
            "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
            "IpOwnerId": "123456789012"
        },
        "NetworkInterfaceId": "eni-e5aa89a3",
        "PrivateIpAddresses": [
            {
                "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
                "Association": {
                    "PublicIp": "203.0.113.12",
                    "AssociationId": "eipassoc-0fbb766a",
                    "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
                    "IpOwnerId": "123456789012"
                },
                "Primary": true,
                "PrivateIpAddress": "10.0.1.17"
            }
        ],
        "RequesterManaged": false,
        "Ipv6Addresses": [],
        "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
        "AvailabilityZone": "us-east-1d",
        "Attachment": {
            "Status": "attached",
            "DeviceIndex": 1,
            "AttachTime": "2013-11-30T23:36:42.000Z",
            "InstanceId": "i-1234567890abcdef0",
            "DeleteOnTermination": false,
            "AttachmentId": "eni-attach-66c4350a",
            "InstanceOwnerId": "123456789012"
        },
        "Groups": [
            {
                "GroupName": "default",
                "GroupId": "sg-8637d3e3"
            }
        ],
        "SubnetId": "subnet-b61f49f0",
        "OwnerId": "123456789012",
        "TagSet": [],
        "PrivateIpAddress": "10.0.1.17"
    },
    {
```

```
            "Status": "in-use",
            "MacAddress": "02:58:f5:ef:4b:06",
            "SourceDestCheck": true,
            "VpcId": "vpc-a01106c2",
            "Description": "Primary network interface",
            "Association": {
                "PublicIp": "198.51.100.0",
                "IpOwnerId": "amazon"
            },
            "NetworkInterfaceId": "eni-f9ba99bf",
            "PrivateIpAddresses": [
                {
                    "Association": {
                        "PublicIp": "198.51.100.0",
                        "IpOwnerId": "amazon"
                    },
                    "Primary": true,
                    "PrivateIpAddress": "10.0.1.149"
                }
            ],
            "RequesterManaged": false,
            "Ipv6Addresses": [],
            "AvailabilityZone": "us-east-1d",
            "Attachment": {
                "Status": "attached",
                "DeviceIndex": 0,
                "AttachTime": "2013-11-30T23:35:33.000Z",
                "InstanceId": "i-0598c7d356eba48d7",
                "DeleteOnTermination": true,
                "AttachmentId": "eni-attach-1b9db777",
                "InstanceOwnerId": "123456789012"
            },
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-8637d3e3"
                }
            ],
            "SubnetId": "subnet-b61f49f0",
            "OwnerId": "123456789012",
            "TagSet": [],
            "PrivateIpAddress": "10.0.1.149"
        }
    ]
```

```
}
```

此示例描述了具有带有键 Purpose 和值 Prod 的标签的网络接口。

命令:

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

输出：

```
{
  "NetworkInterfaces": [
      {
          "Status": "available",
          "MacAddress": "12:2c:bd:f9:bf:17",
          "SourceDestCheck": true,
          "VpcId": "vpc-8941ebec",
          "Description": "ProdENI",
          "NetworkInterfaceId": "eni-b9a5ac93",
          "PrivateIpAddresses": [
              {
                  "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
                  "Primary": true,
                  "PrivateIpAddress": "10.0.1.55"
              },
              {
                  "PrivateDnsName": "ip-10-0-1-117.ec2.internal",
                  "Primary": false,
                  "PrivateIpAddress": "10.0.1.117"
              }
          ],
          "RequesterManaged": false,
          "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
          "AvailabilityZone": "us-east-1d",
          "Ipv6Addresses": [],
          "Groups": [
              {
                  "GroupName": "MySG",
                  "GroupId": "sg-905002f5"
              }
          ],
          "SubnetId": "subnet-31d6c219",
          "OwnerId": "123456789012",
```

```
            "TagSet": [
                {
                    "Value": "Prod",
                    "Key": "Purpose"
                }
            ],
            "PrivateIpAddress": "10.0.1.55"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNetworkInterfaces。

`describe-placement-groups`

以下代码示例演示了如何使用 `describe-placement-groups`。

Amazon CLI

描述置放群组

此示例命令描述了所有置放群组。

命令:

```
aws ec2 describe-placement-groups
```

输出：

```
{
    "PlacementGroups": [
        {
            "GroupName": "my-cluster",
            "State": "available",
            "Strategy": "cluster"
        },
        ...
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePlacementGroups。

describe-prefix-lists

以下代码示例演示了如何使用 describe-prefix-lists。

Amazon CLI

描述前缀列表

此示例列出了该区域的所有可用前缀列表。

命令:

```
aws ec2 describe-prefix-lists
```

输出：

```
{
  "PrefixLists": [
    {
      "PrefixListName": "com.amazonaws.us-east-1.s3",
      "Cidrs": [
        "54.231.0.0/17"
      ],
      "PrefixListId": "pl-63a5400a"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePrefixLists。

describe-principal-id-format

以下代码示例演示了如何使用 describe-principal-id-format。

Amazon CLI

描述已启用长 ID 格式的 IAM 用户和角色的 ID 格式

以下 describe-principal-id-format 示例描述了已启用长 ID 格式的根用户、所有 IAM 角色和所有 IAM 用户的 ID 格式。

```
aws ec2 describe-principal-id-format \
```

```
    --resource instance
```

输出：

```
{
    "Principals": [
        {
            "Arn": "arn:aws:iam::123456789012:root",
            "Statuses": [
                {
                    "Deadline": "2016-12-15T00:00:00.000Z",
                    "Resource": "reservation",
                    "UseLongIds": true
                },
                {
                    "Deadline": "2016-12-15T00:00:00.000Z",
                    "Resource": "instance",
                    "UseLongIds": true
                },
                {
                    "Deadline": "2016-12-15T00:00:00.000Z",
                    "Resource": "volume",
                    "UseLongIds": true
                },
            ]
        },
        ...
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePrincipalIdFormat。

describe-public-ipv4-pools

以下代码示例演示了如何使用 describe-public-ipv4-pools。

Amazon CLI

描述公有 IPv4 地址池

以下 describe-public-ipv4-pools 示例显示了有关使用自带 IP 地址（BYOIP）预置公有 IPv4 地址范围时创建的地址池的详细信息。

```
aws ec2 describe-public-ipv4-pools
```

输出：

```
{
    "PublicIpv4Pools": [
        {
            "PoolId": "ipv4pool-ec2-1234567890abcdef0",
            "PoolAddressRanges": [
                {
                    "FirstAddress": "203.0.113.0",
                    "LastAddress": "203.0.113.255",
                    "AddressCount": 256,
                    "AvailableAddressCount": 256
                }
            ],
            "TotalAddressCount": 256,
            "TotalAvailableAddressCount": 256
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePublicIpv4Pools。

describe-regions

以下代码示例演示了如何使用 describe-regions。

Amazon CLI

示例 1：描述所有已启用的区域

以下 describe-regions 示例描述了为您的账户启用的所有区域。

```
aws ec2 describe-regions
```

输出：

```
{
    "Regions": [
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
```

```
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
```

```
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的区域和区。

示例 2：描述具有端点的已启用区域，其名称包含特定字符串

以下 describe-regions 示例描述了在端点中具有字符串“us”的所有已启用区域。

```
aws ec2 describe-regions \
    --filters "Name=endpoint,Values=*us*"
```

输出：

```
{
    "Regions": [
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的区域和区。

示例 3：描述所有区域

以下 describe-regions 示例描述了所有可用区域，包括已禁用的区域。

```
aws ec2 describe-regions \
    --all-regions
```

输出：

```
{
    "Regions": [
```

```
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.me-south-1.amazonaws.com",
            "RegionName": "me-south-1",
            "OptInStatus": "not-opted-in"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
            "OptInStatus": "opt-in-not-required"
```

```
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-east-1.amazonaws.com",
            "RegionName": "ap-east-1",
            "OptInStatus": "not-opted-in"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1",
```

```
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的区域和区。

示例 4：仅列出区域名称

以下 `describe-regions` 示例使用 `--query` 参数筛选输出，并仅以文本形式返回区域的名称。

```
aws ec2 describe-regions \
    --all-regions \
    --query "Regions[].{Name:RegionName}" \
    --output text
```

输出：

```
eu-north-1
ap-south-1
eu-west-3
eu-west-2
eu-west-1
ap-northeast-3
ap-northeast-2
me-south-1
ap-northeast-1
sa-east-1
ca-central-1
ap-east-1
ap-southeast-1
ap-southeast-2
eu-central-1
us-east-1
us-east-2
us-west-1
us-west-2
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的区域和区。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRegions。

describe-replace-root-volume-tasks

以下代码示例演示了如何使用 describe-replace-root-volume-tasks。

Amazon CLI

示例 1：查看有关特定根卷替换任务的信息

以下 describe-replace-root-volume-tasks 示例描述了根卷替换任务 replacevol-0111122223333abcd。

```
aws ec2 describe-replace-root-volume-tasks \
    --replace-root-volume-task-ids replacevol-0111122223333abcd
```

输出：

```
{
    "ReplaceRootVolumeTasks": [
        {
            "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",
            "Tags": [],
            "InstanceId": "i-0123456789abcdefa",
            "TaskState": "succeeded",
            "StartTime": "2022-03-14T15:16:28Z",
            "CompleteTime": "2022-03-14T15:16:52Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的替换根卷。

示例 2：查看有关特定实例的所有根卷替换任务的信息

以下 describe-replace-root-volume-tasks 示例描述了实例 i-0123456789abcdefa 的所有根卷替换任务。

```
aws ec2 describe-replace-root-volume-tasks \
```

```
    --filters Name=instance-id,Values=i-0123456789abcdefa
```

输出：

```
{
    "ReplaceRootVolumeTasks": [
        {
            "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",
            "Tags": [],
            "InstanceId": "i-0123456789abcdefa",
            "TaskState": "succeeded",
            "StartTime": "2022-03-14T15:06:38Z",
            "CompleteTime": "2022-03-14T15:07:03Z"
        },
        {
            "ReplaceRootVolumeTaskId": "replacevol-0444455555555abcd",
            "Tags": [],
            "InstanceId": "i-0123456789abcdefa",
            "TaskState": "succeeded",
            "StartTime": "2022-03-14T15:16:28Z",
            "CompleteTime": "2022-03-14T15:16:52Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的替换根卷。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplaceRootVolumeTasks。

describe-reserved-instances-listings

以下代码示例演示了如何使用 describe-reserved-instances-listings。

Amazon CLI

描述预留实例列示

以下 describe-reserved-instances-listings 示例检索有关指定预留实例列示的信息。

```
aws ec2 describe-reserved-instances-listings \
    --reserved-instances-listing-id 5ec28771-05ff-4b9b-aa31-9e57dexample
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedInstancesListings。

describe-reserved-instances-modifications

以下代码示例演示了如何使用 describe-reserved-instances-modifications。

Amazon CLI

描述预留实例修改

此示例命令描述了为您的账户提交的所有预留实例修改请求。

命令:

```
aws ec2 describe-reserved-instances-modifications
```

输出：

```
{
    "ReservedInstancesModifications": [
        {
            "Status": "fulfilled",
            "ModificationResults": [
                {
                    "ReservedInstancesId": "93bbbca2-62f1-4d9d-b225-16bada29e6c7",
                    "TargetConfiguration": {
                        "AvailabilityZone": "us-east-1b",
                        "InstanceType": "m1.large",
                        "InstanceCount": 3
                    }
                },
                {
                     "ReservedInstancesId": "1ba8e2e3-aabb-46c3-bcf5-3fe2fda922e6",
                     "TargetConfiguration": {
                         "AvailabilityZone": "us-east-1d",
                         "InstanceType": "m1.xlarge",
                         "InstanceCount": 1
                     }
                 }
```

```
            ],
            "EffectiveDate": "2015-08-12T17:00:00.000Z",
            "CreateDate": "2015-08-12T17:52:52.630Z",
            "UpdateDate": "2015-08-12T18:08:06.698Z",
            "ClientToken": "c9adb218-3222-4889-8216-0cf0e52dc37e:
            "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-
ab31-0f13aaf46687",
            "ReservedInstancesIds": [
                {
                    "ReservedInstancesId": "b847fa93-e282-4f55-b59a-1342f5bd7c02"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedInstancesModifications。

describe-reserved-instances-offerings

以下代码示例演示了如何使用 describe-reserved-instances-offerings。

Amazon CLI

描述预留实例产品

此示例命令描述了该区域中可供购买的所有预留实例。

命令:

```
aws ec2 describe-reserved-instances-offerings
```

输出：

```
{
  "ReservedInstancesOfferings": [
      {
          "OfferingType": "Partial Upfront",
          "AvailabilityZone": "us-east-1b",
          "InstanceTenancy": "default",
          "PricingDetails": [],
```

```
            "ProductDescription": "Red Hat Enterprise Linux",
            "UsagePrice": 0.0,
            "RecurringCharges": [
                {
                    "Amount": 0.088,
                    "Frequency": "Hourly"
                }
            ],
            "Marketplace": false,
            "CurrencyCode": "USD",
            "FixedPrice": 631.0,
            "Duration": 94608000,
            "ReservedInstancesOfferingId": "9a06095a-bdc6-47fe-a94a-2a382f016040",
            "InstanceType": "c1.medium"
        },
        {
            "OfferingType": "PartialUpfront",
            "AvailabilityZone": "us-east-1b",
            "InstanceTenancy": "default",
            "PricingDetails": [],
            "ProductDescription": "Linux/UNIX",
            "UsagePrice": 0.0,
            "RecurringCharges": [
                {
                    "Amount": 0.028,
                    "Frequency": "Hourly"
                }
            ],
            "Marketplace": false,
            "CurrencyCode": "USD",
            "FixedPrice": 631.0,
            "Duration": 94608000,
            "ReservedInstancesOfferingId": "bfbefc6c-0d10-418d-b144-7258578d329d",
            "InstanceType": "c1.medium"
        },
    ...
}
```

使用选项描述预留实例产品

此示例列出了 Amazon 提供的具有以下规格的预留实例：t1.micro 实例类型，Windows（Amazon VPC）产品，以及高利用率产品。

命令:

```
aws ec2 describe-reserved-instances-offerings --no-include-marketplace --instance-type "t1.micro" --product-description "Windows (Amazon VPC)" --offering-type "no upfront"
```

输出：

```
{
    "ReservedInstancesOfferings": [
      {
          "OfferingType": "No Upfront",
          "AvailabilityZone": "us-east-1b",
          "InstanceTenancy": "default",
          "PricingDetails": [],
          "ProductDescription": "Windows",
          "UsagePrice": 0.0,
          "RecurringCharges": [
              {
                  "Amount": 0.015,
                  "Frequency": "Hourly"
              }
          ],
          "Marketplace": false,
          "CurrencyCode": "USD",
          "FixedPrice": 0.0,
          "Duration": 31536000,
          "ReservedInstancesOfferingId": "c48ab04c-fe69-4f94-8e39-a23842292823",
          "InstanceType": "t1.micro"
      },

              ...
      {
          "OfferingType": "No Upfront",
          "AvailabilityZone": "us-east-1d",
          "InstanceTenancy": "default",
          "PricingDetails": [],
          "ProductDescription": "Windows (Amazon VPC)",
          "UsagePrice": 0.0,
          "RecurringCharges": [
              {
                  "Amount": 0.015,
                  "Frequency": "Hourly"
              }
          ],
```

```
            "Marketplace": false,
            "CurrencyCode": "USD",
            "FixedPrice": 0.0,
            "Duration": 31536000,
            "ReservedInstancesOfferingId": "3a98bf7d-2123-42d4-b4f5-8dbec4b06dc6",
            "InstanceType": "t1.micro"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedInstancesOfferings。

describe-reserved-instances

以下代码示例演示了如何使用 describe-reserved-instances。

Amazon CLI

描述预留实例

此示例命令描述了您拥有的预留实例。

命令:

```
aws ec2 describe-reserved-instances
```

输出：

```
{
  "ReservedInstances": [
      {
          "ReservedInstancesId": "b847fa93-e282-4f55-b59a-1342fexample",
          "OfferingType": "No Upfront",
          "AvailabilityZone": "us-west-1c",
          "End": "2016-08-14T21:34:34.000Z",
          "ProductDescription": "Linux/UNIX",
          "UsagePrice": 0.00,
          "RecurringCharges": [
              {
                  "Amount": 0.104,
                  "Frequency": "Hourly"
              }
```

```
            ],
            "Start": "2015-08-15T21:34:35.086Z",
            "State": "active",
            "FixedPrice": 0.0,
            "CurrencyCode": "USD",
            "Duration": 31536000,
            "InstanceTenancy": "default",
            "InstanceType": "m3.medium",
            "InstanceCount": 2
        },
        ...
    ]
}
```

使用筛选条件描述预留实例

此示例筛选了响应，以仅包含 us-west-1c 中三年期 t2.micro Linux/UNIX 预留实例。

命令:

```
aws ec2 describe-reserved-instances --
filters Name=duration,Values=94608000 Name=instance-
type,Values=t2.micro Name=product-description,Values=Linux/UNIX Name=availability-
zone,Values=us-east-1e
```

输出：

```
{
    "ReservedInstances": [
        {
            "ReservedInstancesId": "f127bd27-edb7-44c9-a0eb-0d7e09259af0",
            "OfferingType": "All Upfront",
            "AvailabilityZone": "us-east-1e",
            "End": "2018-03-26T21:34:34.000Z",
            "ProductDescription": "Linux/UNIX",
            "UsagePrice": 0.00,
            "RecurringCharges": [],
            "Start": "2015-03-27T21:34:35.848Z",
            "State": "active",
            "FixedPrice": 151.0,
            "CurrencyCode": "USD",
            "Duration": 94608000,
            "InstanceTenancy": "default",
```

```
            "InstanceType": "t2.micro",
            "InstanceCount": 1
        }
    ]
}
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的“使用 Amazon EC2 实例”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedInstances。

describe-route-tables

以下代码示例演示了如何使用 describe-route-tables。

Amazon CLI

描述路由表

以下 describe-route-tables 示例检索有关路由表的详细信息。

```
aws ec2 describe-route-tables
```

输出：

```
{
    "RouteTables": [
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-0df3f54e06EXAMPLE",
                    "RouteTableId": "rtb-09ba434c1bEXAMPLE"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-09ba434c1bEXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
```

```
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "NatGatewayId": "nat-06c018cbd8EXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "blackhole"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-9EXAMPLE",
                    "RouteTableId": "rtb-a1eec7de"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-a1eec7de",
            "Routes": [
                {
                    "DestinationCidrBlock": "172.31.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-fEXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": false,
                    "RouteTableAssociationId": "rtbassoc-0b100c28b2EXAMPLE",
```

```
                    "RouteTableId": "rtb-07a98f76e5EXAMPLE",
                    "SubnetId": "subnet-0d3d002af8EXAMPLE"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-07a98f76e5EXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-06cf664d80EXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRouteTables。

`describe-scheduled-instance-availability`

以下代码示例演示了如何使用 `describe-scheduled-instance-availability`。

Amazon CLI

描述有效的计划

此示例描述了从指定日期开始每周星期日发生的计划。

命令:

```
aws ec2 describe-scheduled-instance-availability --
recurrence Frequency=Weekly,Interval=1,OccurrenceDays=[1] --first-slot-start-time-
range EarliestTime=2016-01-31T00:00:00Z,LatestTime=2016-01-31T04:00:00Z
```

输出：

```
{
  "ScheduledInstanceAvailabilitySet": [
    {
        "AvailabilityZone": "us-west-2b",
        "TotalScheduledInstanceHours": 1219,
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "MinTermDurationInDays": 366,
        "AvailableInstanceCount": 20,
        "Recurrence": {
            "OccurrenceDaySet": [
                1
            ],
            "Interval": 1,
            "Frequency": "Weekly",
            "OccurrenceRelativeToEnd": false
        },
        "Platform": "Linux/UNIX",
        "FirstSlotStartTime": "2016-01-31T00:00:00Z",
        "MaxTermDurationInDays": 366,
        "SlotDurationInHours": 23,
        "NetworkPlatform": "EC2-VPC",
        "InstanceType": "c4.large",
        "HourlyPrice": "0.095"
    },
    ...
  ]
}
```

要缩小结果范围，您可以添加筛选条件，用于指定操作系统、网络和实例类型。

命令:

--filters Name=platform,Values=Linux/UNIX Name=network-platform,Values=EC2-VPC Name=instance-type,Values=c4.large

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledInstanceAvailability。

describe-scheduled-instances

以下代码示例演示了如何使用 describe-scheduled-instances。

Amazon CLI

描述计划实例

此示例描述了指定的计划实例。

命令:

```
aws ec2 describe-scheduled-instances --scheduled-instance-
ids sci-1234-1234-1234-1234-123456789012
```

输出：

```
{
  "ScheduledInstanceSet": [
      {
          "AvailabilityZone": "us-west-2b",
          "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
          "HourlyPrice": "0.095",
          "CreateDate": "2016-01-25T21:43:38.612Z",
          "Recurrence": {
              "OccurrenceDaySet": [
                  1
              ],
              "Interval": 1,
              "Frequency": "Weekly",
              "OccurrenceRelativeToEnd": false,
              "OccurrenceUnit": ""
          },
          "Platform": "Linux/UNIX",
          "TermEndDate": "2017-01-31T09:00:00Z",
          "InstanceCount": 1,
          "SlotDurationInHours": 32,
          "TermStartDate": "2016-01-31T09:00:00Z",
          "NetworkPlatform": "EC2-VPC",
          "TotalScheduledInstanceHours": 1696,
          "NextSlotStartTime": "2016-01-31T09:00:00Z",
          "InstanceType": "c4.large"
      }
```

```
  ]
}
```

此示例描述了您的所有计划实例。

命令:

```
aws ec2 describe-scheduled-instances
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledInstances。

describe-security-group-references

以下代码示例演示了如何使用 describe-security-group-references。

Amazon CLI

描述安全组引用

此示例描述了 sg-bbbb2222 的安全组引用。该响应表示安全组 sg-bbbb2222 正在被 VPC vpc-aaaaaaaa 中的安全组引用。

命令:

```
aws ec2 describe-security-group-references --group-id sg-bbbbb22222
```

输出：

```
{
  "SecurityGroupsReferenceSet": [
    {
      "ReferencingVpcId": "vpc-aaaaaaaa ",
      "GroupId": "sg-bbbbb22222",
      "VpcPeeringConnectionId": "pcx-b04deed9"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecurityGroupReferences。

describe-security-group-rules

以下代码示例演示了如何使用 describe-security-group-rules。

Amazon CLI

示例 1：描述安全组的安全组规则

以下 describe-security-group-rules 示例描述了指定安全组的安全组规则。使用 filters 选项将结果范围限定为特定的安全组。

```
aws ec2 describe-security-group-rules \
    --filters Name="group-id",Values="sg-1234567890abcdef0"
```

输出：

```
{
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-abcdef01234567890",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "111122223333",
            "IsEgress": false,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "ReferencedGroupInfo": {
                "GroupId": "sg-1234567890abcdef0",
                "UserId": "111122223333"
            },
            "Tags": []
        },
        {
            "SecurityGroupRuleId": "sgr-bcdef01234567890a",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "111122223333",
            "IsEgress": true,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv6": "::/0",
            "Tags": []
        },
```

```
        {
            "SecurityGroupRuleId": "sgr-cdef01234567890ab",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "111122223333",
            "IsEgress": true,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv4": "0.0.0.0/0",
            "Tags": []
        }
    ]
}
```

示例 2：描述安全组规则

以下 `describe-security-group-rules` 示例描述了指定的安全组规则。

```
aws ec2 describe-security-group-rules \
    --security-group-rule-ids sgr-cdef01234567890ab
```

输出：

```
{
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-cdef01234567890ab",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "111122223333",
            "IsEgress": true,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv4": "0.0.0.0/0",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅 Amazon VPC 用户指南中的安全组规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecurityGroupRules。

describe-security-group-vpc-associations

以下代码示例演示了如何使用 describe-security-group-vpc-associations。

Amazon CLI

描述 VPC 关联

以下 describe-security-group-vpc-associations 示例描述了指定安全组的 VPC 关联。

```
aws ec2 describe-security-group-vpc-associations \
    --filters Name=group-id,Values=sg-04dbb43907d3f8a78
```

输出：

```
{
    "SecurityGroupVpcAssociations": [
        {
            "GroupId": "sg-04dbb43907d3f8a78",
            "VpcId": "vpc-0bf4c2739bc05a694",
            "VpcOwnerId": "123456789012",
            "State": "associated"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的将安全组与多个 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeSecurityGroupVpcAssociations。

describe-security-groups

以下代码示例演示了如何使用 describe-security-groups。

Amazon CLI

示例 1：描述安全组

以下 describe-security-groups 示例描述了指定的安全组。

```
aws ec2 describe-security-groups \
    --group-ids sg-903004f8
```

输出：


```
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": [],
                    "PrefixListIds": []
                }
            ],
            "Description": "My security group",
            "Tags": [
                {
                    "Value": "SG1",
                    "Key": "Name"
                }
            ],
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [],
                    "UserIdGroupPairs": [
                        {
                            "UserId": "123456789012",
                            "GroupId": "sg-903004f8"
                        }
                    ],
                    "PrefixListIds": []
                },
                {
                    "PrefixListIds": [],
                    "FromPort": 22,
                    "IpRanges": [
                        {
                            "Description": "Access from NY office",
                            "CidrIp": "203.0.113.0/24"
                        }
```

```
                    ],
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "UserIdGroupPairs": []
                    }
            ],
            "GroupName": "MySecurityGroup",
            "VpcId": "vpc-1a2b3c4d",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8",
        }
    ]
}
```

示例 2：描述具有特定规则的安全组

以下 describe-security-groups 示例使用筛选条件将结果范围限定为具有允许 SSH 流量（端口 22）的规则和允许来自所有地址的流量 (0.0.0.0/0) 的规则的安全组。示例使用 --query 参数仅显示安全组的名称。安全组必须匹配要在结果中返回的所有筛选条件；但是，单个规则不必匹配所有筛选条件。例如，输出返回一个安全组，其中一个规则允许来自特定 IP 地址的 SSH 流量，另一个规则允许来自所有地址的 HTTP 流量。

```
aws ec2 describe-security-groups \
    --filters Name=ip-permission.from-port,Values=22 Name=ip-permission.to-
port,Values=22 Name=ip-permission.cidr,Values='0.0.0.0/0' \
    --query "SecurityGroups[*].[GroupName]" \
    --output text
```

输出：

```
default
my-security-group
web-servers
launch-wizard-1
```

示例 3：根据标签描述安全组

以下 describe-security-groups 示例使用筛选器，将结果范围限定为安全组名称中包含 test 且带有标签 Test=To-delete 的安全组。示例使用 --query 参数仅显示安全组的名称和 ID。

```
aws ec2 describe-security-groups \
```

```
    --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
    --query "SecurityGroups[*].{Name:GroupName,ID:GroupId}"
```

输出：

```
[
    {
        "Name": "testfornewinstance",
        "ID": "sg-33bb22aa"
    },
    {
        "Name": "newgrouptest",
        "ID": "sg-1a2b3c4d"
    }
]
```

有关使用标签筛选器的更多示例，请参阅《Amazon EC2 用户指南》中的使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecurityGroups。

describe-snapshot-attribute

以下代码示例演示了如何使用 describe-snapshot-attribute。

Amazon CLI

描述快照的快照属性

以下 describe-snapshot-attribute 示例列出了与之共享快照的账户。

```
aws ec2 describe-snapshot-attribute \
    --snapshot-id snap-01234567890abcedf \
    --attribute createVolumePermission
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "CreateVolumePermissions": [
        {
            "UserId": "123456789012"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的共享 Amazon EBS 快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshotAttribute。

describe-snapshot-tier-status

以下代码示例演示了如何使用 describe-snapshot-tier-status。

Amazon CLI

查看有关已存档快照的存档信息

以下 describe-snapshot-tier-status 示例提供有关已存档快照的存档信息。

```
aws ec2 describe-snapshot-tier-status \
    --filters "Name=snapshot-id, Values=snap-01234567890abcedf"
```

输出：

```
{
    "SnapshotTierStatuses": [
        {
            "Status": "completed",
            "ArchivalCompleteTime": "2021-09-15T17:33:16.147Z",
            "LastTieringProgress": 100,
            "Tags": [],
            "VolumeId": "vol-01234567890abcedf",
            "LastTieringOperationState": "archival-completed",
            "StorageTier": "archive",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-01234567890abcedf",
            "LastTieringStartTime": "2021-09-15T16:44:37.574Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的查看已存档快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshotTierStatus。

describe-snapshots

以下代码示例演示了如何使用 describe-snapshots。

Amazon CLI

示例 1：描述快照

以下 describe-snapshots 示例描述了指定的快照。

```
aws ec2 describe-snapshots \
    --snapshot-ids snap-1234567890abcdef0
```

输出：

```
{
    "Snapshots": [
        {
            "Description": "This is my snapshot",
            "Encrypted": false,
            "VolumeId": "vol-049df61146c4d7901",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2019-02-28T21:28:32.000Z",
            "Progress": "100%",
            "OwnerId": "012345678910",
            "SnapshotId": "snap-01234567890abcdef",
            "Tags": [
                {
                    "Key": "Stack",
                    "Value": "test"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 快照。

示例 2：描述基于筛选器的快照

以下 describe-snapshots 示例使用筛选器，将结果范围限定为 Amazon 账户拥有的处于 pending 状态的快照。示例使用 --query 参数仅显示快照 ID 和快照的启动时间。

```
aws ec2 describe-snapshots \
    --owner-ids self \
    --filters Name=status,Values=pending \
    --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

输出：

```
[
    {
        "ID": "snap-1234567890abcdef0",
        "Time": "2019-08-04T12:48:18.000Z"
    },
    {
        "ID": "snap-066877671789bd71b",
        "Time": "2019-08-04T02:45:16.000Z
    },
    ...
]
```

以下 describe-snapshots 示例使用筛选器，将结果范围限定为从指定卷创建的快照。示例使用 --query 参数仅显示快照 ID。

```
aws ec2 describe-snapshots \
    --filters Name=volume-id,Values=049df61146c4d7901 \
    --query "Snapshots[*].[SnapshotId]" \
    --output text
```

输出：

```
snap-1234567890abcdef0
snap-08637175a712c3fb9
...
```

有关使用筛选器的其他示例，请参阅《Amazon EC2 用户指南》中的列出和筛选资源。

示例 3：根据标签描述快照

以下 describe-snapshots 示例使用标签筛选器，将结果范围限定为带有标签 Stack=Prod 的快照。

```
aws ec2 describe-snapshots \
```

```
    --filters Name=tag:Stack,Values=prod
```

有关 describe-snapshots 的输出示例，请参阅示例 1。

有关使用标签筛选器的更多示例，请参阅《Amazon EC2 用户指南》中的使用标签。

示例 4：根据期限描述快照

以下 describe-snapshots 示例使用 JMESPath 表达式，来描述 Amazon 账户在指定日期之前创建的所有快照。仅显示快照 ID。

```
aws ec2 describe-snapshots \
    --owner-ids 012345678910 \
    --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

有关使用筛选器的其他示例，请参阅《Amazon EC2 用户指南》中的列出和筛选资源。

示例 5：仅查看存档的快照

以下 describe-snapshots 示例列出归档层中存储的快照。

```
aws ec2 describe-snapshots \
    --filters "Name=storage-tier,Values=archive"
```

输出：

```
{
    "Snapshots": [
        {
            "Description": "Snap A",
            "Encrypted": false,
            "VolumeId": "vol-01234567890aaaaaa",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-09-07T21:00:00.000Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-01234567890aaaaaa",
            "StorageTier": "archive",
            "Tags": []
        },
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的查看已存档快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshots。

describe-spot-datafeed-subscription

以下代码示例演示了如何使用 describe-spot-datafeed-subscription。

Amazon CLI

描述账户的竞价型实例数据源订阅

此示例命令描述了账户的数据源。

命令:

```
aws ec2 describe-spot-datafeed-subscription
```

输出：

```
{
    "SpotDatafeedSubscription": {
        "OwnerId": "123456789012",
        "Prefix": "spotdata",
        "Bucket": "amzn-s3-demo-bucket",
        "State": "Active"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotDatafeedSubscription。

describe-spot-fleet-instances

以下代码示例演示了如何使用 describe-spot-fleet-instances。

Amazon CLI

描述与竞价型实例集关联的竞价型实例

此示例命令列出了与指定竞价型实例集关联的竞价型实例。

命令:

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

输出：

```
{
  "ActiveInstances": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "InstanceType": "m3.medium",
          "SpotInstanceRequestId": "sir-08b93456"
      },
      ...
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotFleetInstances。

describe-spot-fleet-request-history

以下代码示例演示了如何使用 describe-spot-fleet-request-history。

Amazon CLI

描述竞价型实例集历史记录

此示例命令返回从指定时间开始的指定竞价型实例集的历史记录。

命令:

```
aws ec2 describe-spot-fleet-request-history --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --start-time 2015-05-26T00:00:00Z
```

以下示例输出显示了竞价型实例集的两个竞价型实例的成功启动。

输出：

```
{
  "HistoryRecords": [
      {
          "Timestamp": "2015-05-26T23:17:20.697Z",
          "EventInformation": {
              "EventSubType": "submitted"
          },
          "EventType": "fleetRequestChange"
      },
      {
          "Timestamp": "2015-05-26T23:17:20.873Z",
          "EventInformation": {
              "EventSubType": "active"
          },
          "EventType": "fleetRequestChange"
      },
      {
          "Timestamp": "2015-05-26T23:21:21.712Z",
          "EventInformation": {
              "InstanceId": "i-1234567890abcdef0",
              "EventSubType": "launched"
          },
          "EventType": "instanceChange"
      },
      {
          "Timestamp": "2015-05-26T23:21:21.816Z",
          "EventInformation": {
              "InstanceId": "i-1234567890abcdef1",
              "EventSubType": "launched"
          },
          "EventType": "instanceChange"
      }
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
  "NextToken": "CpHNsscimcV5oH7bSbub03CI2Qms5+ypNpNm
+53MNlR0YcXAkp0xFlfKf91yVxSExmbtma3awYxMFzNA663ZskT0AHtJ6TCb2Z8bQC2EnZgyELbymtWPfpZ1ZbauVg
+P+TfGlWxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
  "StartTime": "2015-05-26T00:00:00Z"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotFleetRequestHistory。

describe-spot-fleet-requests

以下代码示例演示了如何使用 describe-spot-fleet-requests。

Amazon CLI

描述竞价型实例集请求

此示例描述了您的所有竞价型实例集请求。

命令:

```
aws ec2 describe-spot-fleet-requests
```

输出：

```
{
  "SpotFleetRequestConfigs": [
      {
          "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
          "SpotFleetRequestConfig": {
              "TargetCapacity": 20,
              "LaunchSpecifications": [
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
                              "AssociatePublicIpAddress": true,
                              "SecondaryPrivateIpAddressCount": 0
                          }
                      ],
                      "InstanceType": "cc2.8xlarge",
                      "ImageId": "ami-1a2b3c4d"
                  },
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
```

```
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "r3.8xlarge",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    },
    {
        "SpotFleetRequestId": "sfr-306341ed-9739-402e-881b-ce47bEXAMPLE",
        "SpotFleetRequestConfig": {
            "TargetCapacity": 20,
            "LaunchSpecifications": [
                {
                    "EbsOptimized": false,
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-6e7f829e",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "m3.medium",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    }
  ]
}
```

描述竞价型实例集请求

此示例描述了指定的竞价型实例集请求。

命令:

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

输出：

```
{
  "SpotFleetRequestConfigs": [
      {
          "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
          "SpotFleetRequestConfig": {
              "TargetCapacity": 20,
              "LaunchSpecifications": [
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
                              "AssociatePublicIpAddress": true,
                              "SecondaryPrivateIpAddressCount": 0
                          }
                      ],
                      "InstanceType": "cc2.8xlarge",
                      "ImageId": "ami-1a2b3c4d"
                  },
                  {
                      "EbsOptimized": false,
                      "NetworkInterfaces": [
                          {
                              "SubnetId": "subnet-a61dafcf",
                              "DeviceIndex": 0,
                              "DeleteOnTermination": false,
                              "AssociatePublicIpAddress": true,
                              "SecondaryPrivateIpAddressCount": 0
                          }
                      ],
                      "InstanceType": "r3.8xlarge",
                      "ImageId": "ami-1a2b3c4d"
                  }
              ],
```

```
                "SpotPrice": "0.05",
                "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
            },
            "SpotFleetRequestState": "active"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotFleetRequests。

describe-spot-instance-requests

以下代码示例演示了如何使用 describe-spot-instance-requests。

Amazon CLI

示例 1：描述竞价型实例请求

以下 describe-spot-instance-requests 示例描述了指定的竞价型实例请求。

```
aws ec2 describe-spot-instance-requests \
    --spot-instance-request-ids sir-08b93456
```

输出：

```
{
    "SpotInstanceRequests": [
        {
            "CreateTime": "2018-04-30T18:14:55.000Z",
            "InstanceId": "i-1234567890abcdef1",
            "LaunchSpecification": {
                "InstanceType": "t2.micro",
                "ImageId": "ami-003634241a8fcdec0",
                "KeyName": "my-key-pair",
                "SecurityGroups": [
                    {
                        "GroupName": "default",
                        "GroupId": "sg-e38f24a7"
                    }
                ],
                "BlockDeviceMappings": [
                    {
```

```
                        "DeviceName": "/dev/sda1",
                        "Ebs": {
                            "DeleteOnTermination": true,
                            "SnapshotId": "snap-0e54a519c999adbbd",
                            "VolumeSize": 8,
                            "VolumeType": "standard",
                            "Encrypted": false
                        }
                    }
                ],
                "NetworkInterfaces": [
                    {
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "SubnetId": "subnet-049df61146c4d7901"
                    }
                ],
                "Placement": {
                    "AvailabilityZone": "us-east-2b",
                    "Tenancy": "default"
                },
                "Monitoring": {
                    "Enabled": false
                }
            },
            "LaunchedAvailabilityZone": "us-east-2b",
            "ProductDescription": "Linux/UNIX",
            "SpotInstanceRequestId": "sir-08b93456",
            "SpotPrice": "0.010000"
            "State": "active",
            "Status": {
                "Code": "fulfilled",
                "Message": "Your Spot request is fulfilled.",
                "UpdateTime": "2018-04-30T18:16:21.000Z"
            },
            "Tags": [],
            "Type": "one-time",
            "InstanceInterruptionBehavior": "terminate"
        }
    ]
}
```

示例 2：描述基于筛选条件的竞价型实例请求

以下 describe-spot-instance-requests 示例使用筛选条件将结果范围限定为在指定可用区中具有指定实例类型的竞价型实例请求。该示例使用 --query 参数仅显示实例 ID。

```
aws ec2 describe-spot-instance-requests \
    --filters Name=launch.instance-type,Values=m3.medium Name=launched-availability-zone,Values=us-east-2a \
    --query "SpotInstanceRequests[*].[InstanceId]" \
    --output text
```

输出：

```
i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...
```

有关使用筛选条件的其他示例，请参阅《Amazon Elastic Compute Cloud 用户指南》中的列出并筛选您的资源。

示例 3：描述基于标签的竞价型实例请求

以下 describe-spot-instance-requests 示例使用标签筛选条件将结果范围限定为具有标签 cost-center=cc123 的竞价型实例请求。

```
aws ec2 describe-spot-instance-requests \
    --filters Name=tag:cost-center,Values=cc123
```

有关 describe-spot-instance-requests 的输出示例，请参阅示例 1。

有关使用标签筛选器的更多示例，请参阅《Amazon EC2 用户指南》中的使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotInstanceRequests。

describe-spot-price-history

以下代码示例演示了如何使用 describe-spot-price-history。

Amazon CLI

描述竞价型实例价格历史记录

此示例命令返回 m1.xlarge 实例在 1 月份特定日期的竞价型实例价格历史记录。

命令:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --start-
time 2014-01-06T07:08:09 --end-time 2014-01-06T08:09:10
```

输出：

```
{
  "SpotPriceHistory": [
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1b"
          },
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1c"
          },
          {
              "Timestamp": "2014-01-06T05:42:36.000Z",
              "ProductDescription": "SUSE Linux (Amazon VPC)",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1a"
      },
      ...
}
```

描述 Linux/UNIX Amazon VPC 的竞价型实例价格历史记录

此示例命令返回 1 月份特定日期的 m1.xlarge、Linux/UNIX Amazon VPC 实例的竞价型实例价格历史记录。

命令:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --product-
description "Linux/UNIX (Amazon VPC)" --start-time 2014-01-06T07:08:09 --end-
time 2014-01-06T08:09:10
```

输出：

```
{
  "SpotPriceHistory": [
      {
          "Timestamp": "2014-01-06T04:32:53.000Z",
          "ProductDescription": "Linux/UNIX (Amazon VPC)",
          "InstanceType": "m1.xlarge",
          "SpotPrice": "0.080000",
          "AvailabilityZone": "us-west-1a"
      },
      {
          "Timestamp": "2014-01-05T11:28:26.000Z",
          "ProductDescription": "Linux/UNIX (Amazon VPC)",
          "InstanceType": "m1.xlarge",
          "SpotPrice": "0.080000",
          "AvailabilityZone": "us-west-1c"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSpotPriceHistory。

describe-stale-security-groups

以下代码示例演示了如何使用 describe-stale-security-groups。

Amazon CLI

描述过时的安全组

此示例描述了 vpc-11223344 的过时安全组规则。响应显示，您账户中的 sg-5fa68d3a 具有引用对等 VPC 中的 sg-279ab042 的过时入口 SSH 规则，而您账户中的 sg-fe6fba9a 具有引用对等 VPC 中的 sg-ef6fba8b 的过时出口 SSH 规则。

命令:

```
aws ec2 describe-stale-security-groups --vpc-id vpc-11223344
```

输出：

```
{
  "StaleSecurityGroupSet": [
      {
          "VpcId": "vpc-11223344",
          "StaleIpPermissionsEgress": [
              {
                  "ToPort": 22,
                  "FromPort": 22,
                  "UserIdGroupPairs": [
                      {
                          "VpcId": "vpc-7a20e51f",
                          "GroupId": "sg-ef6fba8b",
                          "VpcPeeringConnectionId": "pcx-b04deed9",
                          "PeeringStatus": "active"
                      }
                  ],
                  "IpProtocol": "tcp"
              }
          ],
          "GroupName": "MySG1",
          "StaleIpPermissions": [],
          "GroupId": "sg-fe6fba9a",
          "Description": MySG1"
      },
      {
          "VpcId": "vpc-11223344",
          "StaleIpPermissionsEgress": [],
          "GroupName": "MySG2",
          "StaleIpPermissions": [
              {
                  "ToPort": 22,
                  "FromPort": 22,
                  "UserIdGroupPairs": [
                      {
                          "VpcId": "vpc-7a20e51f",
                          "GroupId": "sg-279ab042",
                          "Description": "Access from pcx-b04deed9",
                          "VpcPeeringConnectionId": "pcx-b04deed9",
                          "PeeringStatus": "active"
```

```
                    }
                ],
                "IpProtocol": "tcp"
            }
        ],
        "GroupId": "sg-5fa68d3a",
        "Description": "MySG2"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStaleSecurityGroups。

describe-store-image-tasks

以下代码示例演示了如何使用 describe-store-image-tasks。

Amazon CLI

描述 AMI 存储任务的进度

以下 describe-store-image-tasks 示例描述了 AMI 存储任务的进度。

```
aws ec2 describe-store-image-tasks
```

输出：

```
{
    "StoreImageTaskResults": [
        {
            "AmiId": "ami-1234567890abcdef0",
            "Bucket": "my-ami-bucket",
            "ProgressPercentage": 17,
            "S3objectKey": "ami-1234567890abcdef0.bin",
            "StoreTaskState": "InProgress",
            "StoreTaskFailureReason": null,
            "TaskStartTime": "2022-01-01T01:01:01.001Z"
        }
    ]
}
```

有关使用 S3 存储和还原 AMI 的更多信息，请参阅《Amazon EC2 用户指南》中的“使用 S3 存储和还原 AMI”<https://docs.aws.amazon.com/AmazonEC2/latest/UserGuide/ami-store-restore.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStoreImageTasks。

describe-subnets

以下代码示例演示了如何使用 describe-subnets。

Amazon CLI

示例 1：描述所有子网

以下 describe-subnets 示例显示子网的详细信息。

```
aws ec2 describe-subnets
```

输出：

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": false,
            "MapCustomerOwnedIpOnLaunch": true,
            "State": "available",
            "SubnetId": "subnet-0bb1c79de3EXAMPLE",
            "VpcId": "vpc-0ee975135dEXAMPLE",
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "CustomerOwnedIpv4Pool:": 'pool-2EXAMPLE',
            "SubnetArn": "arn:aws:ec2:us-east-2:111122223333:subnet/
subnet-0bb1c79de3EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
```

```
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 VPC 和子网。

示例 2：描述特定 VPC 的子网

以下 `describe-subnets` 示例使用筛选条件检索指定 VPC 的子网的详细信息。

```
aws ec2 describe-subnets \
```

```
    --filters "Name=vpc-id,Values=vpc-3EXAMPLE"
```

输出：

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 VPC 和子网。

示例 3：描述带有特定标签的子网

以下 describe-subnets 示例使用筛选器检索带有标签 CostCenter=123 的子网的详细信息，并使用 --query 参数显示带有此标签子网的子网 ID。

```
aws ec2 describe-subnets \
    --filters "Name=tag:CostCenter,Values=123" \
    --query "Subnets[*].SubnetId" \
    --output text
```

输出：

```
subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 VPC 和子网。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSubnets。

describe-tags

以下代码示例演示了如何使用 describe-tags。

Amazon CLI

示例 1：描述单个资源的所有标签

以下 describe-tags 示例描述了指定实例的标签。

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

输出：

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
```

```
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Beta Server",
            "Key": "Name"
        }
    ]
}
```

示例 2：描述资源类型的所有标签

以下 describe-tags 示例描述了卷的标签。

```
aws ec2 describe-tags \
    --filters "Name=resource-type,Values=volume"
```

输出：

```
{
    "Tags": [
        {
            "ResourceType": "volume",
            "ResourceId": "vol-1234567890abcdef0",
            "Value": "Project1",
            "Key": "Purpose"
        },
        {
            "ResourceType": "volume",
            "ResourceId": "vol-049df61146c4d7901",
            "Value": "Logs",
            "Key": "Purpose"
        }
    ]
}
```

示例 3：描述所有标签

以下 describe-tags 示例描述了所有资源的标签。

```
aws ec2 describe-tags
```

示例 4：根据标签键描述资源的标签

以下 describe-tags 示例描述了具有键 Stack 的标签的资源的标签。

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack
```

输出：

```
{
    "Tags": [
        {
            "ResourceType": "volume",
            "ResourceId": "vol-027552a73f021f3b",
            "Value": "Production",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        }
    ]
}
```

示例 5：根据标签键和标签值描述资源的标签

以下 describe-tags 示例描述了具有标签 Stack=Test 的资源的标签。

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack Name=value,Values=Test
```

输出：

```
{
    "Tags": [
        {
            "ResourceType": "image",
            "ResourceId": "ami-3ac336533f021f3bd",
            "Value": "Test",
```

```
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        }
    ]
}
```

以下 describe-tags 示例使用替代语法来描述具有标签 Stack=Test 的资源。

```
aws ec2 describe-tags \
    --filters "Name=tag:Stack,Values=Test"
```

以下 describe-tags 示例描述了具有键为 Purpose 但没有值的标签的所有实例的标签。

```
aws ec2 describe-tags \
    --filters "Name=resource-
type,Values=instance" "Name=key,Values=Purpose" "Name=value,Values="
```

输出：

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef5",
            "Value": null,
            "Key": "Purpose"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

describe-traffic-mirror-filters

以下代码示例演示了如何使用 describe-traffic-mirror-filters。

Amazon CLI

查看流量镜像筛选条件

以下 describe-traffic-mirror-filters 示例显示了所有流量镜像筛选条件的详细信息。

```
aws ec2 describe-traffic-mirror-filters
```

输出：

```
{
    "TrafficMirrorFilters": [
        {
            "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",
            "IngressFilterRules": [
                {
                    "TrafficMirrorFilterRuleId": "tmfr-0ca76e0e08EXAMPLE",
                    "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",
                    "TrafficDirection": "ingress",
                    "RuleNumber": 100,
                    "RuleAction": "accept",
                    "Protocol": 6,
                    "DestinationCidrBlock": "10.0.0.0/24",
                    "SourceCidrBlock": "10.0.0.0/24",
                    "Description": "TCP Rule"
                }
            ],
            "EgressFilterRules": [],
            "NetworkServices": [],
            "Description": "Example filter",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《流量镜像指南》中的查看流量镜像筛选条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrafficMirrorFilters。

describe-traffic-mirror-sessions

以下代码示例演示了如何使用 describe-traffic-mirror-sessions。

Amazon CLI

描述流量镜像会话

以下 describe-traffic-mirror-sessions 示例显示了流量镜像会话的详细信息。

```
aws ec2 describe-traffic-mirror-sessions
```

输出：

```
{
    "TrafficMirrorSessions": [
        {
            "Tags": [],
            "VirtualNetworkId": 42,
            "OwnerId": "111122223333",
            "Description": "TCP Session",
            "NetworkInterfaceId": "eni-0a471a5cf3EXAMPLE",
            "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE",
            "TrafficMirrorFilterId": "tmf-083e18f985EXAMPLE",
            "PacketLength": 20,
            "SessionNumber": 1,
            "TrafficMirrorSessionId": "tms-0567a4c684EXAMPLE"
        },
        {
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "tag test"
                }
            ],
            "VirtualNetworkId": 13314501,
            "OwnerId": "111122223333",
            "Description": "TCP Session",
            "NetworkInterfaceId": "eni-0a471a5cf3EXAMPLE",
            "TrafficMirrorTargetId": "tmt-03665551cbEXAMPLE",
            "TrafficMirrorFilterId": "tmf-06c787846cEXAMPLE",
            "SessionNumber": 2,
            "TrafficMirrorSessionId": "tms-0060101cf8EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的查看流量镜像会话详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrafficMirrorSessions。

describe-traffic-mirror-targets

以下代码示例演示了如何使用 describe-traffic-mirror-targets。

Amazon CLI

描述流量镜像目标

以下 describe-traffic-mirror-targets 示例显示了有关指定流量镜像目标的信息。

```
aws ec2 describe-traffic-mirror-targets \
    --traffic-mirror-target-ids tmt-0dabe9b0a6EXAMPLE
```

输出：

```
{
    "TrafficMirrorTargets": [
        {
            "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE",
            "NetworkLoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:111122223333:loadbalancer/net/NLB/7cdec873fEXAMPLE",
            "Type": "network-load-balancer",
            "Description": "Example Network Load Balancer target",
            "OwnerId": "111122223333",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 流量镜像指南》中的流量镜像目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrafficMirrorTargets。

describe-transit-gateway-attachments

以下代码示例演示了如何使用 describe-transit-gateway-attachments。

Amazon CLI

查看中转网关连接

以下 describe-transit-gateway-attachments 示例显示了中转网关连接的详细信息。

```
aws ec2 describe-transit-gateway-attachments
```

输出：

```
{
    "TransitGatewayAttachments": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-01f8100bc7EXAMPLE",
            "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
            "TransitGatewayOwnerId": "123456789012",
            "ResourceOwnerId": "123456789012",
            "ResourceType": "vpc",
            "ResourceId": "vpc-3EXAMPLE",
            "State": "available",
            "Association": {
                "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
                "State": "associated"
            },
            "CreationTime": "2019-08-26T14:59:25.000Z",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Example"
                }
            ]
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
            "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
            "TransitGatewayOwnerId": "123456789012",
            "ResourceOwnerId": "123456789012",
            "ResourceType": "vpc",
            "ResourceId": "vpc-0065acced4EXAMPLE",
            "State": "available",
            "Association": {
                "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
                "State": "associated"
```

```
            },
            "CreationTime": "2019-08-07T17:03:07.000Z",
            "Tags": []
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",
            "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
            "TransitGatewayOwnerId": "123456789012",
            "ResourceOwnerId": "123456789012",
            "ResourceType": "direct-connect-gateway",
            "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
            "State": "available",
            "Association": {
                "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
                "State": "associated"
            },
            "CreationTime": "2019-08-14T20:27:44.000Z",
            "Tags": []
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
            "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
            "TransitGatewayOwnerId": "123456789012",
            "ResourceOwnerId": "123456789012",
            "ResourceType": "direct-connect-gateway",
            "ResourceId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
            "State": "available",
            "Association": {
                "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
                "State": "associated"
            },
            "CreationTime": "2019-08-14T20:33:02.000Z",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的使用中转网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayAttachments。

describe-transit-gateway-connect-peers

以下代码示例演示了如何使用 describe-transit-gateway-connect-peers。

Amazon CLI

描述 Transit Gateway Connect 对等

以下 describe-transit-gateway-connect-peers 示例描述了指定的 Connect 对等。

```
aws ec2 describe-transit-gateway-connect-peers \
    --transit-gateway-connect-peer-ids tgw-connect-peer-0666adbac4EXAMPLE
```

输出：

```
{
    "TransitGatewayConnectPeers": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",
            "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",
            "State": "available",
            "CreationTime": "2021-10-13T03:35:17.000Z",
            "ConnectPeerConfiguration": {
                "TransitGatewayAddress": "10.0.0.234",
                "PeerAddress": "172.31.1.11",
                "InsideCidrBlocks": [
                    "169.254.6.0/29"
                ],
                "Protocol": "gre",
                "BgpConfigurations": [
                    {
                        "TransitGatewayAsn": 64512,
                        "PeerAsn": 64512,
                        "TransitGatewayAddress": "169.254.6.2",
                        "PeerAddress": "169.254.6.1",
                        "BgpStatus": "down"
                    },
                    {
                        "TransitGatewayAsn": 64512,
                        "PeerAsn": 64512,
                        "TransitGatewayAddress": "169.254.6.3",
                        "PeerAddress": "169.254.6.1",
                        "BgpStatus": "down"
```

```
                    }
                ]
            },
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayConnectPeers。

describe-transit-gateway-connects

以下代码示例演示了如何使用 describe-transit-gateway-connects。

Amazon CLI

描述 Transit Gateway Connect 连接

以下 describe-transit-gateway-connects 示例描述了指定的 Connect 连接。

```
aws ec2 describe-transit-gateway-connects \
    --transit-gateway-attachment-ids tgw-attach-037012e5dcEXAMPLE
```

输出：

```
{
    "TransitGatewayConnects": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",
            "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
            "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
            "State": "available",
            "CreationTime": "2021-03-09T19:59:17+00:00",
            "Options": {
                "Protocol": "gre"
            },
            "Tags": []
        }
```

```
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的 Transit Gateway Connect 连接和 Transit Gateway Connect 对等节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayConnects。

describe-transit-gateway-multicast-domains

以下代码示例演示了如何使用 describe-transit-gateway-multicast-domains。

Amazon CLI

描述中转网关组播域

以下 describe-transit-gateway-multicast-domains 示例显示了所有中转网关组播域的详细信息。

```
aws ec2 describe-transit-gateway-multicast-domains
```

输出：

```
{

    "TransitGatewayMulticastDomains": [
        {
            "TransitGatewayMulticastDomainId": "tgw-mcast-domain-000fb24d04EXAMPLE",
            "TransitGatewayId": "tgw-0bf0bffefaEXAMPLE",
            "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
east-1:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-000fb24d04EXAMPLE",
            "OwnerId": "123456789012",
            "Options": {
                "Igmpv2Support": "disable",
                "StaticSourcesSupport": "enable",
                "AutoAcceptSharedAssociations": "disable"
            },
            "State": "available",
            "CreationTime": "2019-12-10T18:32:50+00:00",
            "Tags": [
                {
```

```
                    "Key": "Name",
                    "Value": "mc1"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的管理组播域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayMulticastDomains。

describe-transit-gateway-peering-attachments

以下代码示例演示了如何使用 describe-transit-gateway-peering-attachments。

Amazon CLI

描述中转网关对等连接

以下 describe-transit-gateway-peering-attachments 示例显示了所有中转网关对等连接的详细信息。

```
aws ec2 describe-transit-gateway-peering-attachments
```

输出：

```
{
    "TransitGatewayPeeringAttachments": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
            "RequesterTgwInfo": {
                "TransitGatewayId": "tgw-123abc05e04123abc",
                "OwnerId": "123456789012",
                "Region": "us-west-2"
            },
            "AccepterTgwInfo": {
                "TransitGatewayId": "tgw-11223344aabbcc112",
                "OwnerId": "123456789012",
                "Region": "us-east-2"
```

```
            },
            "State": "pendingAcceptance",
            "CreationTime": "2019-12-09T11:38:05.000Z",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关对等连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayPeeringAttachments。

describe-transit-gateway-policy-tables

以下代码示例演示了如何使用 describe-transit-gateway-policy-tables。

Amazon CLI

描述中转网关策略表

以下 describe-transit-gateway-policy-tables 示例描述了指定的中转网关策略表。

```
aws ec2 describe-transit-gateway-policy-tables \
    --transit-gateway-policy-table-ids tgw-ptb-0a16f134b78668a81
```

输出：

```
{
    "TransitGatewayPolicyTables": [
        {
            "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
            "TransitGatewayId": "tgw-067f8505c18f0bd6e",
            "State": "available",
            "CreationTime": "2023-11-28T16:36:43+00:00",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关用户指南》中的中转网关策略表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayPolicyTables。

describe-transit-gateway-route-tables

以下代码示例演示了如何使用 describe-transit-gateway-route-tables。

Amazon CLI

描述中转网关路由表

以下 describe-transit-gateway-route-tables 示例显示中转网关路由表的详细信息。

```
aws ec2 describe-transit-gateway-route-tables
```

输出：

```
{
    "TransitGatewayRouteTables": [
        {
            "TransitGatewayRouteTableId": "tgw-rtb-0ca78a549EXAMPLE",
            "TransitGatewayId": "tgw-0bc994abffEXAMPLE",
            "State": "available",
            "DefaultAssociationRouteTable": true,
            "DefaultPropagationRouteTable": true,
            "CreationTime": "2018-11-28T14:24:49.000Z",
            "Tags": []
        },
        {
            "TransitGatewayRouteTableId": "tgw-rtb-0e8f48f148EXAMPLE",
            "TransitGatewayId": "tgw-0043d72bb4EXAMPLE",
            "State": "available",
            "DefaultAssociationRouteTable": true,
            "DefaultPropagationRouteTable": true,
            "CreationTime": "2018-11-28T14:24:00.000Z",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的查看中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayRouteTables。

describe-transit-gateway-vpc-attachments

以下代码示例演示了如何使用 describe-transit-gateway-vpc-attachments。

Amazon CLI

描述中转网关 VPC 连接

以下 describe-transit-gateway-vpc-attachments 示例显示了中转网关 VPC 连接的详细信息。

```
aws ec2 describe-transit-gateway-vpc-attachments
```

输出：

```
{
    "TransitGatewayVpcAttachments": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-0a08e88308EXAMPLE",
            "TransitGatewayId": "tgw-0043d72bb4EXAMPLE",
            "VpcId": "vpc-0f501f7ee8EXAMPLE",
            "VpcOwnerId": "111122223333",
            "State": "available",
            "SubnetIds": [
                "subnet-045d586432EXAMPLE",
                "subnet-0a0ad478a6EXAMPLE"
            ],
            "CreationTime": "2019-02-13T11:04:02.000Z",
            "Options": {
                "DnsSupport": "enable",
                "Ipv6Support": "disable"
            },
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "attachment name"
                }
            ]
        }
```

```
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的查看 VPC 连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGatewayVpcAttachments。

describe-transit-gateways

以下代码示例演示了如何使用 describe-transit-gateways。

Amazon CLI

描述中转网关

以下 describe-transit-gateways 示例检索有关中转网关的详细信息。

```
aws ec2 describe-transit-gateways
```

输出：

```
{
    "TransitGateways": [
        {
            "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
            "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-
gateway/tgw-0262a0e521EXAMPLE",
            "State": "available",
            "OwnerId": "111122223333",
            "Description": "MyTGW",
            "CreationTime": "2019-07-10T14:02:12.000Z",
            "Options": {
                "AmazonSideAsn": 64516,
                "AutoAcceptSharedAttachments": "enable",
                "DefaultRouteTableAssociation": "enable",
                "AssociationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
                "DefaultRouteTablePropagation": "enable",
                "PropagationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
                "VpnEcmpSupport": "enable",
                "DnsSupport": "enable"
            },
            "Tags": []
```

```
        },
        {
            "TransitGatewayId": "tgw-0fb8421e2dEXAMPLE",
            "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-
gateway/tgw-0fb8421e2da853bf3",
            "State": "available",
            "OwnerId": "111122223333",
            "CreationTime": "2019-03-15T22:57:33.000Z",
            "Options": {
                "AmazonSideAsn": 65412,
                "AutoAcceptSharedAttachments": "disable",
                "DefaultRouteTableAssociation": "enable",
                "AssociationDefaultRouteTableId": "tgw-rtb-06a241a3d8EXAMPLE",
                "DefaultRouteTablePropagation": "enable",
                "PropagationDefaultRouteTableId": "tgw-rtb-06a241a3d8EXAMPLE",
                "VpnEcmpSupport": "enable",
                "DnsSupport": "enable"
            },
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "TGW1"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransitGateways。

describe-verified-access-endpoints

以下代码示例演示了如何使用 describe-verified-access-endpoints。

Amazon CLI

描述已验证访问端点

以下 describe-verified-access-endpoints 示例描述了指定的已验证访问端点。

```
aws ec2 describe-verified-access-endpoints \
    --verified-access-endpoint-ids vae-066fac616d4d546f2
```

输出：

```
{
    "VerifiedAccessEndpoints": [
        {
            "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
            "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
            "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
            "ApplicationDomain": "example.com",
            "EndpointType": "network-interface",
            "AttachmentType": "vpc",
            "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
            "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
            "SecurityGroupIds": [
                "sg-004915970c4c8f13a"
            ],
            "NetworkInterfaceOptions": {
                "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
                "Protocol": "https",
                "Port": 443
            },
            "Status": {
                "Code": "active"
            },
            "Description": "",
            "CreationTime": "2023-08-25T20:54:43",
            "LastUpdatedTime": "2023-08-25T22:17:26",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-va-endpoint"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVerifiedAccessEndpoints。

describe-verified-access-groups

以下代码示例演示了如何使用 describe-verified-access-groups。

Amazon CLI

描述已验证访问组

以下 describe-verified-access-groups 示例描述指定的已验证访问组。

```
aws ec2 describe-verified-access-groups \
    --verified-access-group-ids vagr-0dbe967baf14b7235
```

输出：

```
{
    "VerifiedAccessGroups": [
        {
            "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
            "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
            "Description": "Testing Verified Access",
            "Owner": "123456789012",
            "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
            "CreationTime": "2023-08-25T19:55:19",
            "LastUpdatedTime": "2023-08-25T22:17:25",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-va-group"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVerifiedAccessGroups。

describe-verified-access-instance-logging-configurations

以下代码示例演示了如何使用 describe-verified-access-instance-logging-configurations。

Amazon CLI

描述已验证访问实例的日志记录配置

以下 describe-verified-access-instance-logging-configurations 示例描述了指定的已验证访问实例的日志记录配置。

```
aws ec2 describe-verified-access-instance-logging-configurations \
    --verified-access-instance-ids vai-0ce000c0b7643abea
```

输出：

```
{
    "LoggingConfigurations": [
        {
            "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
            "AccessLogs": {
                "S3": {
                    "Enabled": false
                },
                "CloudWatchLogs": {
                    "Enabled": true,
                    "DeliveryStatus": {
                        "Code": "success"
                    },
                    "LogGroup": "my-log-group"
                },
                "KinesisDataFirehose": {
                    "Enabled": false
                },
                "LogVersion": "ocsf-1.0.0-rc.2",
                "IncludeTrustContext": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVerifiedAccessInstanceLoggingConfigurations。

describe-verified-access-instances

以下代码示例演示了如何使用 describe-verified-access-instances。

Amazon CLI

描述已验证访问实例

以下 describe-verified-access-instances 示例描述了指定的已验证访问实例。

```
aws ec2 describe-verified-access-instances \
    --verified-access-instance-ids vai-0ce000c0b7643abea
```

输出：

```
{
    "VerifiedAccessInstances": [
        {
            "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
            "Description": "Testing Verified Access",
            "VerifiedAccessTrustProviders": [
                {
                    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
                    "TrustProviderType": "user",
                    "UserTrustProviderType": "iam-identity-center"
                }
            ],
            "CreationTime": "2023-08-25T18:27:56",
            "LastUpdatedTime": "2023-08-25T19:03:32",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-ava-instance"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVerifiedAccessInstances。

describe-verified-access-trust-providers

以下代码示例演示了如何使用 describe-verified-access-trust-providers。

Amazon CLI

描述已验证访问信任提供商

以下 describe-verified-access-trust-providers 示例描述了指定的已验证访问信任提供商。

```
aws ec2 describe-verified-access-trust-providers \
    --verified-access-trust-provider-ids vatp-0bb32de759a3e19e7
```

输出：

```
{
    "VerifiedAccessTrustProviders": [
        {
            "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
            "Description": "Testing Verified Access",
            "TrustProviderType": "user",
            "UserTrustProviderType": "iam-identity-center",
            "PolicyReferenceName": "idc",
            "CreationTime": "2023-08-25T19:00:38",
            "LastUpdatedTime": "2023-08-25T19:03:32",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-va-trust-provider"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问的信任提供商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVerifiedAccessTrustProviders。

describe-volume-attribute

以下代码示例演示了如何使用 describe-volume-attribute。

Amazon CLI

描述卷属性

此示例命令描述了 ID 为 vol-049df61146c4d7901 的卷的 autoEnableIo 属性。

命令:

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --
attribute autoEnableIO
```

输出：

```
{
    "AutoEnableIO": {
        "Value": false
    },
    "VolumeId": "vol-049df61146c4d7901"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVolumeAttribute。

describe-volume-status

以下代码示例演示了如何使用 describe-volume-status。

Amazon CLI

描述单个卷的状态

此示例命令描述了卷 vol-1234567890abcdef0 的状态。

命令:

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

输出：

```
{
    "VolumeStatuses": [
        {
            "VolumeStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "io-enabled"
                    },
                    {
                        "Status": "not-applicable",
                        "Name": "io-performance"
                    }
                ]
            },
            "AvailabilityZone": "us-east-1a",
            "VolumeId": "vol-1234567890abcdef0",
            "Actions": [],
            "Events": []
        }
    ]
}
```

描述受损卷的状态

此示例命令描述了所有受损卷的状态。在此示例输出中，没有受损卷。

命令:

```
aws ec2 describe-volume-status --filters Name=volume-status.status,Values=impaired
```

输出：

```
{
    "VolumeStatuses": []
}
```

如果您的卷状态检查失败（状态为“受损”），请参阅《Amazon EC2 用户指南》中的“使用受损卷”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVolumeStatus。

describe-volumes-modifications

以下代码示例演示了如何使用 describe-volumes-modifications。

Amazon CLI

描述卷的修改状态

以下 describe-volumes-modifications 示例描述了指定卷的卷修改状态。

```
aws ec2 describe-volumes-modifications \
    --volume-ids vol-1234567890abcdef0
```

输出：

```
{
    "VolumeModification": {
        "TargetSize": 150,
        "TargetVolumeType": "io1",
        "ModificationState": "optimizing",
        "VolumeId": " vol-1234567890abcdef0",
        "TargetIops": 100,
        "StartTime": "2019-05-17T11:27:19.000Z",
        "Progress": 70,
        "OriginalVolumeType": "io1",
        "OriginalIops": 100,
        "OriginalSize": 100
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVolumesModifications。

describe-volumes

以下代码示例演示了如何使用 describe-volumes。

Amazon CLI

示例 1：描述卷

以下 describe-volumes 示例描述了当前区域中的指定卷。

```
aws ec2 describe-volumes \
    --volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

输出：

```
{
    "Volumes": [
        {
            "AvailabilityZone": "us-east-1a",
            "Attachments": [
                {
                    "AttachTime": "2013-12-18T22:35:00.000Z",
                    "InstanceId": "i-1234567890abcdef0",
                    "VolumeId": "vol-049df61146c4d7901",
                    "State": "attached",
                    "DeleteOnTermination": true,
                    "Device": "/dev/sda1"
                }
            ],
            "Encrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-2a:123456789012:key/8c5b2c63-b9bc-45a3-
a87a-5513eEXAMPLE,
            "VolumeType": "gp2",
            "VolumeId": "vol-049df61146c4d7901",
            "State": "in-use",
            "Iops": 100,
            "SnapshotId": "snap-1234567890abcdef0",
            "CreateTime": "2019-12-18T22:35:00.084Z",
            "Size": 8
        },
        {
            "AvailabilityZone": "us-east-1a",
            "Attachments": [],
            "Encrypted": false,
            "VolumeType": "gp2",
            "VolumeId": "vol-1234567890abcdef0",
            "State": "available",
            "Iops": 300,
            "SnapshotId": "",
            "CreateTime": "2020-02-27T00:02:41.791Z",
            "Size": 100
```

```
        }
    ]
}
```

示例 2：描述连接到特定实例的卷

以下 describe-volumes 示例描述了既连接到指定实例又设置为在实例终止时删除的所有卷。

```
aws ec2 describe-volumes \
    --region us-east-1 \
    --filters Name=attachment.instance-
id,Values=i-1234567890abcdef0 Name=attachment.delete-on-termination,Values=true
```

有关 describe-volumes 的输出示例，请参阅示例 1。

示例 3：描述特定可用区中的可用卷

以下 describe-volumes 示例描述了状态为 available 且位于指定可用区中的所有卷。

```
aws ec2 describe-volumes \
    --filters Name=status,Values=available Name=availability-zone,Values=us-east-1a
```

有关 describe-volumes 的输出示例，请参阅示例 1。

示例 4：根据标签描述卷

以下 describe-volumes 示例描述了具有标签键 Name 和以 Test 开头的值的所有卷。然后，使用仅显示卷的标签和 ID 的查询筛选输出。

```
aws ec2 describe-volumes \
    --filters Name=tag:Name,Values=Test* \
    --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

输出：

```
[
    {
       "Tag": [
           {
               "Value": "Test2",
               "Key": "Name"
```

```
            }
        ],
        "ID": "vol-1234567890abcdef0"
    },
    {
        "Tag": [
            {
                "Value": "Test1",
                "Key": "Name"
            }
        ],
        "ID": "vol-049df61146c4d7901"
     }
]
```

有关使用标签筛选器的更多示例，请参阅《Amazon EC2 用户指南》中的使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVolumes。

describe-vpc-attribute

以下代码示例演示了如何使用 describe-vpc-attribute。

Amazon CLI

描述 enableDnsSupport 属性

此示例描述了 enableDnsSupport 属性。此属性指示是否为 VPC 启用 DNS 解析。如果该属性为 true，则 Amazon DNS 服务器会将您实例的 DNS 主机名称解析为相应的 IP 地址，否则不会解析。

命令:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

输出：

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsSupport": {
        "Value": true
    }
```

```
}
```

描述 enableDnsHostnames 属性

此示例描述了 enableDnsHostnames 属性。此属性指示在 VPC 中启动的实例是否可获得 DNS 主机名。如果该属性为 true，则 VPC 内的实例可获得 DNS 主机名称，否则将无法获得。

命令:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsHostnames
```

输出：

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsHostnames": {
        "Value": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcAttribute。

describe-vpc-classic-link-dns-support

以下代码示例演示了如何使用 describe-vpc-classic-link-dns-support。

Amazon CLI

描述 VPC 的 ClassicLink DNS 支持

此示例描述了所有 VPC 的 ClassicLink DNS 支持状态。

命令:

```
aws ec2 describe-vpc-classic-link-dns-support
```

输出：

```
{
  "Vpcs": [
    {
```

```
      "VpcId": "vpc-88888888",
      "ClassicLinkDnsSupported": true
    },
    {
      "VpcId": "vpc-1a2b3c4d",
      "ClassicLinkDnsSupported": false
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcClassicLinkDnsSupport。

describe-vpc-classic-link

以下代码示例演示了如何使用 describe-vpc-classic-link。

Amazon CLI

描述 VPC 的 ClassicLink 状态

此示例列出了 vpc-88888888 的 ClassicLink 状态。

命令:

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

输出：

```
{
  "Vpcs": [
    {
      "ClassicLinkEnabled": true,
      "VpcId": "vpc-88888888",
      "Tags": [
        {
          "Value": "classiclinkvpc",
          "Key": "Name"
        }
      ]
    }
  ]
```

```
}
```

此示例仅列出了启用 ClassicLink 的 VPC（`is-classic-link-enabled` 的筛选条件值设置为 `true`）。

命令:

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-
enabled,Values=true"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcClassicLink。

describe-vpc-endpoint-associations

以下代码示例演示了如何使用 `describe-vpc-endpoint-associations`。

Amazon CLI

描述 VPC 端点关联

以下 `describe-vpc-endpoint-associations` 示例描述了 VPC 端点关联。

```
aws ec2 describe-vpc-endpoint-associations
```

输出：

```
{
    "VpcEndpointAssociations": [
        {
            "Id": "vpce-rsc-asc-0a810ca6ac8866bf9",
            "VpcEndpointId": "vpce-019b90d6f16d4f958",
            "AssociatedResourceAccessibility": "Accessible",
            "DnsEntry": {
                "DnsName":
 "vpce-019b90d6f16d4f958.rcfg-07129f3acded87625.4232ccc.vpc-lattice-rsc.us-
east-2.on.aws",
                "HostedZoneId": "Z03265862FOUNWMZOKUF4"
            },
            "AssociatedResourceArn": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-07129f3acded87625"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Manage VPC endpoint associations。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeVpcEndpointAssociations。

`describe-vpc-endpoint-connection-notifications`

以下代码示例演示了如何使用 `describe-vpc-endpoint-connection-notifications`。

Amazon CLI

描述端点连接通知

以下 `describe-vpc-endpoint-connection-notifications` 示例描述了您的所有端点连接通知。

```
aws ec2 describe-vpc-endpoint-connection-notifications
```

输出：

```
{
   "ConnectionNotificationSet": [
       {
           "ConnectionNotificationState": "Enabled",
           "ConnectionNotificationType": "Topic",
           "ConnectionEvents": [
               "Accept",
               "Reject",
               "Delete",
               "Connect"
           ],
           "ConnectionNotificationId": "vpce-nfn-04bcb952bc8af7abc",
           "ConnectionNotificationArn": "arn:aws:sns:us-
east-1:123456789012:VpceNotification",
           "VpcEndpointId": "vpce-0324151a02f327123"
       }
   ]
 }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpointConnectionNotifications。

`describe-vpc-endpoint-connections`

以下代码示例演示了如何使用 `describe-vpc-endpoint-connections`。

Amazon CLI

描述 VPC 端点连接

此示例描述了与您的端点服务的接口端点连接，并筛选结果以显示状态为 `PendingAcceptance` 的端点。

命令:

```
aws ec2 describe-vpc-endpoint-connections --filters Name=vpc-endpoint-state,Values=pendingAcceptance
```

输出：

```
{
  "VpcEndpointConnections": [
      {
          "VpcEndpointId": "vpce-0abed31004e618123",
          "ServiceId": "vpce-svc-0abced088d20def56",
          "CreationTimestamp": "2017-11-30T10:00:24.350Z",
          "VpcEndpointState": "pendingAcceptance",
          "VpcEndpointOwner": "123456789012"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpointConnections。

`describe-vpc-endpoint-service-configurations`

以下代码示例演示了如何使用 `describe-vpc-endpoint-service-configurations`。

Amazon CLI

描述端点服务配置

以下 describe-vpc-endpoint-service-configurations 示例描述了您的端点服务配置。

```
aws ec2 describe-vpc-endpoint-service-configurations
```

输出：

```
{
    "ServiceConfigurations": [
        {
            "ServiceType": [
                {
                    "ServiceType": "GatewayLoadBalancer"
                }
            ],
            "ServiceId": "vpce-svc-012d33a1c4321cabc",
            "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-012d33a1c4321cabc",
            "ServiceState": "Available",
            "AvailabilityZones": [
                "us-east-1d"
            ],
            "AcceptanceRequired": false,
            "ManagesVpcEndpoints": false,
            "GatewayLoadBalancerArns": [
                "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
gwy/GWLBService/123210844e429123"
            ],
            "Tags": []
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "ServiceId": "vpce-svc-123cabc125efa123",
            "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123cabc125efa123",
            "ServiceState": "Available",
            "AvailabilityZones": [
                "us-east-1a"
            ],
            "AcceptanceRequired": true,
            "ManagesVpcEndpoints": false,
```

```
            "NetworkLoadBalancerArns": [
                "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
net/NLBforService/1238753950b25123"
            ],
            "BaseEndpointDnsNames": [
                "vpce-svc-123cabc125efa123.us-east-1.vpce.amazonaws.com"
            ],
            "PrivateDnsName": "example.com",
            "PrivateDnsNameConfiguration": {
                "State": "failed",
                "Type": "TXT",
                "Value": "vpce:qUAth3FdeABCApUiXabc",
                "Name": "_1d367jvbg34znqvyefrj"
            },
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Concepts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpointServiceConfigurations。

`describe-vpc-endpoint-service-permissions`

以下代码示例演示了如何使用 `describe-vpc-endpoint-service-permissions`。

Amazon CLI

描述端点服务权限

此示例描述了指定端点服务的权限。

命令:

```
aws ec2 describe-vpc-endpoint-service-permissions --service-id vpce-
svc-03d5ebb7d9579a2b3
```

输出：

```
{
   "AllowedPrincipals": [
```

```
        {
            "PrincipalType": "Account",
            "Principal": "arn:aws:iam::123456789012:root"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpointServicePermissions。

describe-vpc-endpoint-services

以下代码示例演示了如何使用 describe-vpc-endpoint-services。

Amazon CLI

示例 1：描述所有 VPC 端点服务

以下 describe-vpc-endpoint-services 示例列出 Amazon 区域的所有 VPC 端点服务。

```
aws ec2 describe-vpc-endpoint-services
```

输出：

```
{
    "ServiceDetails": [
        {
            "ServiceType": [
                {
                    "ServiceType": "Gateway"
                }
            ],
            "AcceptanceRequired": false,
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
```

```
                "us-east-1f"
            ],
            "BaseEndpointDnsNames": [
                "dynamodb.us-east-1.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ec2.us-east-1.amazonaws.com",
            "ServiceName": "com.amazonaws.us-east-1.ec2",
            "VpcEndpointPolicySupported": false,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ],
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ec2.us-east-1.vpce.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ssm.us-east-1.amazonaws.com",
            "ServiceName": "com.amazonaws.us-east-1.ssm",
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
```

```
                "us-east-1e"
            ],
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ssm.us-east-1.vpce.amazonaws.com"
            ]
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.dynamodb",
        "com.amazonaws.us-east-1.ec2",
        "com.amazonaws.us-east-1.ec2messages",
        "com.amazonaws.us-east-1.elasticloadbalancing",
        "com.amazonaws.us-east-1.kinesis-streams",
        "com.amazonaws.us-east-1.s3",
        "com.amazonaws.us-east-1.ssm"
    ]
}
```

示例 2：描述端点服务的详细信息

以下 `describe-vpc-endpoint-services` 示例列出 Amazon S3 接口端点服务的详细信息。

```
aws ec2 describe-vpc-endpoint-services \
    --filter 'Name=service-type,Values=Interface' Name=service-
name,Values=com.amazonaws.us-east-1.s3
```

输出：

```
{
    "ServiceDetails": [
        {
            "ServiceName": "com.amazonaws.us-east-1.s3",
            "ServiceId": "vpce-svc-081d84efcdEXAMPLE",
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
```

```
                "us-east-1d",
                "us-east-1e",
            "us-east-1f"
            ],
            "Owner": "amazon",
            "BaseEndpointDnsNames": [
                "s3.us-east-1.vpce.amazonaws.com"
            ],
            "VpcEndpointPolicySupported": true,
            "AcceptanceRequired": false,
            "ManagesVpcEndpoints": false,
            "Tags": []
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.s3"
    ]
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 View available Amazon service names。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpointServices。

`describe-vpc-endpoints`

以下代码示例演示了如何使用 `describe-vpc-endpoints`。

Amazon CLI

描述 VPC 端点

以下 `describe-vpc-endpoints` 示例显示了所有 VPC 端点的详细信息。

```
aws ec2 describe-vpc-endpoints
```

输出：

```
{
    "VpcEndpoints": [
        {
            "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",
```

```
            "VpcId": "vpc-aabb1122",
            "NetworkInterfaceIds": [],
            "SubnetIds": [],
            "PrivateDnsEnabled": true,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
            "RouteTableIds": [
                "rtb-3d560345"
            ],
            "Groups": [],
            "VpcEndpointId": "vpce-032a826a",
            "VpcEndpointType": "Gateway",
            "CreationTimestamp": "2017-09-05T20:41:28Z",
            "DnsEntries": [],
            "OwnerId": "123456789012"
        },
        {
            "PolicyDocument": "{\n  \"Statement\": [\n    {\n      \"Action\": \"*
\", \n      \"Effect\": \"Allow\", \n      \"Principal\": \"*\", \n      \"Resource
\": \"*\"\n    }\n  ]\n}",
            "VpcId": "vpc-1a2b3c4d",
            "NetworkInterfaceIds": [
                "eni-2ec2b084",
                "eni-1b4a65cf"
            ],
            "SubnetIds": [
                "subnet-d6fcaa8d",
                "subnet-7b16de0c"
            ],
            "PrivateDnsEnabled": false,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.elasticloadbalancing",
            "RouteTableIds": [],
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-54e8bf31"
                }
            ],
            "VpcEndpointId": "vpce-0f89a33420c1931d7",
            "VpcEndpointType": "Interface",
            "CreationTimestamp": "2017-09-05T17:55:27.583Z",
            "DnsEntries": [
                {
```

```
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-
bluzidnv.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1b.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1a.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                }
            ],
            "OwnerId": "123456789012"
        },
        {
            "VpcEndpointId": "vpce-aabbaabbaabbaabba",
            "VpcEndpointType": "GatewayLoadBalancer",
            "VpcId": "vpc-111122223333aabbc",
            "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-123123a1c43abc123",
            "State": "available",
            "SubnetIds": [
                "subnet-0011aabbcc2233445"
            ],
            "RequesterManaged": false,
            "NetworkInterfaceIds": [
                "eni-01010120203030405"
            ],
            "CreationTimestamp": "2020-11-11T08:06:03.522Z",
            "Tags": [],
            "OwnerId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon PrivateLink User Guide》中的 Concepts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcEndpoints。

describe-vpc-peering-connections

以下代码示例演示了如何使用 describe-vpc-peering-connections。

Amazon CLI

描述 VPC 对等连接

此示例描述了您的所有 VPC 对等连接。

命令:

```
aws ec2 describe-vpc-peering-connections
```

输出：

```
{
    "VpcPeeringConnections": [
        {
            "Status": {
                "Message": "Active",
                "Code": "active"
            },
            "Tags": [
                {
                    "Value": "Peering-1",
                    "Key": "Name"
                }
            ],
            "AccepterVpcInfo": {
                "OwnerId": "111122223333",
                "VpcId": "vpc-1a2b3c4d",
                "CidrBlock": "10.0.1.0/28"
            },
            "VpcPeeringConnectionId": "pcx-11122233",
            "RequesterVpcInfo": {
                "PeeringOptions": {
                    "AllowEgressFromLocalVpcToRemoteClassicLink": false,
                    "AllowEgressFromLocalClassicLinkToRemoteVpc": false
                },
                "OwnerId": "444455556666",
                "VpcId": "vpc-123abc45",
                "CidrBlock": "192.168.0.0/16"
```

```
            }
        },
        {
            "Status": {
                "Message": "Pending Acceptance by 444455556666",
                "Code": "pending-acceptance"
            },
            "Tags": [],
            "RequesterVpcInfo": {
                "PeeringOptions": {
                    "AllowEgressFromLocalVpcToRemoteClassicLink": false,
                    "AllowEgressFromLocalClassicLinkToRemoteVpc": false
                },
                "OwnerId": "444455556666",
                "VpcId": "vpc-11aa22bb",
                "CidrBlock": "10.0.0.0/28"
            },
            "VpcPeeringConnectionId": "pcx-abababab",
            "ExpirationTime": "2014-04-03T09:12:43.000Z",
            "AccepterVpcInfo": {
                "OwnerId": "444455556666",
                "VpcId": "vpc-33cc44dd"
            }
        }
    ]
}
```

描述特定的 VPC 对等连接

此示例描述了您的所有处于待接受状态的 VPC 对等连接。

命令:

```
aws ec2 describe-vpc-peering-connections --filters Name=status-code,Values=pending-acceptance
```

此示例描述了具有标签 Owner=Finance 的所有 VPC 对等连接。

命令:

```
aws ec2 describe-vpc-peering-connections --filters Name=tag:Owner,Values=Finance
```

此示例描述了您为指定 VPC vpc-1a2b3c4d 请求的所有 VPC 对等连接。

命令:

```
aws ec2 describe-vpc-peering-connections --filters Name=requester-vpc-info.vpc-id,Values=vpc-1a2b3c4d
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcPeeringConnections。

describe-vpcs

以下代码示例演示了如何使用 describe-vpcs。

Amazon CLI

示例 1：描述所有 VPC

以下 describe-vpcs 示例检索 VPC 的详细信息。

```
aws ec2 describe-vpcs
```

输出：

```
{
    "Vpcs": [
        {
            "CidrBlock": "30.1.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-0e9801d129EXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-062c64cfafEXAMPLE",
                    "CidrBlock": "30.1.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
```

```
                    "Key": "Name",
                    "Value": "Not Shared"
                }
            ]
        },
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "222222222222",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

示例 2：描述指定的 VPC

以下 `describe-vpcs` 示例检索指定 VPC 的详细信息。

```
aws ec2 describe-vpcs \
    --vpc-ids vpc-06e4ab6c6cEXAMPLE
```

输出：

```
{
    "Vpcs": [
```

```
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpcs。

describe-vpn-connections

以下代码示例演示了如何使用 describe-vpn-connections。

Amazon CLI

示例 1：描述 VPN 连接

以下 describe-vpn-connections 示例描述了您的所有站点到站点 VPN 连接。

```
aws ec2 describe-vpn-connections
```

输出：

```
{
    "VpnConnections": [
        {
            "CustomerGatewayConfiguration": "...configuration information...",
            "CustomerGatewayId": "cgw-01234567abcde1234",
            "Category": "VPN",
            "State": "available",
            "Type": "ipsec.1",
            "VpnConnectionId": "vpn-1122334455aabbccd",
            "TransitGatewayId": "tgw-00112233445566aab",
            "Options": {
                "EnableAcceleration": false,
                "StaticRoutesOnly": true,
                "LocalIpv4NetworkCidr": "0.0.0.0/0",
                "RemoteIpv4NetworkCidr": "0.0.0.0/0",
                "TunnelInsideIpVersion": "ipv4"
            },
            "Routes": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "CanadaVPN"
                }
            ],
            "VgwTelemetry": [
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-07-29T10:35:11.000Z",
                    "OutsideIpAddress": "203.0.113.3",
                    "Status": "DOWN",
                    "StatusMessage": ""
                },
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-09-02T09:09:33.000Z",
                    "OutsideIpAddress": "203.0.113.5",
                    "Status": "UP",
                    "StatusMessage": ""
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

示例 2：描述可用的 VPN 连接

以下 `describe-vpn-connections` 示例描述了状态为 `available` 的站点到站点 VPN 连接。

```
aws ec2 describe-vpn-connections \
    --filters "Name=state,Values=available"
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的 Amazon Site-to-Site VPN 的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpnConnections。

`describe-vpn-gateways`

以下代码示例演示了如何使用 `describe-vpn-gateways`。

Amazon CLI

描述虚拟私有网关

此示例描述了您的虚拟私有网关。

命令:

```
aws ec2 describe-vpn-gateways
```

输出：

```
{
    "VpnGateways": [
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-f211f09b",
            "VpcAttachments": [
                {
                    "State": "attached",
                    "VpcId": "vpc-98eb5ef5"
```

```
                }
            ]
        },
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-9a4cacf3",
            "VpcAttachments": [
                {
                    "State": "attaching",
                    "VpcId": "vpc-a01106c2"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVpnGateways。

`detach-classic-link-vpc`

以下代码示例演示了如何使用 `detach-classic-link-vpc`。

Amazon CLI

将 EC2-Classic 实例与 VPC 取消链接（分离

此示例将实例 i-0598c7d356eba48d7 与 VPC vpc-88888888 取消链接。

命令:

```
aws ec2 detach-classic-link-vpc --instance-id i-0598c7d356eba48d7 --vpc-
id vpc-88888888
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachClassicLinkVpc。

detach-internet-gateway

以下代码示例演示了如何使用 detach-internet-gateway。

Amazon CLI

将互联网网关与 VPC 分离

以下 detach-internet-gateway 示例将指定的互联网网关与特定 VPC 分离。

```
aws ec2 detach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC 用户指南》中的互联网网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachInternetGateway。

detach-network-interface

以下代码示例演示了如何使用 detach-network-interface。

Amazon CLI

将网络接口与实例分离

此示例将指定的网络接口与指定的实例分离。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachNetworkInterface。

detach-verified-access-trust-provider

以下代码示例演示了如何使用 detach-verified-access-trust-provider。

Amazon CLI

将信任提供商与实例分离

以下 detach-verified-access-trust-provider 示例将指定的已验证访问信任提供商与指定的已验证访问实例分离。

```
aws ec2 detach-verified-access-trust-provider \
    --verified-access-instance-id vai-0ce000c0b7643abea \
    --verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

输出：

```
{
    "VerifiedAccessTrustProvider": {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "Description": "Testing Verified Access",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center",
        "PolicyReferenceName": "idc",
        "CreationTime": "2023-08-25T19:00:38",
        "LastUpdatedTime": "2023-08-25T19:00:38"
    },
    "VerifiedAccessInstance": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "Testing Verified Access",
        "VerifiedAccessTrustProviders": [],
        "CreationTime": "2023-08-25T18:27:56",
        "LastUpdatedTime": "2023-08-25T18:27:56"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachVerifiedAccessTrustProvider。

detach-volume

以下代码示例演示了如何使用 detach-volume。

Amazon CLI

将卷与实例分离

此示例命令将卷 (vol-049df61146c4d7901) 与其连接到的实例分离。

命令:

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

输出：

```
{
    "AttachTime": "2014-02-27T19:23:06.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "VolumeId": "vol-049df61146c4d7901",
    "State": "detaching",
    "Device": "/dev/sdb"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachVolume。

detach-vpn-gateway

以下代码示例演示了如何使用 detach-vpn-gateway。

Amazon CLI

将虚拟私有网关与 VPC 分离

此示例将指定的虚拟私有网关与指定 VPC 分离。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 detach-vpn-gateway --vpn-gateway-id vgw-9a4cacf3 --vpc-id vpc-a01106c2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachVpnGateway。

disable-address-transfer

以下代码示例演示了如何使用 disable-address-transfer。

Amazon CLI

禁用弹性 IP 地址转移

以下 disable-address-transfer 示例禁用指定弹性 IP 地址的弹性 IP 地址转移。

```
aws ec2 disable-address-transfer \
    --allocation-id eipalloc-09ad461b0d03f6aaf
```

输出：

```
{
    "AddressTransfer": {
        "PublicIp": "100.21.184.216",
        "AllocationId": "eipalloc-09ad461b0d03f6aaf",
        "AddressTransferStatus": "disabled"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的转移弹性 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableAddressTransfer。

disable-aws-network-performance-metric-subscription

以下代码示例演示了如何使用 disable-aws-network-performance-metric-subscription。

Amazon CLI

禁用指标订阅

以下 disable-aws-network-performance-metric-subscription 示例禁用对指定源区域和目标区域之间聚合网络延迟的监控。

```
aws ec2 disable-aws-network-performance-metric-subscription \
    --source us-east-1 \
    --destination eu-west-1 \
    --metric aggregate-latency \
    --statistic p50
```

输出：

```
{
    "Output": true
}
```

有关更多信息，请参阅《Infrastructure Performance User Guide》中的 Manage CloudWatch subscriptions using the CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableAwsNetworkPerformanceMetricSubscription。

disable-ebs-encryption-by-default

以下代码示例演示了如何使用 disable-ebs-encryption-by-default。

Amazon CLI

默认禁用 EBS 加密

以下 disable-ebs-encryption-by-default 示例默认为当前区域中的 Amazon 账户禁用 EBS 加密。

```
aws ec2 disable-ebs-encryption-by-default
```

输出：

```
{
    "EbsEncryptionByDefault": false
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableEbsEncryptionByDefault。

disable-fast-launch

以下代码示例演示了如何使用 disable-fast-launch。

Amazon CLI

停止映像的快速启动

以下 disable-fast-launch 示例停止指定 AMI 上的快速启动，并清理现有的预置快照。

```
aws ec2 disable-fast-launch \
    --image-id ami-01234567890abcedf
```

输出：

```
{
    "ImageId": "ami-01234567890abcedf",
    "ResourceType": "snapshot",
    "SnapshotConfiguration": {},
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-01234567890abcedf",
        "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-
a8c6215d-94e6-441b-9272-dbd1f87b07e2",
        "Version": "1"
    },
    "MaxParallelLaunches": 6,
    "OwnerId": "0123456789123",
    "State": "disabling",
    "StateTransitionReason": "Client.UserInitiated",
    "StateTransitionTime": "2022-01-27T22:47:29.265000+00:00"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的为 Windows AMI 配置 EC2 快速启动设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableFastLaunch。

disable-fast-snapshot-restores

以下代码示例演示了如何使用 disable-fast-snapshot-restores。

Amazon CLI

禁用快速快照还原

以下 disable-fast-snapshot-restores 示例对指定可用区中的指定快照禁用快速快照还原。

```
aws ec2 disable-fast-snapshot-restores \
    --availability-zones us-east-2a \
    --source-snapshot-ids snap-1234567890abcdef0
```

输出：

```
{
    "Successful": [
```

```
        {
            "SnapshotId": "snap-1234567890abcdef0"
            "AvailabilityZone": "us-east-2a",
            "State": "disabling",
            "StateTransitionReason": "Client.UserInitiated",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.602Z"
        }
    ],
    "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableFastSnapshotRestores。

disable-image-block-public-access

以下代码示例演示了如何使用 disable-image-block-public-access。

Amazon CLI

在指定区域禁用 AMI 的屏蔽公共访问权限

以下 disable-image-block-public-access 示例在指定区域中的账户级别禁用 AMI 的屏蔽公共访问权限。

```
aws ec2 disable-image-block-public-access \
    --region us-east-1
```

输出：

```
{
    "ImageBlockPublicAccessState": "unblocked"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的屏蔽对 AMI 的公共访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableImageBlockPublicAccess。

disable-image-deprecation

以下代码示例演示了如何使用 disable-image-deprecation。

Amazon CLI

取消弃用 AMI

以下 disable-image-deprecation 示例取消弃用 AMI，这样将从 describe-images 输出中删除 DeprecationTime 字段。您必须是 AMI 拥有者才能执行此过程。

```
aws ec2 disable-image-deprecation \
    --image-id ami-1234567890abcdef0
```

输出：

```
{
    "RequestID": "11aabb229-4eac-35bd-99ed-be587EXAMPLE",
    "Return": "true"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弃用 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableImageDeprecation。

disable-image-deregistration-protection

以下代码示例演示了如何使用 disable-image-deregistration-protection。

Amazon CLI

禁用取消注册保护

以下 disable-image-deregistration-protection 示例为指定的映像禁用取消注册保护。

```
aws ec2 disable-image-deregistration-protection \
    --image-id ami-0b1a928a144a74ec9
```

输出：

```
{
    "Return": "disabled"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的保护 AMI 免遭取消注册。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableImageDeregistrationProtection。

`disable-image`

以下代码示例演示了如何使用 `disable-image`。

Amazon CLI

禁用 AMI

以下 `disable-image` 示例禁用指定的 AMI。

```
aws ec2 disable-image \
    --image-id ami-1234567890abcdef0
```

输出：

```
{
    "Return": "true"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的禁用 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableImage。

`disable-ipam-organization-admin-account`

以下代码示例演示了如何使用 `disable-ipam-organization-admin-account`。

Amazon CLI

禁用委派 IPAM 管理员

在特定情况下，您需要将 IPAM 与 Amazon Organizations 集成。当您执行此操作时，Amazon Organizations 管理账户会将 Amazon Organizations 成员账户委派为 IPAM 管理员。

在此示例中，您是委派 IPAM 管理员账户的 Amazon Organizations 管理账户，并且您需要禁止该账户成为 IPAM 管理员。

在发送此请求时，您可以将任何 Amazon 区域用于 `--region`。您不必使用最初委派管理员的区域、创建 IPAM 的区域或 IPAM 运营区域。如果禁用委派管理员账户，则可以随时重新启用它，或将新账户委派为 IPAM 管理员。

以下 `disable-ipam-organization-admin-account` 示例在您的 Amazon 账户中禁用委派 IPAM 管理员。

```
aws ec2 disable-ipam-organization-admin-account \
    --delegated-admin-account-id 320805250157 \
    --region ap-south-1
```

输出：

```
{
    "Success": true
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与 Amazon Organizations 中的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableIpamOrganizationAdminAccount。

`disable-serial-console-access`

以下代码示例演示了如何使用 `disable-serial-console-access`。

Amazon CLI

禁用对您的账户的 EC2 Serial Console 的访问

以下 `disable-serial-console-access` 示例禁用账户对该串行控制台的访问权限。

```
aws ec2 disable-serial-console-access
```

输出：

```
{
    "SerialConsoleAccessEnabled": false
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 EC2 Serial Console。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableSerialConsoleAccess。

disable-snapshot-block-public-access

以下代码示例演示了如何使用 disable-snapshot-block-public-access。

Amazon CLI

禁用快照的屏蔽公共访问权限

以下 disable-snapshot-block-public-access 示例禁用快照的屏蔽公共访问权限，以允许公开共享您的快照。

```
aws ec2 disable-snapshot-block-public-access
```

输出：

```
{
    "State": "unblocked"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的屏蔽对快照的公共访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableSnapshotBlockPublicAccess。

disable-transit-gateway-route-table-propagation

以下代码示例演示了如何使用 disable-transit-gateway-route-table-propagation。

Amazon CLI

禁用中转网关连接以将路由传播到指定的传播路由表

以下 disable-transit-gateway-route-table-propagation 示例禁用指定连接，以将路由传播到指定的传播路由表。

```
aws ec2 disable-transit-gateway-route-table-propagation \
    --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE \
    --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE
```

输出：

```
{
    "Propagation": {
        "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
        "ResourceId": "vpc-4d7de228",
        "ResourceType": "vpc",
        "TransitGatewayRouteTableId": "tgw-rtb-0a823edbdeEXAMPLE",
        "State": "disabled"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableTransitGatewayRouteTablePropagation。

disable-vgw-route-propagation

以下代码示例演示了如何使用 disable-vgw-route-propagation。

Amazon CLI

禁用路由传播

此示例禁止指定的虚拟私有网关将静态路由传播到指定的路由表。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-
id vgw-9a4cacf3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableVgwRoutePropagation。

disable-vpc-classic-link-dns-support

以下代码示例演示了如何使用 disable-vpc-classic-link-dns-support。

Amazon CLI

为 VPC 禁用 ClassicLink DNS 支持

此示例为 vpc-88888888 禁用 ClassicLink DNS 支持。

命令:

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableVpcClassicLinkDnsSupport。

disable-vpc-classic-link

以下代码示例演示了如何使用 disable-vpc-classic-link。

Amazon CLI

为 VPC 禁用 ClassicLink

此示例为 vpc-8888888 禁用 ClassicLink。

命令:

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableVpcClassicLink。

disassociate-address

以下代码示例演示了如何使用 disassociate-address。

Amazon CLI

解除关联 EC2-Classic 中的弹性 IP 地址

以下示例将弹性 IP 地址与 EC2 Classic 中的实例解除关联。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

解除关联 EC2-VPC 中的弹性 IP 地址

以下示例将弹性 IP 地址与 VPC 中的实例解除关联。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateAddress。

disassociate-client-vpn-target-network

以下代码示例演示了如何使用 disassociate-client-vpn-target-network。

Amazon CLI

取消网络与 Client VPN 端点之间的关联

以下 disassociate-client-vpn-target-network 示例取消与目标网络的关联，该网络与指定 Client VPN 端点的 cvpn-assoc-12312312312312312 关联 ID 相关联。

```
aws ec2 disassociate-client-vpn-target-network \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --association-id cvpn-assoc-12312312312312312
```

输出：

```
{
    "AssociationId": "cvpn-assoc-12312312312312312",
    "Status": {
        "Code": "disassociating"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的目标网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateClientVpnTargetNetwork。

disassociate-iam-instance-profile

以下代码示例演示了如何使用 disassociate-iam-instance-profile。

Amazon CLI

取消关联 IAM 实例配置文件

此示例取消 IAM 实例配置文件与关联 ID iip-assoc-05020b59952902f5f 的关联。

命令:

```
aws ec2 disassociate-iam-instance-profile --association-id iip-
assoc-05020b59952902f5f
```

输出：

```
{
  "IamInstanceProfileAssociation": {
      "InstanceId": "i-123456789abcde123",
      "State": "disassociating",
      "AssociationId": "iip-assoc-05020b59952902f5f",
      "IamInstanceProfile": {
          "Id": "AIPAI5IVIHMFFYY2DKV5Y",
          "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateIamInstanceProfile。

disassociate-instance-event-window

以下代码示例演示了如何使用 disassociate-instance-event-window。

Amazon CLI

示例 1：取消一个或多个实例与事件窗口的关联

以下 disassociate-instance-event-window 示例取消一个或多个实例与事件窗口的关联。指定 instance-event-window-id 参数以指定事件窗口。要取消关联实例，请指定 association-target 参数，并为参数值指定一个或多个实例 ID。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 2：取消实例标签与事件窗口的关联

以下 disassociate-instance-event-window 示例取消实例标签与事件窗口的关联。指定 instance-event-window-id 参数以指定事件窗口。要取消关联实例标签，请指定 association-target 参数，并为参数值指定一个或多个标签。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 3：取消专属主机与事件窗口的关联

以下 disassociate-instance-event-window 示例取消专属主机与事件窗口的关联。指定 instance-event-window-id 参数以指定事件窗口。要取消关联专属主机，请指定 association-target 参数，并为参数值指定一个或多个专属主机 ID。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target DedicatedHostIds=h-029fa35a02b99801d
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
```

```
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateInstanceEventWindow。

disassociate-ipam-resource-discovery

以下代码示例演示了如何使用 disassociate-ipam-resource-discovery。

Amazon CLI

取消资源发现与 IPAM 的关联

在此示例中，您是 IPAM 委派管理员账户，并且希望取消 IPAM 资源发现与 IPAM 的关联。您运行了描述命令，并注意到 "ResourceDiscoveryStatus": "not-found"，您希望取消其与 IPAM 的关联，以便为其他关联腾出空间。

以下 disassociate-ipam-resource-discovery 示例取消 Amazon 账户中 IPAM 资源发现的关联。

```
aws ec2 disassociate-ipam-resource-discovery \
    --ipam-resource-discovery-association-id ipam-res-disco-assoc-04382a6346357cf82
 \
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscoveryAssociation": {
        "OwnerId": "320805250157",
        "IpamResourceDiscoveryAssociationId": "ipam-res-disco-
assoc-04382a6346357cf82",
        "IpamResourceDiscoveryAssociationArn":
 "arn:aws:ec2::320805250157:ipam-resource-discovery-association/ipam-res-disco-
assoc-04382a6346357cf82",
        "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
        "IpamId": "ipam-005f921c17ebd5107",
        "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
        "IpamRegion": "us-east-1",
```

```
        "IsDefault": false,
        "ResourceDiscoveryStatus": "not-found",
        "State": "disassociate-in-progress"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateIpamResourceDiscovery。

disassociate-nat-gateway-address

以下代码示例演示了如何使用 disassociate-nat-gateway-address。

Amazon CLI

取消弹性 IP 地址与公共 NAT 网关的关联

以下 disassociate-nat-gateway-address 示例取消指定弹性 IP 地址与指定公共 NAT 网关的关联。

```
aws ec2 disassociate-nat-gateway-address \
    --nat-gateway-id nat-1234567890abcdef0 \
    --association-ids eipassoc-0f96bdca17EXAMPLE
```

输出：

```
{
    "NatGatewayId": "nat-1234567890abcdef0",
    "NatGatewayAddresses": [
        {
            "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
            "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
            "PrivateIp": "10.0.0.74",
            "PublicIp": "3.211.231.218",
            "AssociationId": "eipassoc-0f96bdca17EXAMPLE",
            "IsPrimary": false,
            "Status": "disassociating"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateNatGatewayAddress。

disassociate-route-table

以下代码示例演示了如何使用 disassociate-route-table。

Amazon CLI

取消关联路由表

此示例取消指定路由表与指定子网的关联。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateRouteTable。

disassociate-security-group-vpc

以下代码示例演示了如何使用 disassociate-security-group-vpc。

Amazon CLI

取消安全组与 VPC 的关联

以下 disassociate-security-group-vpc 示例将指定的安全组与指定的 VPC 取消关联。

```
aws ec2 disassociate-security-group-vpc \
    --group-id sg-04dbb43907d3f8a78 \
    --vpc-id vpc-0bf4c2739bc05a694
```

输出：

```
{
    "State": "disassociating"
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的将安全组与多个 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateSecurityGroupVpc。

disassociate-subnet-cidr-block

以下代码示例演示了如何使用 disassociate-subnet-cidr-block。

Amazon CLI

取消 IPv6 CIDR 块与子网的关联

此示例使用 CIDR 块的关联 ID 取消 IPv6 CIDR 块与子网的关联。

命令:

```
aws ec2 disassociate-subnet-cidr-block --association-id subnet-cidr-assoc-3aa54053
```

输出：

```
{
  "SubnetId": "subnet-5f46ec3b",
  "Ipv6CidrBlockAssociation": {
      "Ipv6CidrBlock": "2001:db8:1234:1a00::/64",
      "AssociationId": "subnet-cidr-assoc-3aa54053",
      "Ipv6CidrBlockState": {
          "State": "disassociating"
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateSubnetCidrBlock。

disassociate-transit-gateway-multicast-domain

以下代码示例演示了如何使用 disassociate-transit-gateway-multicast-domain。

Amazon CLI

取消子网与组播域的关联

以下 disassociate-transit-gateway-multicast-domain 示例取消子网与指定组播域的关联。

```
aws ec2 disassociate-transit-gateway-multicast-domain \
    --transit-gateway-attachment-id tgw-attach-070e571cd1EXAMPLE \
    --subnet-id subnet-000de86e3bEXAMPLE \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

输出：

```
{
    "Associations": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef7EXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
        "ResourceId": "vpc-7EXAMPLE",
        "ResourceType": "vpc",
        "Subnets": [
            {
                "SubnetId": "subnet-000de86e3bEXAMPLE",
                "State": "disassociating"
            }
        ]
    }
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Multicast domains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateTransitGatewayMulticastDomain。

disassociate-transit-gateway-route-table

以下代码示例演示了如何使用 disassociate-transit-gateway-route-table。

Amazon CLI

取消中转网关路由表与资源连接的关联

以下 disassociate-transit-gateway-route-table 示例取消指定连接与中转网关路由表的关联。

```
aws ec2 disassociate-transit-gateway-route-table \
    --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE \
    --transit-gateway-attachment-id tgw-attach-08e0bc912cEXAMPLE
```

输出：

```
{
    "Association": {
        "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",
        "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
        "ResourceType": "direct-connect-gateway",
        "State": "disassociating"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateTransitGatewayRouteTable。

disassociate-vpc-cidr-block

以下代码示例演示了如何使用 disassociate-vpc-cidr-block。

Amazon CLI

取消 IPv6 CIDR 块与 VPC 的关联

此示例使用 CIDR 块的关联 ID 取消 IPv6 CIDR 块与 VPC 的关联。

命令:

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-eca54085
```

输出：

```
{
  "Ipv6CidrBlockAssociation": {
      "Ipv6CidrBlock": "2001:db8:1234:1a00::/56",
      "AssociationId": "vpc-cidr-assoc-eca54085",
      "Ipv6CidrBlockState": {
          "State": "disassociating"
      }
  },
  "VpcId": "vpc-a034d6c4"
```

```
}
```

取消 IPv4 CIDR 块与 VPC 的关联

此示例取消 IPv4 CIDR 块与 VPC 的关联。

命令:

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-0287ac6b
```

输出：

```
{
  "CidrBlockAssociation": {
      "AssociationId": "vpc-cidr-assoc-0287ac6b",
      "CidrBlock": "172.18.0.0/16",
      "CidrBlockState": {
          "State": "disassociating"
      }
  },
  "VpcId": "vpc-27621243"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateVpcCidrBlock。

enable-address-transfer

以下代码示例演示了如何使用 enable-address-transfer。

Amazon CLI

启用弹性 IP 地址转移

以下 enable-address-transfer 示例启用将指定弹性 IP 地址的弹性 IP 地址转移到指定账户。

```
aws ec2 enable-address-transfer \
    --allocation-id eipalloc-09ad461b0d03f6aaf \
    --transfer-account-id 123456789012
```

输出：

```
{
     "AddressTransfer": {
        "PublicIp": "100.21.184.216",
        "AllocationId": "eipalloc-09ad461b0d03f6aaf",
        "TransferAccountId": "123456789012",
        "TransferOfferExpirationTimestamp": "2023-02-22T20:51:01.000Z",
        "AddressTransferStatus": "pending"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的转移弹性 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableAddressTransfer。

enable-aws-network-performance-metric-subscription

以下代码示例演示了如何使用 enable-aws-network-performance-metric-subscription。

Amazon CLI

启用指标订阅

以下 enable-aws-network-performance-metric-subscription 示例启用对指定源区域和目标区域之间的聚合网络延迟的监控。

```
aws ec2 enable-aws-network-performance-metric-subscription \
    --source us-east-1 \
    --destination eu-west-1 \
    --metric aggregate-latency \
    --statistic p50
```

输出：

```
{
    "Output": true
}
```

有关更多信息，请参阅《基础结构性能用户指南》中的管理订阅。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableAwsNetworkPerformanceMetricSubscription。

enable-ebs-encryption-by-default

以下代码示例演示了如何使用 enable-ebs-encryption-by-default。

Amazon CLI

默认启用 EBS 加密

以下 enable-ebs-encryption-by-default 示例默认为当前区域中的 Amazon 账户启用 EBS 加密。

```
aws ec2 enable-ebs-encryption-by-default
```

输出：

```
{
    "EbsEncryptionByDefault": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableEbsEncryptionByDefault。

enable-fast-launch

以下代码示例演示了如何使用 enable-fast-launch。

Amazon CLI

开始映像的快速启动

以下 enable-fast-launch 示例为指定的 AMI 配置快速启动，并将要启动的最大并行实例数设置为 6。用于预置 AMI 的资源类型设置为 snapshot，它也是默认值。

```
aws ec2 enable-fast-launch \
    --image-id ami-01234567890abcedf \
    --max-parallel-launches 6 \
    --resource-type snapshot
```

输出：

```
{
    "ImageId": "ami-01234567890abcedf",
```

```
    "ResourceType": "snapshot",
    "SnapshotConfiguration": {
        "TargetResourceCount": 10
    },
    "LaunchTemplate": {},
    "MaxParallelLaunches": 6,
    "OwnerId": "0123456789123",
    "State": "enabling",
    "StateTransitionReason": "Client.UserInitiated",
    "StateTransitionTime": "2022-01-27T22:16:03.199000+00:00"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的为 Windows AMI 配置 EC2 快速启动设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableFastLaunch。

enable-fast-snapshot-restores

以下代码示例演示了如何使用 enable-fast-snapshot-restores。

Amazon CLI

启用快速快照还原

以下 enable-fast-snapshot-restores 示例对指定可用区中的指定快照启用快速快照还原。

```
aws ec2 enable-fast-snapshot-restores \
    --availability-zones us-east-2a us-east-2b \
    --source-snapshot-ids snap-1234567890abcdef0
```

输出：

```
{
    "Successful": [
        {
            "SnapshotId": "snap-1234567890abcdef0"
            "AvailabilityZone": "us-east-2a",
            "State": "enabling",
            "StateTransitionReason": "Client.UserInitiated",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.602Z"
        },
        {
```

```
            "SnapshotId": "snap-1234567890abcdef0"
            "AvailabilityZone": "us-east-2b",
            "State": "enabling",
            "StateTransitionReason": "Client.UserInitiated",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.596Z"
        }
    ],
    "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableFastSnapshotRestores。

enable-image-block-public-access

以下代码示例演示了如何使用 enable-image-block-public-access。

Amazon CLI

在指定区域中启用 AMI 的屏蔽公共访问权限

以下 enable-image-block-public-access 示例在指定区域中的账户级别启用 AMI 的屏蔽公共访问权限。

```
aws ec2 enable-image-block-public-access \
    --region us-east-1 \
    --image-block-public-access-state block-new-sharing
```

输出：

```
{
    "ImageBlockPublicAccessState": "block-new-sharing"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的屏蔽对 AMI 的公共访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableImageBlockPublicAccess。

enable-image-deprecation

以下代码示例演示了如何使用 enable-image-deprecation。

Amazon CLI

弃用 AMI

以下 enable-image-deprecation 示例在特定日期和时间弃用 AMI。如果指定以秒为单位的值，Amazon EC2 会将秒四舍五入到最近的分钟数。您必须是 AMI 拥有者才能执行此过程。

```
aws ec2 enable-image-deprecation \
    --image-id ami-1234567890abcdef0 \
    --deprecate-at '2022-10-15T13:17:12.000Z'
```

输出：

```
{
    "RequestID": "59dbff89-35bd-4eac-99ed-be587EXAMPLE",
    "Return": "true"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的弃用 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableImageDeprecation。

enable-image-deregistration-protection

以下代码示例演示了如何使用 enable-image-deregistration-protection。

Amazon CLI

启用取消注册保护

以下 enable-image-deregistration-protection 示例为指定的映像启用取消注册保护。

```
aws ec2 enable-image-deregistration-protection \
    --image-id ami-0b1a928a144a74ec9
```

输出：

```
{
    "Return": "enabled-without-cooldown"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的保护 EC2 AMI 免遭取消注册。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 EnableImageDeregistrationProtection。

enable-image

以下代码示例演示了如何使用 enable-image。

Amazon CLI

启用 AMI

以下 enable-image 示例启用指定的 AMI。

```
aws ec2 enable-image \
    --image-id ami-1234567890abcdef0
```

输出：

```
{
    "Return": "true"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的禁用 AMI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableImage。

enable-ipam-organization-admin-account

以下代码示例演示了如何使用 enable-ipam-organization-admin-account。

Amazon CLI

与 Amazon Organizations 集成并将成员账户委派为 IPAM 账户

以下 enable-ipam-organization-admin-account 示例将 IPAM 与 Amazon Organizations 集成，并将成员账户委派为 IPAM 账户。

```
aws ec2 enable-ipam-organization-admin-account \
    --delegated-admin-account-id 320805250157
```

输出：

```
{
    "Success": true
}
```

有关更多信息，请参阅 Amazon VPC IPAM 用户指南中的将 IPAM 与 Amazon Organizations 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableIpamOrganizationAdminAccount。

enable-reachability-analyzer-organization-sharing

以下代码示例演示了如何使用 enable-reachability-analyzer-organization-sharing。

Amazon CLI

为 Reachability Analyzer 启用可信访问

以下 enable-reachability-analyzer-organization-sharing 示例为 Reachability Analyzer 启用可信访问。

```
aws ec2 enable-reachability-analyzer-organization-sharing
```

此命令不生成任何输出。

有关更多信息，请参阅《Reachability Analyzer 用户指南》中的跨账户分析。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableReachabilityAnalyzerOrganizationSharing。

enable-serial-console-access

以下代码示例演示了如何使用 enable-serial-console-access。

Amazon CLI

启用对您的账户的串行控制台的访问

以下 enable-serial-console-access 示例启用对串行控制台的账户访问。

```
aws ec2 enable-serial-console-access
```

输出：

```
{
    "SerialConsoleAccessEnabled": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 EC2 Serial Console。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableSerialConsoleAccess。

enable-snapshot-block-public-access

以下代码示例演示了如何使用 enable-snapshot-block-public-access。

Amazon CLI

启用快照的屏蔽公共访问权限

以下 enable-snapshot-block-public-access 示例将屏蔽快照的所有公开共享行为。

```
aws ec2 enable-snapshot-block-public-access \
    --state block-all-sharing
```

输出：

```
{
    "State": "block-all-sharing"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的屏蔽对快照的公共访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableSnapshotBlockPublicAccess。

enable-transit-gateway-route-table-propagation

以下代码示例演示了如何使用 enable-transit-gateway-route-table-propagation。

Amazon CLI

启用中转网关连接以将路由传播到指定的传播路由表

以下 enable-transit-gateway-route-table-propagation 示例启用指定连接，以将路由传播到指定的传播路由表。

```
aws ec2 enable-transit-gateway-route-table-propagation \
    --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE \
    --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE
```

输出：

```
{
    "Propagation": {
        "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
        "ResourceId": "vpc-4d7de228",
        "ResourceType": "vpc",
        "TransitGatewayRouteTableId": "tgw-rtb-0a823edbdeEXAMPLE",
        "State": "disabled"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableTransitGatewayRouteTablePropagation。

enable-vgw-route-propagation

以下代码示例演示了如何使用 enable-vgw-route-propagation。

Amazon CLI

启用路由传播

此示例允许指定的虚拟私有网关将静态路由传播到指定的路由表。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-
id vgw-9a4cacf3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableVgwRoutePropagation。

enable-volume-io

以下代码示例演示了如何使用 enable-volume-io。

Amazon CLI

为卷启用 I/O

此示例在卷 vol-1234567890abcdef0 上启用 I/O。

命令:

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableVolumeIo。

enable-vpc-classic-link-dns-support

以下代码示例演示了如何使用 enable-vpc-classic-link-dns-support。

Amazon CLI

为 VPC 启用 ClassicLink DNS 支持

此示例为 vpc-88888888 启用 ClassicLink DNS 支持。

命令:

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableVpcClassicLinkDnsSupport。

enable-vpc-classic-link

以下代码示例演示了如何使用 enable-vpc-classic-link。

Amazon CLI

为 VPC 启用 ClassicLink

此示例为 vpc-8888888 禁用 ClassicLink。

命令:

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableVpcClassicLink。

export-client-vpn-client-certificate-revocation-list

以下代码示例演示了如何使用 export-client-vpn-client-certificate-revocation-list。

Amazon CLI

导出客户端证书吊销列表

以下 export-client-vpn-client-certificate-revocation-list 示例导出指定 Client VPN 端点的客户端证书吊销列表。在此示例中，以文本格式返回输出，以方便您阅读。

```
aws ec2 export-client-vpn-client-certificate-revocation-list \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --output text
```

输出：

```
-----BEGIN X509 CRL-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END X509 CRL-----
STATUS      pending
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的客户端证书吊销列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportClientVpnClientCertificateRevocationList。

export-client-vpn-client-configuration

以下代码示例演示了如何使用 export-client-vpn-client-configuration。

Amazon CLI

导出客户端配置

以下 export-client-vpn-client-configuration 示例导出指定 Client VPN 端点的客户端配置。在此示例中，以文本格式返回输出，以方便您阅读。

```
aws ec2 export-client-vpn-client-configuration \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --output text
```

输出：

```
client
dev tun
```

```
proto udp
remote cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-south-1.amazonaws.com 443
remote-random-hostname
resolv-retry infinite
nobind
persist-key
persist-tun
remote-cert-tls server
cipher AES-256-GCM
verb 3
<ca>
-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
</ca>
reneg-sec 0
```

有关更多信息，请参阅《Amazon Client VPN Administrator Guide》中的 Client VPN endpoint configuration file export。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportClientVpnClientConfiguration。

`export-image`

以下代码示例演示了如何使用 `export-image`。

Amazon CLI

从 AMI 导出 VM

以下 export-image 示例以指定格式将指定的 AMI 导出到指定的存储桶。

```
aws ec2 export-image \
    --image-id ami-1234567890abcdef0 \
    --disk-image-format VMDK \
    --s3-export-location S3Bucket=my-export-bucket,S3Prefix=exports/
```

输出：

```
{
    "DiskImageFormat": "vmdk",
    "ExportImageTaskId": "export-ami-1234567890abcdef0"
    "ImageId": "ami-1234567890abcdef0",
    "RoleName": "vmimport",
    "Progress": "0",
    "S3ExportLocation": {
        "S3Bucket": "my-export-bucket",
        "S3Prefix": "exports/"
    },
    "Status": "active",
    "StatusMessage": "validating"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportImage。

get-associated-ipv6-pool-cidrs

以下代码示例演示了如何使用 get-associated-ipv6-pool-cidrs。

Amazon CLI

获取 IPv6 地址池的关联

以下 get-associated-ipv6-pool-cidrs 示例获取指定 IPv6 地址池的关联。

```
aws ec2 get-associated-ipv6-pool-cidrs \
    --pool-id ipv6pool-ec2-012345abc12345abc
```

输出：

```
{
    "Ipv6CidrAssociations": [
```

```
        {
            "Ipv6Cidr": "2001:db8:1234:1a00::/56",
            "AssociatedResource": "vpc-111111222222333ab"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAssociatedIpv6PoolCidrs。

get-aws-network-performance-data

以下代码示例演示了如何使用 get-aws-network-performance-data。

Amazon CLI

获取网络性能数据

以下 get-aws-network-performance-data 示例检索有关指定时间段内指定区域之间的网络性能的数据。

```
aws ec2 get-aws-network-performance-data \
    --start-time 2022-10-26T12:00:00.000Z \
    --end-time 2022-10-26T12:30:00.000Z \
    --data-queries Id=my-query,Source=us-east-1,Destination=eu-
west-1,Metric=aggregate-latency,Statistic=p50,Period=five-minutes
```

输出：

```
{
    "DataResponses": [
        {
            "Id": "my-query",
            "Source": "us-east-1",
            "Destination": "eu-west-1",
            "Metric": "aggregate-latency",
            "Statistic": "p50",
            "Period": "five-minutes",
            "MetricPoints": [
                {
                    "StartDate": "2022-10-26T12:00:00+00:00",
                    "EndDate": "2022-10-26T12:05:00+00:00",
                    "Value": 62.44349,
```

```
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:05:00+00:00",
                    "EndDate": "2022-10-26T12:10:00+00:00",
                    "Value": 62.483498,
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:10:00+00:00",
                    "EndDate": "2022-10-26T12:15:00+00:00",
                    "Value": 62.51248,
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:15:00+00:00",
                    "EndDate": "2022-10-26T12:20:00+00:00",
                    "Value": 62.635475,
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:20:00+00:00",
                    "EndDate": "2022-10-26T12:25:00+00:00",
                    "Value": 62.733974,
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:25:00+00:00",
                    "EndDate": "2022-10-26T12:30:00+00:00",
                    "Value": 62.773975,
                    "Status": "OK"
                },
                {
                    "StartDate": "2022-10-26T12:30:00+00:00",
                    "EndDate": "2022-10-26T12:35:00+00:00",
                    "Value": 62.75349,
                    "Status": "OK"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《基础结构性能用户指南》中的监控网络性能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAwsNetworkPerformanceData。

get-capacity-reservation-usage

以下代码示例演示了如何使用 get-capacity-reservation-usage。

Amazon CLI

查看 Amazon 账户间的容量预留使用情况

以下 get-capacity-reservation-usage 示例显示指定容量预留的使用情况信息。

```
aws ec2 get-capacity-reservation-usage \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

输出：

```
{
    "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
    "InstanceUsages": [
        {
            "UsedInstanceCount": 1,
            "AccountId": "123456789012"
        }
    ],
    "AvailableInstanceCount": 4,
    "TotalInstanceCount": 5,
    "State": "active",
    "InstanceType": "t2.medium"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的共享容量预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCapacityReservationUsage。

get-coip-pool-usage

以下代码示例演示了如何使用 get-coip-pool-usage。

Amazon CLI

获取客户拥有的 IP 地址池使用情况

以下 get-coip-pool-usage 示例获取指定的客户拥有的 IP 地址池的使用情况详情。

```
aws ec2 get-coip-pool-usage \
    --pool-id ipv4pool-coip-123a45678bEXAMPLE
```

输出：

```
{
    "CoipPoolId": "ipv4pool-coip-123a45678bEXAMPLE",
    "CoipAddressUsages": [
        {
            "CoIp": "0.0.0.0"
        },
        {
            "AllocationId": "eipalloc-123ab45c6dEXAMPLE",
            "AwsAccountId": "123456789012",
            "CoIp": "0.0.0.0"
        },
        {
            "AllocationId": "eipalloc-123ab45c6dEXAMPLE",
            "AwsAccountId": "123456789111",
            "CoIp": "0.0.0.0"
        }
    ],
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Outposts User Guide for Outposts racks》中的 Customer-owned IP addresses。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCoipPoolUsage。

get-console-output

以下代码示例演示了如何使用 get-console-output。

Amazon CLI

示例 1：获取控制台输出

以下 get-console-output 示例获取指定 Linux 实例的控制台输出。

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-07-25T21:23:53.000Z",
    "Output": "..."
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例控制台输出。

示例 2：获取最新控制台输出

以下 get-console-output 示例获取指定 Linux 实例的最新控制台输出。

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0 \
    --latest \
    --output text
```

输出：

```
i-1234567890abcdef0 [    0.000000] Command line: root=LABEL=/ console=tty1
 console=ttyS0 selinux=0 nvme_core.io_timeout=4294967295
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point
 registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
...
Cloud-init v. 0.7.6 finished at Wed, 09 May 2018 19:01:13 +0000. Datasource
 DataSourceEc2.  Up 21.50 seconds
Amazon Linux AMI release 2018.03
Kernel 4.14.26-46.32.amzn1.x
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例控制台输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConsoleOutput。

get-console-screenshot

以下代码示例演示了如何使用 get-console-screenshot。

Amazon CLI

检索正在运行实例的屏幕截图

以下 get-console-screenshot 示例以 .jpg 格式检索指定实例的屏幕截图。屏幕截图以 Base64 编码字符串的形式返回。

```
aws ec2 get-console-screenshot \
    --instance-id i-1234567890abcdef0
```

输出：

```
{
    "ImageData": "997987/8kgj49ikjhewkwwe0008084EXAMPLE",
    "InstanceId": "i-1234567890abcdef0"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConsoleScreenshot。

get-default-credit-specification

以下代码示例演示了如何使用 get-default-credit-specification。

Amazon CLI

描述默认积分选项

以下 get-default-credit-specification 示例描述了 T2 实例的默认积分选项。

```
aws ec2 get-default-credit-specification \
    --instance-family t2
```

输出：

```
{
    "InstanceFamilyCreditSpecification": {
        "InstanceFamily": "t2",
```

```
        "CpuCredits": "standard"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDefaultCreditSpecification。

get-ebs-default-kms-key-id

以下代码示例演示了如何使用 get-ebs-default-kms-key-id。

Amazon CLI

描述 EBS 加密的默认 CMK

以下 get-ebs-default-kms-key-id 示例描述了 Amazon 账户的 EBS 加密的默认 CMK。

```
aws ec2 get-ebs-default-kms-key-id
```

该输出显示 EBS 加密的默认 CMK，它是别名为 alias/aws/ebs 的 Amazon 托管 CMK。

```
{
    "KmsKeyId": "alias/aws/ebs"
}
```

以下输出显示了 EBS 加密的自定义 CMK。

```
{
    "KmsKeyId": "arn:aws:kms:us-
west-2:123456789012:key/0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEbsDefaultKmsKeyId。

get-ebs-encryption-by-default

以下代码示例演示了如何使用 get-ebs-encryption-by-default。

Amazon CLI

描述默认情况下是否启用 EBS 加密

以下 get-ebs-encryption-by-default 示例指示是否默认为当前区域中的 Amazon 账户启用 EBS 加密。

```
aws ec2 get-ebs-encryption-by-default
```

以下输出表明 EBS 加密在默认情况下处于禁用状态。

```
{
    "EbsEncryptionByDefault": false
}
```

以下输出指示默认情况下启用 EBS 加密。

```
{
    "EbsEncryptionByDefault": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEbsEncryptionByDefault。

get-flow-logs-integration-template

以下代码示例演示了如何使用 get-flow-logs-integration-template。

Amazon CLI

创建 CloudFormation 模板以自动将 VPC 流日志与 Amazon Athena 集成

以下 get-flow-logs-integration-template 示例创建一个 CloudFormation 模板，以自动将 VPC 流日志与 Amazon Athena 集成。

Linux：

```
aws ec2 get-flow-logs-integration-template \
    --flow-log-id fl-1234567890abcdef0 \
    --config-delivery-s3-destination-arn arn:aws:s3:::amzn-s3-demo-bucket \
    --integrate-services
 AthenaIntegrations='[{IntegrationResultS3DestinationArn=arn:aws:s3:::amzn-s3-demo-
bucket,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
{IntegrationResultS3DestinationArn=arn:aws:s3:::amzn-s3-demo-
bucket,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
```

Windows:

```
aws ec2 get-flow-logs-integration-template ^
    --flow-log-id fl-1234567890abcdef0 ^
    --config-delivery-s3-destination-arn arn:aws:s3:::amzn-s3-demo-bucket ^
    --integrate-
services AthenaIntegrations=[{IntegrationResultS3DestinationArn=arn:aws:s3:::amzn-
s3-demo-
bucket,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
{IntegrationResultS3DestinationArn=arn:aws:s3:::amzn-s3-demo-
bucket,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
```

输出：

```
{
    "Result": "https://amzn-s3-demo-bucket.s3.us-east-2.amazonaws.com/
VPCFlowLogsIntegrationTemplate_fl-1234567890abcdef0_Wed%20Jul
%2021%2000%3A57%3A56%20UTC%202021.yml"
}
```

有关使用 CloudFormation 模板的信息，请参阅《Amazon CloudFormation 用户指南》中的使用 Amazon CloudFormation 模板。

有关使用 Amazon Athena 和流日志的信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的使用 Amazon Athena 查询流日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFlowLogsIntegrationTemplate。

get-groups-for-capacity-reservation

以下代码示例演示了如何使用 get-groups-for-capacity-reservation。

Amazon CLI

列出具有容量预留的资源组

以下 get-groups-for-capacity-reservation 示例列出了向其中添加了指定容量预留的资源组。

```
aws ec2 get-groups-for-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

输出：

```
{
    "CapacityReservationsGroup": [
        {
            "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/my-
resource-group",
            "OwnerId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的容量预留组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupsForCapacityReservation。

get-host-reservation-purchase-preview

以下代码示例演示了如何使用 get-host-reservation-purchase-preview。

Amazon CLI

获取专属主机预留的购买预览

此示例提供您账户中指定专属主机的指定专属主机预留成本的预览。

命令:

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324 --
host-id-set h-013abcd2a00cbd123
```

输出：

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
```

```
            ],
            "UpfrontPrice": "0.000",
            "Duration": 31536000
        }
    ],
    "TotalUpfrontPrice": "0.000"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetHostReservationPurchasePreview。

get-image-block-public-access-state

以下代码示例演示了如何使用 get-image-block-public-access-state。

Amazon CLI

获取指定区域中 AMI 的屏蔽公共访问状态

以下 get-image-block-public-access-state 示例在指定区域中的账户级别获取 AMI 的屏蔽公共访问状态。

```
aws ec2 get-image-block-public-access-state \
    --region us-east-1
```

输出：

```
{
    "ImageBlockPublicAccessState": "block-new-sharing"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的屏蔽对 AMI 的公共访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImageBlockPublicAccessState。

get-instance-types-from-instance-requirements

以下代码示例演示了如何使用 get-instance-types-from-instance-requirements。

Amazon CLI

预览与指定属性匹配的实例类型

以下 get-instance-types-from-instance-requirements 示例首先生成一个可以使用 --generate-cli-skeleton 参数指定的所有可能属性的列表，然后将该列表保存到 JSON 文件中。然后，使用该 JSON 文件自定义要预览其匹配实例类型的属性。

要生成所有可能的属性并将输出直接保存到 JSON 文件中，请使用以下命令。

```
aws ec2 get-instance-types-from-instance-requirements \
    --region us-east-1 \
    --generate-cli-skeleton input > attributes.json
```

输出：

```
{
    "DryRun": true,
    "ArchitectureTypes": [
        "x86_64_mac"
    ],
    "VirtualizationTypes": [
        "paravirtual"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 0,
            "Max": 0
        },
        "MemoryMiB": {
            "Min": 0,
            "Max": 0
        },
        "CpuManufacturers": [
            "intel"
        ],
        "MemoryGiBPerVCpu": {
            "Min": 0.0,
            "Max": 0.0
        },
        "ExcludedInstanceTypes": [
            ""
        ],
        "InstanceGenerations": [
            "current"
        ],
        "SpotMaxPricePercentageOverLowestPrice": 0,
```

```
        "OnDemandMaxPricePercentageOverLowestPrice": 0,
        "BareMetal": "included",
        "BurstablePerformance": "excluded",
        "RequireHibernateSupport": true,
        "NetworkInterfaceCount": {
            "Min": 0,
            "Max": 0
        },
        "LocalStorage": "required",
        "LocalStorageTypes": [
            "hdd"
        ],
        "TotalLocalStorageGB": {
            "Min": 0.0,
            "Max": 0.0
        },
        "BaselineEbsBandwidthMbps": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorTypes": [
            "inference"
        ],
        "AcceleratorCount": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorManufacturers": [
            "xilinx"
        ],
        "AcceleratorNames": [
            "t4"
        ],
        "AcceleratorTotalMemoryMiB": {
            "Min": 0,
            "Max": 0
        }
    },
    "MaxResults": 0,
    "NextToken": ""
}
```

配置 JSON 文件。您必须提供 ArchitectureTypes、VirtualizationTypes、VCpuCount 和 MemoryMiB 的值。您可以省略其他属性。省略时，将使用默认值。有关每个属性及其默认值的描述，请参阅 get-instance-types-from-instance-requirements <https://docs.aws.amazon.com/cli/latest/reference/ec2/get-instance-types-from-instance-requirements.html>。

预览具有 attributes.json 中指定属性的实例类型。使用 --cli-input-json 参数指定 JSON 文件的名称和路径。在以下请求中，输出采用表格形式。

```
aws ec2 get-instance-types-from-instance-requirements \
    --cli-input-json file://attributes.json \
    --output table
```

attributes.json 文件的内容：

```
{

    "ArchitectureTypes": [
        "x86_64"
    ],
    "VirtualizationTypes": [
        "hvm"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 4,
            "Max": 6
        },
        "MemoryMiB": {
            "Min": 2048
        },
        "InstanceGenerations": [
            "current"
        ]
    }
}
```

输出：

```
-----------------------------------------
|GetInstanceTypesFromInstanceRequirements|
+---------------------------------------+
||             InstanceTypes             ||
```

```
|+--------------------------------------+|
||             InstanceType             ||
|+--------------------------------------+|
||  c4.xlarge                           ||
||  c5.xlarge                           ||
||  c5a.xlarge                          ||
||  c5ad.xlarge                         ||
||  c5d.xlarge                          ||
||  c5n.xlarge                          ||
||  d2.xlarge                           ||
...
```

有关基于属性的实例类型选择的更多信息，请参阅《Amazon EC2 用户指南》中的基于属性的实例类型选择的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceTypesFromInstanceRequirements。

get-instance-uefi-data

以下代码示例演示了如何使用 get-instance-uefi-data。

Amazon CLI

从实例中检索 UEFI 数据

以下 get-instance-uefi-data 示例从实例中检索 UEFI 数据。如果输出为空，则实例不包含 UEFI 数据。

```
aws ec2 get-instance-uefi-data \
    --instance-id i-0123456789example
```

输出：

```
{
    "InstanceId": "i-0123456789example",
    "UefiData": "QU1aTlVFRkkf+uLXAAAAAHj5a7fZ9+3dBzxXb/.
    <snipped>
    AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD4L/J/AODshho="
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 UEFI 安全引导。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceUefiData。

get-ipam-address-history

以下代码示例演示了如何使用 get-ipam-address-history。

Amazon CLI

获取 CIDR 的历史记录

以下 get-ipam-address-history 示例获取 CIDR 的历史记录。

（Linux）：

```
aws ec2 get-ipam-address-history \
   --cidr 10.0.0.0/16 \
   --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \
   --start-time 2021-12-08T01:00:00.000Z \
   --end-time 2021-12-10T01:00:00.000Z
```

（Windows）：

```
aws ec2 get-ipam-address-history ^
   --cidr 10.0.0.0/16 ^
   --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^
   --start-time 2021-12-08T01:00:00.000Z ^
   --end-time 2021-12-10T01:00:00.000Z
```

输出：

```
{
    "HistoryRecords": [
        {
            "ResourceOwnerId": "123456789012",
            "ResourceRegion": "us-west-1",
            "ResourceType": "vpc",
            "ResourceId": "vpc-06cbefa9ee907e1c0",
            "ResourceCidr": "10.0.0.0/16",
            "ResourceName": "Demo",
            "ResourceComplianceStatus": "unmanaged",
            "ResourceOverlapStatus": "overlapping",
```

```
            "VpcId": "vpc-06cbefa9ee907e1c0",
            "SampledStartTime": "2021-12-08T19:54:57.675000+00:00"
        },
        {
            "ResourceOwnerId": "123456789012",
            "ResourceRegion": "us-east-2",
            "ResourceType": "vpc",
            "ResourceId": "vpc-042702f474812c9ad",
            "ResourceCidr": "10.0.0.0/16",
            "ResourceName": "test",
            "ResourceComplianceStatus": "unmanaged",
            "ResourceOverlapStatus": "overlapping",
            "VpcId": "vpc-042702f474812c9ad",
            "SampledStartTime": "2021-12-08T19:54:59.019000+00:00"
        },
        {
            "ResourceOwnerId": "123456789012",
            "ResourceRegion": "us-east-2",
            "ResourceType": "vpc",
            "ResourceId": "vpc-042b8a44f64267d67",
            "ResourceCidr": "10.0.0.0/16",
            "ResourceName": "tester",
            "ResourceComplianceStatus": "unmanaged",
            "ResourceOverlapStatus": "overlapping",
            "VpcId": "vpc-042b8a44f64267d67",
            "SampledStartTime": "2021-12-08T19:54:59.019000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的查看 IP 地址的历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamAddressHistory。

get-ipam-discovered-accounts

以下代码示例演示了如何使用 get-ipam-discovered-accounts。

Amazon CLI

查看由 IPAM 发现的账户

在此场景中，您是 IPAM 委派管理员，您想要查看拥有 IPAM 正在发现的资源的 Amazon 账户。

--discovery-region 是您要在其中查看受监控账户状态的 IPAM 运营区域。例如，如果您有三个 IPAM 运营区域，您可能需要发出三次此请求，才能查看其中每个特定区域中特定于发现的时间戳。

以下 get-ipam-discovered-accounts 示例列出了拥有 IPAM 正在发现的资源的 Amazon 账户。

```
aws ec2 get-ipam-discovered-accounts \
   --ipam-resource-discovery-id ipam-res-disco-0365d2977fc1672fe \
   --discovery-region us-east-1
```

输出：

```
{
    "IpamDiscoveredAccounts": [
        {
            "AccountId": "149977607591",
            "DiscoveryRegion": "us-east-1",
            "LastAttemptedDiscoveryTime": "2024-02-09T19:04:31.379000+00:00",
            "LastSuccessfulDiscoveryTime": "2024-02-09T19:04:31.379000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 IPAM 与组织外部的账户集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamDiscoveredAccounts。

get-ipam-discovered-public-addresses

以下代码示例演示了如何使用 get-ipam-discovered-public-addresses。

Amazon CLI

查看发现的公有 IP 地址

在此示例中，您是 IPAM 委派管理员，您想要查看由 IPAM 发现的资源的 IP 地址。您可以使用 describe-ipam-resource-discoveries 获取资源发现 ID。

以下 get-ipam-discovered-public-addresses 示例显示了针对资源发现而发现的公有 IP 地址。

```
aws ec2 get-ipam-discovered-public-addresses \
    --ipam-resource-discovery-id ipam-res-disco-0f4ef577a9f37a162 \
    --address-region us-east-1 \
    --region us-east-1
```

输出：

```
{
    "IpamDiscoveredPublicAddresses": [
        {
           "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",
            "AddressRegion": "us-east-1",
            "Address": "54.208.155.7",
            "AddressOwnerId": "320805250157",
            "AssociationStatus": "associated",
            "AddressType": "ec2-public-ip",
            "VpcId": "vpc-073b294916198ce49",
            "SubnetId": "subnet-0b6c8a8839e9a4f15",
            "NetworkInterfaceId": "eni-081c446b5284a5e06",
            "NetworkInterfaceDescription": "",
            "InstanceId": "i-07459a6fca5b35823",
            "Tags": {},
            "NetworkBorderGroup": "us-east-1c",
            "SecurityGroups": [
                {
                    "GroupName": "launch-wizard-2",
                    "GroupId": "sg-0a489dd6a65c244ce"
                }
            ],
            "SampleTime": "2024-04-05T15:13:59.228000+00:00"
        },
        {
            "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",
            "AddressRegion": "us-east-1",
            "Address": "44.201.251.218",
            "AddressOwnerId": "470889052923",
            "AssociationStatus": "associated",
            "AddressType": "ec2-public-ip",
            "VpcId": "vpc-6c31a611",
            "SubnetId": "subnet-062f47608b99834b1",
            "NetworkInterfaceId": "eni-024845359c2c3ae9b",
            "NetworkInterfaceDescription": "",
            "InstanceId": "i-04ef786d9c4e03f41",
```

```
            "Tags": {},
            "NetworkBorderGroup": "us-east-1a",
            "SecurityGroups": [
                {
                    "GroupName": "launch-wizard-32",
                    "GroupId": "sg-0ed1a426e96a68374"
                }
            ],
            "SampleTime": "2024-04-05T15:13:59.145000+00:00"
        }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的查看公共 IP 洞察功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamDiscoveredPublicAddresses。

get-ipam-discovered-resource-cidrs

以下代码示例演示了如何使用 get-ipam-discovered-resource-cidrs。

Amazon CLI

查看由 IPAM 发现的 IP 地址 CIDR

在此示例中，您是 IPAM 委派管理员，想要查看与 IPAM 正在发现的资源的 IP 地址 CIDR 相关的详细信息。

要完成此请求，请执行以下操作：

您选择的资源发现必须与 IPAM 相关联。--resource-region 是在其中创建资源的 Amazon 区域。

以下 get-ipam-discovered-resource-cidrs 示例列出了 IPAM 正在发现的资源的 IP 地址。

```
aws ec2 get-ipam-discovered-resource-cidrs \
   --ipam-resource-discovery-id ipam-res-disco-0365d2977fc1672fe \
   --resource-region us-east-1
```

输出：

```
{
    {
```

```
        "IpamDiscoveredResourceCidrs": [
        {
            "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
            "ResourceRegion": "us-east-1",
            "ResourceId": "vpc-0c974c95ca7ceef4a",
            "ResourceOwnerId": "149977607591",
            "ResourceCidr": "172.31.0.0/16",
            "ResourceType": "vpc",
            "ResourceTags": [],
            "IpUsage": 0.375,
            "VpcId": "vpc-0c974c95ca7ceef4a",
            "SampleTime": "2024-02-09T19:15:16.529000+00:00"
        },
        {
            "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
            "ResourceRegion": "us-east-1",
            "ResourceId": "subnet-07fe028119082a8c1",
            "ResourceOwnerId": "149977607591",
            "ResourceCidr": "172.31.0.0/20",
            "ResourceType": "subnet",
            "ResourceTags": [],
            "IpUsage": 0.0012,
            "VpcId": "vpc-0c974c95ca7ceef4a",
            "SampleTime": "2024-02-09T19:15:16.529000+00:00"
        },
        {
            "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
            "ResourceRegion": "us-east-1",
            "ResourceId": "subnet-0a96893763984cc4e",
            "ResourceOwnerId": "149977607591",
            "ResourceCidr": "172.31.64.0/20",
            "ResourceType": "subnet",
            "ResourceTags": [],
            "IpUsage": 0.0012,
            "VpcId": "vpc-0c974c95ca7ceef4a",
            "SampleTime": "2024-02-09T19:15:16.529000+00:00"
        }
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的按资源监控 CIDR 使用情况。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamDiscoveredResourceCidrs。

get-ipam-pool-allocations

以下代码示例演示了如何使用 get-ipam-pool-allocations。

Amazon CLI

获取从 IPAM 池分配的 CIDR

以下 get-ipam-pool-allocations 示例获取从 IPAM 池分配的 CIDR。

（Linux）：

```
aws ec2 get-ipam-pool-allocations \
    --ipam-pool-id ipam-pool-0533048da7d823723 \
    --filters Name=ipam-pool-allocation-id,Values=ipam-pool-
alloc-0e6186d73999e47389266a5d6991e6220
```

（Windows）：

```
aws ec2 get-ipam-pool-allocations ^
    --ipam-pool-id ipam-pool-0533048da7d823723 ^
    --filters Name=ipam-pool-allocation-id,Values=ipam-pool-
alloc-0e6186d73999e47389266a5d6991e6220
```

输出：

```
{
    "IpamPoolAllocations": [
        {
            "Cidr": "10.0.0.0/16",
            "IpamPoolAllocationId": "ipam-pool-
alloc-0e6186d73999e47389266a5d6991e6220",
            "ResourceType": "custom",
            "ResourceOwner": "123456789012"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamPoolAllocations。

get-ipam-pool-cidrs

以下代码示例演示了如何使用 get-ipam-pool-cidrs。

Amazon CLI

获取预置到 IPAM 池的 CIDR

以下 get-ipam-pool-cidrs 示例获取预置到 IPAM 池的 CIDR。

（Linux）：

```
aws ec2 get-ipam-pool-cidrs \
    --ipam-pool-id ipam-pool-0533048da7d823723 \
    --filters 'Name=cidr,Values=10.*'
```

（Windows）：

```
aws ec2 get-ipam-pool-cidrs ^
    --ipam-pool-id ipam-pool-0533048da7d823723 ^
    --filters Name=cidr,Values=10.*
```

输出：

```
{
    "IpamPoolCidr": {
        "Cidr": "10.0.0.0/24",
        "State": "provisioned"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamPoolCidrs。

get-ipam-resource-cidrs

以下代码示例演示了如何使用 get-ipam-resource-cidrs。

Amazon CLI

获取分配给资源的 CIDR

以下 get-ipam-resource-cidrs 示例获取分配给资源的 CIDR。

（Linux）：

```
aws ec2 get-ipam-resource-cidrs \
    --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \
    --filters Name=management-state,Values=unmanaged
```

（Windows）：

```
aws ec2 get-ipam-resource-cidrs ^
    --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^
    --filters Name=management-state,Values=unmanaged
```

输出：

```
{
    "IpamResourceCidrs": [
        {
            "IpamId": "ipam-08440e7a3acde3908",
            "IpamScopeId": "ipam-scope-02fc38cd4c48e7d38",
            "ResourceRegion": "us-east-2",
            "ResourceOwnerId": "123456789012",
            "ResourceId": "vpc-621b8709",
            "ResourceName": "Default AWS VPC",
            "ResourceCidr": "172.33.0.0/16",
            "ResourceType": "vpc",
            "ResourceTags": [
                {
                    "Key": "Environment",
                    "Value": "Test"
                },
                {
                    "Key": "Name",
                    "Value": "Default AWS VPC"
                }
            ],
            "IpUsage": 0.0039,
            "ComplianceStatus": "unmanaged",
            "ManagementState": "unmanaged",
            "OverlapStatus": "nonoverlapping",
            "VpcId": "vpc-621b8709"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的按资源监控 CIDR 使用情况。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpamResourceCidrs。

get-launch-template-data

以下代码示例演示了如何使用 get-launch-template-data。

Amazon CLI

获取启动模板的实例数据

此示例将获取有关指定实例的数据，并使用 --query 选项返回 LaunchTemplateData 中的内容。您可以将输出作为基础以创建新的启动模板或启动模板版本。

命令:

```
aws ec2 get-launch-template-data --instance-id i-0123d646e8048babc --query
 'LaunchTemplateData'
```

输出：

```
{
      "Monitoring": {},
      "ImageId": "ami-8c1be5f6",
      "BlockDeviceMappings": [
          {
              "DeviceName": "/dev/xvda",
              "Ebs": {
                  "DeleteOnTermination": true
              }
          }
      ],
      "EbsOptimized": false,
      "Placement": {
          "Tenancy": "default",
          "GroupName": "",
          "AvailabilityZone": "us-east-1a"
      },
      "InstanceType": "t2.micro",
      "NetworkInterfaces": [
```

```
            {
                "Description": "",
                "NetworkInterfaceId": "eni-35306abc",
                "PrivateIpAddresses": [
                    {
                        "Primary": true,
                        "PrivateIpAddress": "10.0.0.72"
                    }
                ],
                "SubnetId": "subnet-7b16de0c",
                "Groups": [
                    "sg-7c227019"
                ],
                "Ipv6Addresses": [
                    {
                        "Ipv6Address": "2001:db8:1234:1a00::123"
                    }
                ],
                "PrivateIpAddress": "10.0.0.72"
            }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLaunchTemplateData。

get-managed-prefix-list-associations

以下代码示例演示了如何使用 get-managed-prefix-list-associations。

Amazon CLI

获取前缀列表关联

以下 get-managed-prefix-list-associations 示例将获取与指定前缀列表关联的资源。

```
aws ec2 get-managed-prefix-list-associations \
    --prefix-list-id pl-0123456abcabcabc1
```

输出：

```
{
    "PrefixListAssociations": [
```

```
        {
            "ResourceId": "sg-0abc123456abc12345",
            "ResourceOwner": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetManagedPrefixListAssociations。

get-managed-prefix-list-entries

以下代码示例演示了如何使用 get-managed-prefix-list-entries。

Amazon CLI

获取前缀列表的条目

以下 get-managed-prefix-list-entries 将获取指定前缀列表的条目。

```
aws ec2 get-managed-prefix-list-entries \
    --prefix-list-id pl-0123456abcabcabc1
```

输出：

```
{
    "Entries": [
        {
            "Cidr": "10.0.0.0/16",
            "Description": "vpc-a"
        },
        {
            "Cidr": "10.2.0.0/16",
            "Description": "vpc-b"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetManagedPrefixListEntries。

get-network-insights-access-scope-analysis-findings

以下代码示例演示了如何使用 get-network-insights-access-scope-analysis-findings。

Amazon CLI

获取 Network Insights 访问范围分析的结果

以下 get-network-insights-access-scope-analysis-findings 示例将获取您 Amazon 账户中的选定范围分析结果。

```
aws ec2 get-network-insights-access-scope-analysis-findings \
    --region us-east-1 \
    --network-insights-access-scope-analysis-id nis \
    --nis-123456789111
```

输出：

```
{
    "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
    "AnalysisFindings": [
        {
            "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
            "NetworkInsightsAccessScopeId": "nis-123456789111",
            "FindingComponents": [
                {
                    "SequenceNumber": 1,
                    "Component": {
                        "Id": "eni-02e3d42d5cceca67d",
                        "Arn": "arn:aws:ec2:us-east-1:936459623503:network-
interface/eni-02e3d32d9cceca17d"
                    },
                    "OutboundHeader": {
                        "DestinationAddresses": [
                            "0.0.0.0/5",
                            "11.0.0.0/8",
                            "12.0.0.0/6",
                            "128.0.0.0/3",
                            "16.0.0.0/4",
                            "160.0.0.0/5",
                            "168.0.0.0/6",
                            "172.0.0.0/12"
                            "8.0.0.0/7"
```

```
                        ],
                        "DestinationPortRanges": [
                            {
                                "From": 0,
                                "To": 65535
                            }
                        ],
                        "Protocol": "6",
                        "SourceAddresses": [
                            "10.0.2.253/32"
                        ],
                        "SourcePortRanges": [
                            {
                                "From": 0,
                                "To": 65535
                            }
                        ]
                    }, [etc]
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetNetworkInsightsAccessScopeAnalysisFindings。

get-network-insights-access-scope-content

以下代码示例演示了如何使用 get-network-insights-access-scope-content。

Amazon CLI

获取 Network Insights 访问范围内容

以下 get-network-insights-access-scope-content 示例将获取您 Amazon 账户中选定范围分析 ID 的内容。

```
aws ec2 get-network-insights-access-scope-content \
    --region us-east-1 \
    --network-insights-access-scope-id nis-123456789222
```

输出：

```
{
    "NetworkInsightsAccessScopeContent": {
        "NetworkInsightsAccessScopeId": "nis-123456789222",
        "MatchPaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "ResourceTypes": [
                            "AWS::EC2::NetworkInterface"
                        ]
                    }
                },
                "Destination": {
                    "ResourceStatement": {
                        "ResourceTypes": [
                            "AWS::EC2::InternetGateway"
                        ]
                    }
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetNetworkInsightsAccessScopeContent。

get-password-data

以下代码示例演示了如何使用 get-password-data。

Amazon CLI

获取加密密码

此示例将获取加密密码。

命令:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-07T22:18:38.000Z",
    "PasswordData": "gSlJFq+VpcZXqy+iktxMF6NyxQ4qCrT4+gaOuNOenX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTYP7WmU3TUnhsuBd+p6LVk7T2lKUml6OXbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcpRFigzhniQgDDeO1InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNVelVz/
53TkDtxbNoU606M1gK9zUWSxqEgwvbV2j8c5rP0WCuaMWSFl4ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
DPGzKrF5yLlf3etP2L4ZR6CvG7K1hx7VKOQVN32Dajw=="
}
```

获取解密密码

此示例将获取解密密码。

命令:

```
aws ec2 get-password-data --instance-id  i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-30T23:18:05.000Z",
    "PasswordData": "&ViJ652e*u"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPasswordData。

get-reserved-instances-exchange-quote

以下代码示例演示了如何使用 get-reserved-instances-exchange-quote。

Amazon CLI

获取交换可转换预留实例的报价

此示例将获取指定可转换预留实例的交换信息。

命令:

```
aws ec2 get-reserved-instances-exchange-quote --reserved-
instance-ids 7b8750c3-397e-4da4-bbcb-a45ebexample --target-
configurations OfferingId=6fea5434-b379-434c-b07b-a7abexample
```

输出：

```
{
  "CurrencyCode": "USD",
  "ReservedInstanceValueSet": [
      {
          "ReservedInstanceId": "7b8750c3-397e-4da4-bbcb-a45ebexample",
          "ReservationValue": {
              "RemainingUpfrontValue": "0.000000",
              "HourlyPrice": "0.027800",
              "RemainingTotalValue": "730.556200"
          }
      }
  ],
  "PaymentDue": "424.983828",
  "TargetConfigurationValueSet": [
      {
          "TargetConfiguration": {
              "InstanceCount": 5,
              "OfferingId": "6fea5434-b379-434c-b07b-a7abexample"
          },
          "ReservationValue": {
              "RemainingUpfrontValue": "424.983828",
              "HourlyPrice": "0.016000",
              "RemainingTotalValue": "845.447828"
          }
      }
  ],
  "IsValidExchange": true,
  "OutputReservedInstancesWillExpireAt": "2020-10-01T13:03:39Z",
  "ReservedInstanceValueRollup": {
      "RemainingUpfrontValue": "0.000000",
      "HourlyPrice": "0.027800",
      "RemainingTotalValue": "730.556200"
  },
  "TargetConfigurationValueRollup": {
      "RemainingUpfrontValue": "424.983828",
```

```
      "HourlyPrice": "0.016000",
      "RemainingTotalValue": "845.447828"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReservedInstancesExchangeQuote。

get-security-groups-for-vpc

以下代码示例演示了如何使用 get-security-groups-for-vpc。

Amazon CLI

。查看可与指定 VPC 中的网络接口相关联的安全组。

以下 get-security-groups-for-vpc 示例显示了可以与 VPC 中的网络接口相关联的安全组。

```
aws ec2 get-security-groups-for-vpc \
    --vpc-id vpc-6c31a611 \
    --region us-east-1
```

输出：

```
{
    "SecurityGroupForVpcs": [
        {
            "Description": "launch-wizard-36 created 2022-08-29T15:59:35.338Z",
            "GroupName": "launch-wizard-36",
            "OwnerId": "470889052923",
            "GroupId": "sg-007e0c3027ee885f5",
            "Tags": [],
            "PrimaryVpcId": "vpc-6c31a611"
        },
        {
            "Description": "launch-wizard-18 created 2024-01-19T20:22:27.527Z",
            "GroupName": "launch-wizard-18",
            "OwnerId": "470889052923",
            "GroupId": "sg-0147193bef51c9eef",
            "Tags": [],
            "PrimaryVpcId": "vpc-6c31a611"
        }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSecurityGroupsForVpc。

get-serial-console-access-status

以下代码示例演示了如何使用 get-serial-console-access-status。

Amazon CLI

查看账户对串行控制台的访问权限状态

以下 get-serial-console-access-status 示例确定是否为您的账户启用串行控制台访问权限。

```
aws ec2 get-serial-console-access-status
```

输出：

```
{
    "SerialConsoleAccessEnabled": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 EC2 Serial Console。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSerialConsoleAccessStatus。

get-snapshot-block-public-access-state

以下代码示例演示了如何使用 get-snapshot-block-public-access-state。

Amazon CLI

获取快照的屏蔽公共访问权限的当前状态

以下 get-snapshot-block-public-access-state 示例获取快照的屏蔽公共访问权限的当前状态。

```
aws ec2 get-snapshot-block-public-access-state
```

输出：

```
{
    "State": "block-all-sharing"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的屏蔽对快照的公共访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSnapshotBlockPublicAccessState。

get-spot-placement-scores

以下代码示例演示了如何使用 get-spot-placement-scores。

Amazon CLI

计算指定要求的竞价置放分数

以下 get-spot-placement-scores 示例首先生成一个列表，其中包含可以使用 --generate-cli-skeleton 参数为竞价置放分数配置指定的所有可能参数，并将该列表保存到 JSON 文件中。然后，使用该 JSON 文件配置用于计算竞价置放分数的要求。

生成可以为竞价置放分数配置指定的所有可能参数，并将输出直接保存到 JSON 文件中。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --generate-cli-skeleton input > attributes.json
```

输出：

```
{
    "InstanceTypes": [
        ""
    ],
    "TargetCapacity": 0,
    "TargetCapacityUnitType": "vcpu",
    "SingleAvailabilityZone": true,
    "RegionNames": [
        ""
    ],
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": [
            "x86_64_mac"
```

```
        ],
        "VirtualizationTypes": [
            "hvm"
        ],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 0,
                "Max": 0
            },
            "MemoryMiB": {
                "Min": 0,
                "Max": 0
            },
            "CpuManufacturers": [
                "amd"
            ],
            "MemoryGiBPerVCpu": {
                "Min": 0.0,
                "Max": 0.0
            },
            "ExcludedInstanceTypes": [
                ""
            ],
            "InstanceGenerations": [
                "previous"
            ],
            "SpotMaxPricePercentageOverLowestPrice": 0,
            "OnDemandMaxPricePercentageOverLowestPrice": 0,
            "BareMetal": "excluded",
            "BurstablePerformance": "excluded",
            "RequireHibernateSupport": true,
            "NetworkInterfaceCount": {
                "Min": 0,
                "Max": 0
            },
            "LocalStorage": "included",
            "LocalStorageTypes": [
                "hdd"
            ],
            "TotalLocalStorageGB": {
                "Min": 0.0,
                "Max": 0.0
            },
            "BaselineEbsBandwidthMbps": {
```

```
                "Min": 0,
                "Max": 0
            },
            "AcceleratorTypes": [
                "fpga"
            ],
            "AcceleratorCount": {
                "Min": 0,
                "Max": 0
            },
            "AcceleratorManufacturers": [
                "amd"
            ],
            "AcceleratorNames": [
                "vu9p"
            ],
            "AcceleratorTotalMemoryMiB": {
                "Min": 0,
                "Max": 0
            }
        }
    },
    "DryRun": true,
    "MaxResults": 0,
    "NextToken": ""
}
```

配置 JSON 文件。您必须为 TargetCapacity 提供一个值。有关每个参数及其默认值的描述，请参阅“计算竞价置放分数（Amazon CLI）”<https://docs.aws.amazon.com/AmazonEC2/latest/UserGuide/spot-placement-score.html#calculate-sps-cli>。

计算 attributes.json 中指定要求的竞价置放分数。使用 --cli-input-json 参数指定 JSON 文件的名称和路径。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --cli-input-json file://attributes.json
```

SingleAvailabilityZone 设置为 false 或省略时的输出（如果省略，则默认为 false）。返回区域的评分列表。

```
"Recommendation": [
```

```
    {
        "Region": "us-east-1",
        "Score": 7
    },
    {
        "Region": "us-west-1",
        "Score": 5
    },
   ...
```

SingleAvailabilityZone 设置为 true 时的输出。返回单个可用区的评分列表。

```
"Recommendation": [
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "use1-az1"
        "Score": 8
    },
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "usw2-az3"
        "Score": 6
    },
   ...
```

有关计算竞价置放分数以及示例配置的更多信息，请参阅《Amazon EC2 用户指南》中的计算竞价置放分数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSpotPlacementScores。

get-subnet-cidr-reservations

以下代码示例演示了如何使用 get-subnet-cidr-reservations。

Amazon CLI

获取有关子网 CIDR 预留的信息

以下 get-subnet-cidr-reservations 示例显示有关指定子网 CIDR 预留的信息。

```
aws ec2 get-subnet-cidr-reservations \
    --subnet-id subnet-03c51e2e6cEXAMPLE
```

输出：

```
{
    "SubnetIpv4CidrReservations": [
        {
            "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",
            "SubnetId": "subnet-03c51e2e6cEXAMPLE",
            "Cidr": "10.1.0.16/28",
            "ReservationType": "prefix",
            "OwnerId": "123456789012"
        }
    ],
    "SubnetIpv6CidrReservations": []
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的子网 CIDR 预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubnetCidrReservations。

get-transit-gateway-attachment-propagations

以下代码示例演示了如何使用 get-transit-gateway-attachment-propagations。

Amazon CLI

列出指定的资源连接将路由传播到的路由表

以下 get-transit-gateway-attachment-propagations 示例列出了指定的资源连接将路由传播到的路由表。

```
aws ec2 get-transit-gateway-attachment-propagations \
    --transit-gateway-attachment-id tgw-attach-09fbd47ddfEXAMPLE
```

输出：

```
{
    "TransitGatewayAttachmentPropagations": [
        {
            "TransitGatewayRouteTableId": "tgw-rtb-0882c61b97EXAMPLE",
            "State": "enabled"
        }
    ]
```

```
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayAttachmentPropagations。

get-transit-gateway-multicast-domain-associations

以下代码示例演示了如何使用 get-transit-gateway-multicast-domain-associations。

Amazon CLI

查看有关中转网关组播域关联的信息

以下 get-transit-gateway-multicast-domain-associations 示例返回指定组播域的关联。

```
aws ec2 get-transit-gateway-multicast-domain-associations \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

输出：

```
{
    "MulticastDomainAssociations": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-028c1dd0f8EXAMPLE",
            "ResourceId": "vpc-01128d2c24EXAMPLE",
            "ResourceType": "vpc",
            "Subnet": {
                "SubnetId": "subnet-000de86e3bEXAMPLE",
                "State": "associated"
            }
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
            "ResourceId": "vpc-7EXAMPLE",
            "ResourceType": "vpc",
            "Subnet": {
                "SubnetId": "subnet-4EXAMPLE",
                "State": "associated"
            }
```

```
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
            "ResourceId": "vpc-7EXAMPLE",
            "ResourceType": "vpc",
            "Subnet": {
                "SubnetId": "subnet-5EXAMPLE",
                "State": "associated"
            }
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
            "ResourceId": "vpc-7EXAMPLE",
            "ResourceType": "vpc",
            "Subnet": {
                "SubnetId": "subnet-aEXAMPLE",
                "State": "associated"
            }
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
            "ResourceId": "vpc-7EXAMPLE",
            "ResourceType": "vpc",
            "Subnet": {
                "SubnetId": "subnet-fEXAMPLE",
                "State": "associated"
            }
        }
    ]
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Multicast domains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayMulticastDomainAssociations。

`get-transit-gateway-prefix-list-references`

以下代码示例演示了如何使用 `get-transit-gateway-prefix-list-references`。

Amazon CLI

获取中转网关路由表中的前缀列表引用

以下 get-transit-gateway-prefix-list-references 示例将获取指定中转网关路由表的前缀列表引用，并按特定前缀列表的 ID 进行筛选。

```
aws ec2 get-transit-gateway-prefix-list-references \
    --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
    --filters Name=prefix-list-id,Values=pl-11111122222222333
```

输出：

```
{
    "TransitGatewayPrefixListReferences": [
        {
            "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
            "PrefixListId": "pl-11111122222222333",
            "PrefixListOwnerId": "123456789012",
            "State": "available",
            "Blackhole": false,
            "TransitGatewayAttachment": {
                "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",
                "ResourceType": "vpc",
                "ResourceId": "vpc-112233445566aabbc"
            }
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的前缀列表引用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayPrefixListReferences。

get-transit-gateway-route-table-associations

以下代码示例演示了如何使用 get-transit-gateway-route-table-associations。

Amazon CLI

获取有关指定中转网关路由表的关联的信息

以下 get-transit-gateway-route-table-associations 示例显示了有关指定中转网关路由表的关联的信息。

```
aws ec2 get-transit-gateway-route-table-associations \
    --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE
```

输出：

```
{
    "Associations": [
        {
            "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
            "ResourceId": "vpc-4d7de228",
            "ResourceType": "vpc",
            "State": "associating"
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayRouteTableAssociations。

get-transit-gateway-route-table-propagations

以下代码示例演示了如何使用 get-transit-gateway-route-table-propagations。

Amazon CLI

显示有关指定中转网关路由表的路由表传播的信息

以下 get-transit-gateway-route-table-propagations 示例返回指定路由表的路由表传播。

```
aws ec2 get-transit-gateway-route-table-propagations \
    --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE
```

输出：

```
{
    "TransitGatewayRouteTablePropagations": [
        {
```

```
            "TransitGatewayAttachmentId": "tgw-attach-01f8100bc7EXAMPLE",
            "ResourceId": "vpc-3EXAMPLE",
            "ResourceType": "vpc",
            "State": "enabled"
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",
            "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
            "ResourceType": "direct-connect-gateway",
            "State": "enabled"
        },
        {
            "TransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
            "ResourceId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
            "ResourceType": "direct-connect-gateway",
            "State": "enabled"
        }
    ]
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayRouteTablePropagations。

get-verified-access-endpoint-policy

以下代码示例演示了如何使用 get-verified-access-endpoint-policy。

Amazon CLI

获取端点的已验证访问策略

以下 get-verified-access-endpoint-policy 示例获取指定端点的已验证访问策略。

```
aws ec2 get-verified-access-endpoint-policy \
    --verified-access-endpoint-id vae-066fac616d4d546f2
```

输出：

```
{
    "PolicyEnabled": true,
```

```
    "PolicyDocument": "permit(principal,action,resource)\nwhen
 {\n    context.identity.groups.contains(\"finance\") &&\n
 context.identity.email_verified == true\n};"
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVerifiedAccessEndpointPolicy。

get-verified-access-group-policy

以下代码示例演示了如何使用 get-verified-access-group-policy。

Amazon CLI

获取组的已验证访问策略

以下 get-verified-access-group-policy 示例将获取指定组的已验证访问策略。

```
aws ec2 get-verified-access-group-policy \
    --verified-access-group-id vagr-0dbe967baf14b7235
```

输出：

```
{
    "PolicyEnabled": true,
    "PolicyDocument": "permit(principal,action,resource)\nwhen
 {\n    context.identity.groups.contains(\"finance\") &&\n
 context.identity.email_verified == true\n};"
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVerifiedAccessGroupPolicy。

get-vpn-connection-device-sample-configuration

以下代码示例演示了如何使用 get-vpn-connection-device-sample-configuration。

Amazon CLI

下载示例配置文件

以下 get-vpn-connection-device-sample-configuration 示例下载指定的示例配置文件。要列出具有示例配置文件的网关设备，请调用 get-vpn-connection-device-types 命令。

```
aws ec2 get-vpn-connection-device-sample-configuration \
    --vpn-connection-id vpn-123456789abc01234 \
    --vpn-connection-device-type-id 5fb390ba
```

输出：

```
{
    "VpnConnectionDeviceSampleConfiguration": "contents-of-the-sample-configuration-
file"
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的下载配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVpnConnectionDeviceSampleConfiguration。

get-vpn-connection-device-types

以下代码示例演示了如何使用 get-vpn-connection-device-types。

Amazon CLI

列出具有示例配置文件的网关设备

以下 get-vpn-connection-device-types 示例列出了 Palo Alto Networks 中具有配置文件示例的网关设备。

```
aws ec2 get-vpn-connection-device-types \
    --query "VpnConnectionDeviceTypes[?Vendor==`Palo Alto Networks`]"
```

输出：

```
[
    {
        "VpnConnectionDeviceTypeId": "754a6372",
        "Vendor": "Palo Alto Networks",
```

```
        "Platform": "PA Series",
        "Software": "PANOS 4.1.2+"
    },
    {
        "VpnConnectionDeviceTypeId": "9612cbed",
        "Vendor": "Palo Alto Networks",
        "Platform": "PA Series",
        "Software": "PANOS 4.1.2+ (GUI)"
    },
    {
        "VpnConnectionDeviceTypeId": "5fb390ba",
        "Vendor": "Palo Alto Networks",
        "Platform": "PA Series",
        "Software": "PANOS 7.0+"
    }
]
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的下载配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVpnConnectionDeviceTypes。

import-client-vpn-client-certificate-revocation-list

以下代码示例演示了如何使用 import-client-vpn-client-certificate-revocation-list。

Amazon CLI

导入客户端证书吊销列表

以下 import-client-vpn-client-certificate-revocation-list 示例通过指定文件在本地计算机上的位置，从而将客户端证书吊销列表导入到 Client VPN 端点。

```
aws ec2 import-client-vpn-client-certificate-revocation-list \
    --certificate-revocation-list file:///path/to/crl.pem \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

输出：

```
{
    "Return": true
```

```
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的客户端证书吊销列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportClientVpnClientCertificateRevocationList。

`import-image`

以下代码示例演示了如何使用 `import-image`。

Amazon CLI

将 VM 映像文件作为 AMI 导入

以下 `import-image` 示例导入指定的 OVA。

```
aws ec2 import-image \
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/my-server-vm.ova}"
```

输出：

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
    "Progress": "2",
    "SnapshotDetails": [
        {
            "DiskImageSize": 0.0,
            "Format": "ova",
            "UserBucket": {
                "S3Bucket": "my-import-bucket",
                "S3Key": "vms/my-server-vm.ova"
            }
        }
    ],
    "Status": "active",
    "StatusMessage": "pending"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportImage。

import-key-pair

以下代码示例演示了如何使用 import-key-pair。

Amazon CLI

导入公有密钥

首先，使用您选择的工具生成密钥对。例如，使用此 ssh-keygen 命令：

命令:

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

输出：

```
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/my-key.
Your public key has been saved in /home/ec2-user/.ssh/my-key.pub.
...
```

此示例命令导入指定的公有密钥。

命令:

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/my-key.pub
```

输出：

```
{
  "KeyName": "my-key",
  "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportKeyPair。

import-snapshot

以下代码示例演示了如何使用 import-snapshot。

Amazon CLI

导入快照

以下 import-snapshot 示例将指定的磁盘作为快照导入。

```
aws ec2 import-snapshot \
    --description "My server VMDK" \
    --disk-container Format=VMDK,UserBucket={'S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.vmdk'}
```

输出：

```
{
    "Description": "My server VMDK",
    "ImportTaskId": "import-snap-1234567890abcdef0",
    "SnapshotTaskDetail": {
        "Description": "My server VMDK",
        "DiskImageSize": "0.0",
        "Format": "VMDK",
        "Progress": "3",
        "Status": "active",
        "StatusMessage": "pending"
        "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.vmdk"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportSnapshot。

list-images-in-recycle-bin

以下代码示例演示了如何使用 list-images-in-recycle-bin。

Amazon CLI

列出回收站中的映像

以下 list-images-in-recycle-bin 示例列出了当前保留在回收站中的所有映像。

```
aws ec2 list-images-in-recycle-bin
```

输出：

```
{
    "Images": [
        {
            "RecycleBinEnterTime": "2022-03-14T15:35:08.000Z",
            "Description": "Monthly AMI One",
            "RecycleBinExitTime": "2022-03-15T15:35:08.000Z",
            "Name": "AMI_01",
            "ImageId": "ami-0111222333444abcd"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EBS User Guide》中的 Recover deleted AMIs from the Recycle Bin。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImagesInRecycleBin。

list-snapshots-in-recycle-bin

以下代码示例演示了如何使用 list-snapshots-in-recycle-bin。

Amazon CLI

在回收站中查看快照

以下 list-snapshots-in-recycle-bin 示例列出了有关回收站中快照的信息，包括快照 ID、快照的描述、从中创建快照的卷的 ID、删除快照及其进入回收站的日期和时间，以及保留期到期的日期和时间。

```
aws ec2 list-snapshots-in-recycle-bin \
    --snapshot-id snap-01234567890abcdef
```

输出：

```
{
    "SnapshotRecycleBinInfo": [
```

```
        {
            "Description": "Monthly data backup snapshot",
            "RecycleBinEnterTime": "2022-12-01T13:00:00.000Z",
            "RecycleBinExitTime": "2022-12-15T13:00:00.000Z",
            "VolumeId": "vol-abcdef09876543210",
            "SnapshotId": "snap-01234567890abcdef"
        }
    ]
}
```

有关回收站的更多信息，请参阅《Amazon EBS User Guide》中的 Recover deleted snapshots from the Recycle Bin。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSnapshotsInRecycleBin。

`lock-snapshot`

以下代码示例演示了如何使用 `lock-snapshot`。

Amazon CLI

示例 1：在监管模式下锁定快照

以下 `lock-snapshot` 示例在监管模式下锁定指定的快照。

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0b5e733b4a8df6e0d \
    --lock-mode governance \
    --lock-duration 365
```

输出：

```
{
    "SnapshotId": "snap-0b5e733b4a8df6e0d",
    "LockState": "governance",
    "LockDuration": 365,
    "LockCreatedOn": "2024-05-05T00:56:06.208000+00:00",
    "LockExpiresOn": "2025-05-05T00:56:06.208000+00:00",
    "LockDurationStartTime": "2024-05-05T00:56:06.208000+00:00"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的快照锁定。

示例 2：在合规模式下锁定快照

以下 lock-snapshot 示例在合规模式下锁定指定的快照。

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0163a8524c5b9901f \
    --lock-mode compliance \
    --cool-off-period 24 \
    --lock-duration 365
```

输出：

```
{
    "SnapshotId": "snap-0b5e733b4a8df6e0d",
    "LockState": "compliance-cooloff",
    "LockDuration": 365,
    "CoolOffPeriod": 24,
    "CoolOffPeriodExpiresOn": "2024-05-06T01:02:20.527000+00:00",
    "LockCreatedOn": "2024-05-05T01:02:20.527000+00:00",
    "LockExpiresOn": "2025-05-05T01:02:20.527000+00:00",
    "LockDurationStartTime": "2024-05-05T01:02:20.527000+00:00"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的快照锁定。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 LockSnapshot。

modify-address-attribute

以下代码示例演示了如何使用 modify-address-attribute。

Amazon CLI

修改与弹性 IP 地址关联的域名属性

以下 modify-address-attribute 示例修改弹性 IP 地址的域名属性。

Linux：

```
aws ec2 modify-address-attribute \
    --allocation-id eipalloc-abcdef01234567890 \
    --domain-name example.com
```

Windows:

```
aws ec2 modify-address-attribute ^
    --allocation-id eipalloc-abcdef01234567890 ^
    --domain-name example.com
```

输出：

```
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.net."
            "PtrRecordUpdate": {
                "Value": "example.com.",
                "Status": "PENDING"
        }
    ]
}
```

要监控待定更改并查看弹性 IP 地址的修改属性，请参阅《Amazon CLI 命令参考》中的 describe-addresses-attribute。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyAddressAttribute。

modify-availability-zone-group

以下代码示例演示了如何使用 modify-availability-zone-group。

Amazon CLI

启用区组

以下 modify-availability-zone-group 示例启用指定的区组。

```
aws ec2 modify-availability-zone-group \
    --group-name us-west-2-lax-1 \
    --opt-in-status opted-in
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的区域和区。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyAvailabilityZoneGroup。

modify-capacity-reservation-fleet

以下代码示例演示了如何使用 modify-capacity-reservation-fleet。

Amazon CLI

示例 1：修改容量预留实例集的总目标容量

以下 modify-capacity-reservation-fleet 示例修改指定容量预留实例集的总目标容量。当您修改容量预留机群的总目标容量时，机群会自动创建新的容量预留，或者修改或取消机群中的现有容量预留以满足新的总目标容量。当机群处于 modifying 状态时，您无法尝试对其进行其他修改。

```
aws ec2 modify-capacity-reservation-fleet \
    --capacity-reservation-fleet-id crf-01234567890abcedf \
    --total-target-capacity 160
```

输出：

```
{
    "Return": true
}
```

示例 2：修改容量预留实例集的结束日期

以下 modify-capacity-reservation-fleet 示例修改指定容量预留实例集的结束日期。当您修改机群的结束日期时，所有单个容量预留的结束日期都会相应更新。当机群处于 modifying 状态时，您无法尝试对其进行其他修改。

```
aws ec2 modify-capacity-reservation-fleet \
    --capacity-reservation-fleet-id crf-01234567890abcedf \
    --end-date 2022-07-04T23:59:59.000Z
```

输出：

```
{
    "Return": true
}
```

有关容量预留实例集的更多信息，请参阅《Amazon EC2 用户指南》中的容量预留实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCapacityReservationFleet。

modify-capacity-reservation

以下代码示例演示了如何使用 modify-capacity-reservation。

Amazon CLI

示例 1：更改现有容量预留所预留的实例数量

以下 modify-capacity-reservation 示例将更改容量预留为其预留容量的实例数量。

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --instance-count 5
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改容量预留。

示例 2：更改现有容量预留的结束日期和时间

以下 modify-capacity-reservation 示例将现有容量预留修改为在指定的日期和时间结束。

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改容量预留。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCapacityReservation。

modify-client-vpn-endpoint

以下代码示例演示了如何使用 modify-client-vpn-endpoint。

Amazon CLI

修改 Client VPN 端点

以下 modify-client-vpn-endpoint 示例为指定的 Client VPN 端点启用客户端连接日志记录。

```
aws ec2 modify-client-vpn-endpoint \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --connection-log-options Enabled=true,CloudwatchLogGroup=ClientVPNLogs
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的 Client VPN 端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClientVpnEndpoint。

modify-default-credit-specification

以下代码示例演示了如何使用 modify-default-credit-specification。

Amazon CLI

修改默认积分选项

以下 modify-default-credit-specification 示例修改 T2 实例的默认积分选项。

```
aws ec2 modify-default-credit-specification \
    --instance-family t2 \
```

```
    --cpu-credits unlimited
```

输出：

```
{
    "InstanceFamilyCreditSpecification": {
        "InstanceFamily": "t2",
        "CpuCredits": "unlimited"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDefaultCreditSpecification。

modify-ebs-default-kms-key-id

以下代码示例演示了如何使用 modify-ebs-default-kms-key-id。

Amazon CLI

为 EBS 加密设置默认 CMK

以下 modify-ebs-default-kms-key-id 示例将指定的 CMK 设置为当前区域中 Amazon 账户的 EBS 加密的默认 CMK。

```
aws ec2 modify-ebs-default-kms-key-id \
    --kms-key-id alias/my-cmk
```

输出：

```
{
    "KmsKeyId": "arn:aws:kms:us-
west-2:123456789012:key/0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyEbsDefaultKmsKeyId。

modify-fleet

以下代码示例演示了如何使用 modify-fleet。

Amazon CLI

扩展 EC2 实例集

以下 modify-fleet 示例修改指定 EC2 实例集的目标容量。如果指定值大于当前容量，EC2 实例集将启动额外的实例。如果指定值小于当前容量，则 EC2 实例集将取消任何打开的请求，而如果终止策略为 terminate，则 EC2 实例集将终止超过新目标容量的任何实例。

```
aws ec2 modify-fleet \
    --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
    --target-capacity-specification TotalTargetCapacity=5
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的管理 EC2 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyFleet。

modify-fpga-image-attribute

以下代码示例演示了如何使用 modify-fpga-image-attribute。

Amazon CLI

修改 Amazon FPGA 映像的属性

此示例为指定 AFI 的账户 ID 123456789012 添加加载权限。

命令:

```
aws ec2 modify-fpga-image-attribute --attribute loadPermission --fpga-image-
id afi-0d123e123bfc85abc --load-permission Add=[{UserId=123456789012}]
```

输出：

```
{
  "FpgaImageAttribute": {
```

```
      "FpgaImageId": "afi-0d123e123bfc85abc",
      "LoadPermissions": [
          {
              "UserId": "123456789012"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyFpgaImageAttribute。

modify-hosts

以下代码示例演示了如何使用 modify-hosts。

Amazon CLI

示例 1：为专属主机启用自动置放

以下 modify-hosts 示例为专属主机启用自动置放，以便它接受与其实例类型配置相匹配的任何非定向实例启动。

```
aws ec2 modify-hosts \
    --host-id h-06c2f189b4EXAMPLE \
    --auto-placement on
```

输出：

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改专属主机的自动置放设置。

示例 2：为专属主机启用主机恢复

以下 modify-hosts 示例为指定的专属主机启用主机恢复。

```
aws ec2 modify-hosts \
```

```
    --host-id h-06c2f189b4EXAMPLE \
    --host-recovery on
```

输出：

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改专属主机的自动置放设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyHosts。

modify-id-format

以下代码示例演示了如何使用 modify-id-format。

Amazon CLI

为资源启用较长 ID 格式

以下 modify-id-format 示例为 instance 资源类型启用较长 ID 格式。

```
aws ec2 modify-id-format \
    --resource instance \
    --use-long-ids
```

为资源禁用较长 ID 格式

以下 modify-id-format 示例为 instance 资源类型禁用较长 ID 格式。

```
aws ec2 modify-id-format \
    --resource instance \
    --no-use-long-ids
```

以下 modify-id-format 示例为处于选择周期内的所有支持的资源类型启用较长 ID 格式。

```
aws ec2 modify-id-format \
```

```
    --resource all-current \
    --use-long-ids
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIdFormat。

modify-identity-id-format

以下代码示例演示了如何使用 modify-identity-id-format。

Amazon CLI

使 IAM 角色能够对资源使用较长 ID

以下 modify-identity-id-format 示例使您的 Amazon 账户中的 IAM 角色 EC2Role 能够对 instance 资源类型使用长 ID 格式。

```
aws ec2 modify-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:role/EC2Role \
    --resource instance \
    --use-long-ids
```

使 IAM 用户能够对资源使用较长 ID

以下 modify-identity-id-format 示例使 Amazon 账户中的 IAM 用户 AdminUser 能够对 volume 资源类型使用较长 ID 格式。

```
aws ec2 modify-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:user/AdminUser \
    --resource volume \
    --use-long-ids
```

以下 modify-identity-id-format 示例使您的 Amazon 账户中的 IAM 用户 AdminUser 能够对处于选择周期内的所有支持的资源类型使用较长 ID 格式。

```
aws ec2 modify-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:user/AdminUser \
    --resource all-current \
    --use-long-ids
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIdentityIdFormat。

modify-image-attribute

以下代码示例演示了如何使用 modify-image-attribute。

Amazon CLI

示例 1：公开 AMI

以下 modify-instance-attribute 示例将公开指定的 AMI。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Add=[{Group=all}]"
```

此命令不生成任何输出。

示例 2：将 AMI 设为私有

以下 modify-instance-attribute 示例将指定的 AMI 设为私有。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Remove=[{Group=all}]"
```

此命令不生成任何输出。

示例 3：向 Amazon 账户授予启动权限

以下 modify-instance-attribute 示例向指定的 Amazon 账户授予启动权限。

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Add=[{UserId=123456789012}]"
```

此命令不生成任何输出。

示例 4：从 Amazon 账户中删除启动权限

以下 modify-instance-attribute 示例从指定的 Amazon 账户中删除启动权限。

```
aws ec2 modify-image-attribute \
```

```
    --image-id ami-5731123e \
    --launch-permission "Remove=[{UserId=123456789012}]"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyImageAttribute。

modify-instance-attribute

以下代码示例演示了如何使用 modify-instance-attribute。

Amazon CLI

示例 1：修改实例类型

以下 modify-instance-attribute 示例修改指定实例的实例类型。该实例必须处于 stopped 状态。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --instance-type "{\"Value\": \"m1.small\"}"
```

此命令不生成任何输出。

示例 2：在实例上启用增强联网

以下 modify-instance-attribute 示例为指定实例启用增强联网。该实例必须处于 stopped 状态。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --sriov-net-support simple
```

此命令不生成任何输出。

示例 3：修改 sourceDestCheck 属性

以下 modify-instance-attribute 示例将指定实例的 sourceDestCheck 属性设置为 true。该实例必须在 VPC 中。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-dest-
check "{\"Value\": true}"
```

此命令不生成任何输出。

示例 4：修改根卷的 deleteOnTermination 属性

以下 modify-instance-attribute 示例将指定的 Amazon EBS 支持的实例的根卷的 deleteOnTermination 属性设置为 false。默认情况下，根卷的此属性为 true。

命令:

```
aws ec2 modify-instance-attribute \
  --instance-id i-1234567890abcdef0 \
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\",\"Ebs\":
{\"DeleteOnTermination\":false}}]"
```

此命令不生成任何输出。

示例 5：修改附加到实例的用户数据

以下 modify-instance-attribute 示例将文件 UserData.txt 的内容添加为指定实例的 UserData。

原始文件 UserData.txt 的内容：

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

该文件的内容必须采用 base64 编码。第一个命令将文本文件转换为 base64 并将其另存为新文件。

此命令的 Linux/macOS 版本：

```
base64 UserData.txt > UserData.base64.txt
```

此命令不生成任何输出。

此命令的 Windows 版本：

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
 UserData.base64.txt
```

输出：

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

现在，您可以在以下 CLI 命令中引用该文件：

```
aws ec2 modify-instance-attribute \
    --instance-id=i-09b5a14dbca622e76 \
    --attribute userData --value file://UserData.base64.txt
```

此命令不生成任何输出。

有关更多信息，请参阅《EC2 用户指南》中的用户数据和 Amazon CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceAttribute。

modify-instance-capacity-reservation-attributes

以下代码示例演示了如何使用 modify-instance-capacity-reservation-attributes。

Amazon CLI

示例 1：修改实例的容量预留目标设置

以下 modify-instance-capacity-reservation-attributes 示例修改已停止的实例以针对特定容量预留。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-EXAMPLE8765abcd4e \
    --capacity-reservation-specification
 'CapacityReservationTarget={CapacityReservationId= cr-1234abcd56EXAMPLE }'
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改您的实例的容量预留设置。

示例 2：修改实例的容量预留目标设置

以下 modify-instance-capacity-reservation-attributes 示例修改一个已停止的实例，该实例以指定的容量预留为目标，在具有匹配属性（实例类型、平台、可用区）和开放实例匹配条件的任何容量预留中启动。

```
aws ec2 modify-instance-capacity-reservation-attributes \
    --instance-id i-EXAMPLE8765abcd4e \
    --capacity-reservation-specification 'CapacityReservationPreference=open'
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改您的实例的容量预留设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceCapacityReservationAttributes。

modify-instance-credit-specification

以下代码示例演示了如何使用 modify-instance-credit-specification。

Amazon CLI

修改实例的 CPU 使用率积分选项

此示例将指定区域中指定实例的 CPU 使用率的积分选项修改为“无限制”。有效的积分选项为“标准”和“无限制”。

命令:

```
aws ec2 modify-instance-credit-specification --instance-credit-
specification "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

输出：

```
{
```

```
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceCreditSpecification。

`modify-instance-event-start-time`

以下代码示例演示了如何使用 `modify-instance-event-start-time`。

Amazon CLI

修改实例的事件开始时间

以下 `modify-instance-event-start-time` 命令显示如何修改指定实例的事件开始时间。通过使用 `--instance-event-id` 参数来指定事件 ID。通过使用 `--not-before` 参数来指定新的日期和时间。

```
aws ec2 modify-instance-event-start-time --instance-id i-1234567890abcdef0
 --instance-event-id instance-event-0abcdef1234567890 --not-
before 2019-03-25T10:00:00.000
```

输出：

```
"Event": {
    "InstanceEventId": "instance-event-0abcdef1234567890",
    "Code": "system-reboot",
    "Description": "scheduled reboot",
    "NotAfter": "2019-03-25T12:00:00.000Z",
    "NotBefore": "2019-03-25T10:00:00.000Z",
    "NotBeforeDeadline": "2019-04-22T21:00:00.000Z"
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“使用计划为重启的实例”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceEventStartTime。

modify-instance-event-window

以下代码示例演示了如何使用 modify-instance-event-window。

Amazon CLI

示例 1：修改事件窗口的时间范围

以下 modify-instance-event-window 示例修改事件窗口的时间范围。指定 time-range 参数以修改时间范围。您不能同时指定 cron-expression 参数。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
```

```
            }
        ]
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 2：修改事件窗口的一组时间范围

以下 modify-instance-event-window 示例修改事件窗口的时间范围。指定 time-range 参数以修改时间范围。您不能同时指定 cron-expression 参数。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --time-range '[{"StartWeekDay": "monday", "StartHour": 2, "EndWeekDay":
 "wednesday", "EndHour": 8},
        {"StartWeekDay": "thursday", "StartHour": 2, "EndWeekDay": "friday",
 "EndHour": 8}]'
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            },
            {
                "StartWeekDay": "thursday",
                "StartHour": 2,
                "EndWeekDay": "friday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
```

```
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

示例 3：修改事件窗口的 cron 表达式

以下 `modify-instance-event-window` 示例修改事件窗口的 cron 表达式。指定 `cron-expression` 参数以修改 Cron 表达式。您不能同时指定 `time-range` 参数。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --cron-expression "* 21-23 * * 2,3"
```

输出：

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
```

```
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

有关事件窗口约束的信息，请参阅《Amazon EC2 用户指南》的“计划的事件”部分的注意事项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceEventWindow。

modify-instance-maintenance-options

以下代码示例演示了如何使用 modify-instance-maintenance-options。

Amazon CLI

示例 1：禁用实例的恢复行为

以下 modify-instance-maintenance-options 示例禁用正在运行或已停止实例的简化自动恢复。

```
aws ec2 modify-instance-maintenance-options \
    --instance-id i-0abcdef1234567890 \
    --auto-recovery disabled
```

输出：

```
{
    "InstanceId": "i-0abcdef1234567890",
    "AutoRecovery": "disabled"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的配置简化的自动恢复。

示例 2：将实例的恢复行为设置为默认值

以下 modify-instance-maintenance-options 示例将自动恢复行为设置为默认值，这样可以简化支持的实例类型的自动恢复。

```
aws ec2 modify-instance-maintenance-options \
    --instance-id i-0abcdef1234567890 \
    --auto-recovery default
```

输出：

```
{
    "InstanceId": "i-0abcdef1234567890",
    "AutoRecovery": "default"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的配置简化的自动恢复。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceMaintenanceOptions。

modify-instance-metadata-options

以下代码示例演示了如何使用 modify-instance-metadata-options。

Amazon CLI

示例 1：启用 IMDSv2

以下 modify-instance-metadata-options 示例配置在指定实例上使用 IMDSv2。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-tokens required \
    --http-endpoint enabled
```

输出：

```
{
    "InstanceId": "i-1234567898abcdef0",
    "InstanceMetadataOptions": {
        "State": "pending",
        "HttpTokens": "required",
        "HttpPutResponseHopLimit": 1,
        "HttpEndpoint": "enabled"
```

```
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据。

示例 2：禁用实例元数据

以下 modify-instance-metadata-options 示例禁止在指定实例上使用所有版本的实例元数据。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-endpoint disabled
```

输出：

```
{
    "InstanceId": "i-1234567898abcdef0",
    "InstanceMetadataOptions": {
        "State": "pending",
        "HttpTokens": "required",
        "HttpPutResponseHopLimit": 1,
        "HttpEndpoint": "disabled"
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据。

示例 3：为您的实例启用实例元数据 IPv6 端点

以下 modify-instance-metadata-options 示例显示如何打开实例元数据服务的 IPv6 端点。默认禁用 IPv6 端点。即使您已在仅 IPv6 子网中启动了实例，也是如此。IMDS 的 IPv6 端点只能在基于 Nitro System 构建的实例上访问。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-protocol-ipv6 enabled \
    --http-endpoint enabled
```

输出：

```
{
    "InstanceId": "i-1234567898abcdef0",
    "InstanceMetadataOptions": {
        "State": "pending",
        "HttpTokens": "required",
        "HttpPutResponseHopLimit": 1,
        "HttpEndpoint": "enabled",
        HttpProtocolIpv6": "enabled"
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceMetadataOptions。

modify-instance-placement

以下代码示例演示了如何使用 modify-instance-placement。

Amazon CLI

示例 1：删除实例与专属主机的关联

以下 modify-instance-placement 示例删除实例与专属主机的关联，并使其能够在您的账户中支持其实例类型的任何可用专属主机上启动。

```
aws ec2 modify-instance-placement \
    --instance-id i-0e6ddf6187EXAMPLE \
    --affinity default
```

输出：

```
{
    "Return": true
}
```

示例 2：在实例和指定的专属主机之间建立关联

以下 modify-instance-placement 示例在实例和专属主机之间建立启动关系。该实例只能在指定专属主机上运行。

```
aws ec2 modify-instance-placement \
    --instance-id i-0e6ddf6187EXAMPLE \
    --affinity host \
    --host-id i-0e6ddf6187EXAMPLE
```

输出：

```
{
    "Return": true
}
```

示例 3：将实例移动到置放群组

以下 modify-instance-placement 示例将实例移至置放群组，停止该实例，修改实例置放，然后重新启动该实例。

```
aws ec2 stop-instances \
    --instance-ids i-0123a456700123456

aws ec2 modify-instance-placement \
    --instance-id i-0123a456700123456 \
    --group-name MySpreadGroup

aws ec2 start-instances \
    --instance-ids i-0123a456700123456
```

示例 4：从置放群组中删除实例

以下 modify-instance-placement 示例通过停止实例、修改实例置放，然后重新启动实例，从而从置放群组中删除实例。以下示例为置放群组名称指定一个空字符串 ("")，以指示实例不应位于置放群组中。

停止实例：

```
aws ec2 stop-instances \
    --instance-ids i-0123a456700123456
```

修改置放（Windows 命令提示符）：

```
aws ec2 modify-instance-placement \
```

```
    --instance-id i-0123a456700123456 \
    --group-name ""
```

修改置放（Windows PowerShell、Linux 和 macOS）：

```
aws ec2 modify-instance-placement `
    --instance-id i-0123a456700123456 `
    --group-name ''
```

重新启动实例：

```
aws ec2 start-instances \
    --instance-ids i-0123a456700123456
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的修改专属主机租赁和关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstancePlacement。

modify-ipam-pool

以下代码示例演示了如何使用 modify-ipam-pool。

Amazon CLI

修改 IPAM 池

以下 modify-ipam-pool 示例修改 IPAM 池。

（Linux）：

```
aws ec2 modify-ipam-pool \
    --ipam-pool-id ipam-pool-0533048da7d823723 \
    --add-allocation-resource-tags "Key=Owner,Value=Build Team" \
    --clear-allocation-default-netmask-length \
```

```
    --allocation-min-netmask-length 14
```

（Windows）：

```
aws ec2 modify-ipam-pool ^
    --ipam-pool-id ipam-pool-0533048da7d823723 ^
    --add-allocation-resource-tags "Key=Owner,Value=Build Team" ^
    --clear-allocation-default-netmask-length ^
    --allocation-min-netmask-length 14
```

输出：

```
{
    "IpamPool": {
        "OwnerId": "123456789012",
        "IpamPoolId": "ipam-pool-0533048da7d823723",
        "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-0533048da7d823723",
        "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
        "IpamScopeType": "private",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
        "IpamRegion": "us-east-1",
        "Locale": "None",
        "PoolDepth": 1,
        "State": "modify-complete",
        "AutoImport": true,
        "AddressFamily": "ipv4",
        "AllocationMinNetmaskLength": 14,
        "AllocationMaxNetmaskLength": 26,
        "AllocationResourceTags": [
            {
                "Key": "Environment",
                "Value": "Preprod"
            },
            {
                "Key": "Owner",
                "Value": "Build Team"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的编辑池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIpamPool。

modify-ipam-resource-cidr

以下代码示例演示了如何使用 modify-ipam-resource-cidr。

Amazon CLI

修改分配给资源的 CIDR

以下 modify-ipam-resource-cidr 示例修改资源 CIDR。

（Linux）：

```
aws ec2 modify-ipam-resource-cidr \
    --current-ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \
    --destination-ipam-scope-id ipam-scope-0da34c61fd189a141 \
    --resource-id vpc-010e1791024eb0af9 \
    --resource-cidr 10.0.1.0/24 \
    --resource-region us-east-1 \
    --monitored
```

（Windows）：

```
aws ec2 modify-ipam-resource-cidr ^
    --current-ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^
    --destination-ipam-scope-id ipam-scope-0da34c61fd189a141 ^
    --resource-id vpc-010e1791024eb0af9 ^
    --resource-cidr 10.0.1.0/24 ^
    --resource-region us-east-1 ^
    --monitored
```

输出：

```
{
    "IpamResourceCidr": {
        "IpamId": "ipam-08440e7a3acde3908",
        "IpamScopeId": "ipam-scope-0da34c61fd189a141",
        "IpamPoolId": "ipam-pool-0533048da7d823723",
        "ResourceRegion": "us-east-1",
        "ResourceOwnerId": "123456789012",
```

```
        "ResourceId": "vpc-010e1791024eb0af9",
        "ResourceCidr": "10.0.1.0/24",
        "ResourceType": "vpc",
        "ResourceTags": [
            {
                "Key": "Environment",
                "Value": "Preprod"
            },
            {
                "Key": "Owner",
                "Value": "Build Team"
            }
        ],
        "IpUsage": 0.0,
        "ComplianceStatus": "noncompliant",
        "ManagementState": "managed",
        "OverlapStatus": "overlapping",
        "VpcId": "vpc-010e1791024eb0af9"
    }
}
```

有关移动资源的更多信息，请参阅《Amazon VPC IPAM 用户指南》中的在范围之间移动资源 CIDR。

有关更改监控状态的更多信息，请参阅《Amazon VPC IPAM 用户指南》中的更改资源 CIDR 的监控状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIpamResourceCidr。

`modify-ipam-resource-discovery`

以下代码示例演示了如何使用 `modify-ipam-resource-discovery`。

Amazon CLI

修改资源发现的运营区域

在此示例中，您是 IPAM 委派管理员，想要修改资源发现的运营区域。

要完成此请求，请执行以下操作：

您无法修改默认资源发现，并且您必须是资源发现的所有者。您需要资源发现 ID，您可以通过 describe-ipam-resource-discoveries 获取该 ID。

以下 modify-ipam-resource-discovery 示例修改 Amazon 账户中的非默认资源发现。

```
aws ec2 modify-ipam-resource-discovery \
    --ipam-resource-discovery-id ipam-res-disco-0f4ef577a9f37a162 \
    --add-operating-regions RegionName='us-west-1' \
    --remove-operating-regions RegionName='us-east-2' \
    --region us-east-1
```

输出：

```
{
    "IpamResourceDiscovery": {
        "OwnerId": "149977607591",
        "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
        "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0365d2977fc1672fe",
        "IpamResourceDiscoveryRegion": "us-east-1",
        "Description": "Example",
        "OperatingRegions": [
            {
                "RegionName": "us-east-1"
            },
            {
                "RegionName": "us-west-1"
            }
        ],
        "IsDefault": false,
        "State": "modify-in-progress"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的使用资源发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIpamResourceDiscovery。

modify-ipam-scope

以下代码示例演示了如何使用 modify-ipam-scope。

Amazon CLI

修改范围描述

在此场景中，您是 IPAM 委派管理员，想要修改 IPAM 范围的描述。

要完成此请求，您需要范围 ID，您可以通过 describe-ipam-scopes 获取该 ID。

以下 modify-ipam-scope 示例将更新范围的描述。

```
aws ec2 modify-ipam-scope \
    --ipam-scope-id ipam-scope-0d3539a30b57dcdd1 \
    --description example \
    --region us-east-1
```

输出：

```
{
"IpamScope": {
        "OwnerId": "320805250157",
        "IpamScopeId": "ipam-scope-0d3539a30b57dcdd1",
        "IpamScopeArn": "arn:aws:ec2::320805250157:ipam-scope/ipam-
scope-0d3539a30b57dcdd1",
        "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
        "IpamRegion": "us-east-1",
        "IpamScopeType": "public",
        "IsDefault": true,
        "Description": "example",
        "PoolCount": 1,
        "State": "modify-in-progress"
    }
}
```

有关范围的更多信息，请参阅《Amazon VPC IPAM 用户指南》中的 IPAM 的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIpamScope。

modify-ipam

以下代码示例演示了如何使用 modify-ipam。

Amazon CLI

修改 IPAM

以下 modify-ipam 示例通过添加运营区域来修改 IPAM。

（Linux）：

```
aws ec2 modify-ipam \
    --ipam-id ipam-08440e7a3acde3908 \
    --add-operating-regions RegionName=us-west-2
```

（Windows）：

```
aws ec2 modify-ipam ^
    --ipam-id ipam-08440e7a3acde3908 ^
    --add-operating-regions RegionName=us-west-2
```

输出：

```
{
    "Ipam": {
        "OwnerId": "123456789012",
        "IpamId": "ipam-08440e7a3acde3908",
        "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
        "IpamRegion": "us-east-1",
        "PublicDefaultScopeId": "ipam-scope-0b9eed026396dbc16",
        "PrivateDefaultScopeId": "ipam-scope-02fc38cd4c48e7d38",
        "ScopeCount": 3,
        "OperatingRegions": [
            {
                "RegionName": "us-east-1"
            },
            {
                "RegionName": "us-east-2"
            },
            {
                "RegionName": "us-west-1"
            },
            {
                "RegionName": "us-west-2"
            }
        ],
        "State": "modify-in-progress"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyIpam。

modify-launch-template

以下代码示例演示了如何使用 modify-launch-template。

Amazon CLI

更改默认启动模板版本

此示例会将指定启动模板的版本 2 指定为默认版本。

命令:

```
aws ec2 modify-launch-template --launch-template-id lt-0abcd290751193123 --default-version 2
```

输出：

```
{
  "LaunchTemplate": {
      "LatestVersionNumber": 2,
      "LaunchTemplateId": "lt-0abcd290751193123",
      "LaunchTemplateName": "WebServers",
      "DefaultVersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-12-01T13:35:46.000Z"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyLaunchTemplate。

modify-managed-prefix-list

以下代码示例演示了如何使用 modify-managed-prefix-list。

Amazon CLI

修改前缀列表

以下 modify-managed-prefix-list 示例将一个条目添加到指定的前缀列表中。

```
aws ec2 modify-managed-prefix-list \
```

```
    --prefix-list-id pl-0123456abcabcabc1 \
    --add-entries Cidr=10.1.0.0/16,Description=vpc-c \
    --current-version 1
```

输出：

```
{
    "PrefixList": {
        "PrefixListId": "pl-0123456abcabcabc1",
        "AddressFamily": "IPv4",
        "State": "modify-in-progress",
        "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
        "PrefixListName": "vpc-cidrs",
        "MaxEntries": 10,
        "Version": 1,
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyManagedPrefixList。

modify-network-interface-attribute

以下代码示例演示了如何使用 modify-network-interface-attribute。

Amazon CLI

修改网络接口的连接属性

此示例命令修改指定网络接口的 attachment 属性。

命令：

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

修改网络接口的描述属性

此示例命令修改指定网络接口的 description 属性。

命令:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
description "My description"
```

修改网络接口的 groupSet 属性

此示例命令修改指定网络接口的 groupSet 属性。

命令:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
groups sg-903004f8 sg-1a2b3c4d
```

修改网络接口的 sourceDestCheck 属性

此示例命令修改指定网络接口的 sourceDestCheck 属性。

命令:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --no-
source-dest-check
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyNetworkInterfaceAttribute。

modify-private-dns-name-options

以下代码示例演示了如何使用 modify-private-dns-name-options。

Amazon CLI

修改实例主机名的选项

以下 modify-private-dns-name-options 示例会禁用响应具有 DNS A 记录的实例主机名的 DNS 查询的选项。

```
aws ec2 modify-private-dns-name-options \
    --instance-id i-1234567890abcdef0 \
    --no-enable-resource-name-dns-a-record
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 实例主机名类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyPrivateDnsNameOptions。

`modify-reserved-instances`

以下代码示例演示了如何使用 `modify-reserved-instances`。

Amazon CLI

修改预留实例

此示例命令将预留实例移动到同一区域的另一个可用区。

命令:

```
aws ec2 modify-reserved-instances --reserved-instances-ids b847fa93-e282-4f55-b59a-1342f5bd7c02 --target-configurations AvailabilityZone=us-west-1c,Platform=EC2-Classic,InstanceCount=10
```

输出：

```
{
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"
}
```

修改预留实例的网络平台

此示例命令将 EC2-Classic 预留实例转换为 EC2-VPC。

命令:

```
aws ec2 modify-reserved-instances --reserved-instances-ids f127bd27-edb7-44c9-a0eb-0d7e09259af0 --target-configurations AvailabilityZone=us-west-1c,Platform=EC2-VPC,InstanceCount=5
```

输出：

```
{
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的“修改预留实例”。

修改预留实例的实例大小

此示例命令修改在 us-west-1c 中具有 10 个 m1.small Linux/UNIX 实例的预留实例，以便在同一可用区中将 8 个 m1.small 实例变为 2 个 m1.large 实例，其余 2 个 m1.small 实例将变为 1 个 m1.medium 实例。命令:

```
aws ec2 modify-reserved-instances --reserved-instances-
ids 1ba8e2e3-3556-4264-949e-63ee671405a9 --target-
configurations AvailabilityZone=us-west-1c,Platform=EC2-
Classic,InstanceCount=2,InstanceType=m1.large AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=1,InstanceType=m1.medium
```

输出：

```
{
    "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-b3e3-1c6b11fa00b6"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的“修改预留的实例大小”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReservedInstances。

modify-security-group-rules

以下代码示例演示了如何使用 modify-security-group-rules。

Amazon CLI

修改安全组规则以更新规则描述、IP 协议和 CidrIpv4 地址范围

以下 modify-security-group-rules 示例更新指定安全组规则的描述、IP 协议和 IPV4 CIDR 范围。使用 security-group-rules 参数输入指定安全组规则的更新。-1 指定所有协议。

```
aws ec2 modify-security-group-rules \
    --group-id sg-1234567890abcdef0 \
    --security-group-rules SecurityGroupRuleId=sgr-
abcdef01234567890,SecurityGroupRule='{Description=test,IpProtocol=-1,CidrIpv4=0.0.0.0/0}'
```

输出：

```
{
    "Return": true
}
```

有关安全组规则的更多信息，请参阅《Amazon EC2 用户指南》中的安全组规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySecurityGroupRules。

modify-snapshot-attribute

以下代码示例演示了如何使用 modify-snapshot-attribute。

Amazon CLI

示例 1：修改快照属性

以下 modify-snapshot-attribute 示例更新指定快照的 createVolumePermission 属性，删除了指定用户的卷权限。

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type remove \
    --user-ids 123456789012
```

示例 2：公开快照

以下 modify-snapshot-attribute 示例将公开指定的快照。

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type add \
    --group-names all
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySnapshotAttribute。

modify-snapshot-tier

以下代码示例演示了如何使用 modify-snapshot-tier。

Amazon CLI

归档快照

以下 modify-snapshot-tier 示例归档指定的快照。响应参数 TieringStartTime 以 UTC 时间格式（YYYY-MM-DDTHH:MM:SSZ）表示归档过程的启动日期和时间。

```
aws ec2 modify-snapshot-tier \
    --snapshot-id snap-01234567890abcedf \
    --storage-tier archive
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "TieringStartTime": "2021-09-15T16:44:37.574Z"
}
```

有关快照归档的更多信息，请参阅《Amazon EBS User Guide》中的 Archive Amazon EBS snapshots。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySnapshotTier。

modify-spot-fleet-request

以下代码示例演示了如何使用 modify-spot-fleet-request。

Amazon CLI

修改竞价型实例集请求

此示例命令更新指定竞价型实例集请求的目标容量。

命令:

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-
id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

输出：

```
{
    "Return": true
}
```

此示例命令会减少指定竞价型实例集请求的目标容量，而不会因此终止任何竞价型实例。

命令:

```
aws ec2 modify-spot-fleet-request --target-capacity 10 --excess-capacity-
termination-policy NoTermination --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

输出：

```
{
    "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySpotFleetRequest。

modify-subnet-attribute

以下代码示例演示了如何使用 modify-subnet-attribute。

Amazon CLI

更改子网的公有 IPv4 寻址行为

此示例修改 subnet-1a2b3c4d，以指定在该子网中启动的所有实例都被分配一个公有 IPv4 地址。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-
launch
```

更改子网的 IPv6 寻址行为

此示例修改 subnet-1a2b3c4d，以指定在该子网中启动的所有实例都被分配子网范围内的一个 IPv6 地址。

命令:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-address-on-creation
```

有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的“VPC 中的 IP 寻址”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySubnetAttribute。

modify-traffic-mirror-filter-network-services

以下代码示例演示了如何使用 modify-traffic-mirror-filter-network-services。

Amazon CLI

将网络服务添加到流量镜像筛选条件

以下 modify-traffic-mirror-filter-network-services 示例将 Amazon DNS 网络服务添加到指定的筛选条件。

```
aws ec2 modify-traffic-mirror-filter-network-services \
    --traffic-mirror-filter-id tmf-04812ff784EXAMPLE \
    --add-network-service amazon-dns
```

输出：

```
{
    "TrafficMirrorFilter": {
        "Tags": [
            {
                "Key": "Name",
                "Value": "Production"
            }
        ],
        "EgressFilterRules": [],
        "NetworkServices": [
```

```
            "amazon-dns"
        ],
        "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
        "IngressFilterRules": [
            {
                "SourceCidrBlock": "0.0.0.0/0",
                "RuleNumber": 1,
                "DestinationCidrBlock": "0.0.0.0/0",
                "Description": "TCP Rule",
                "Protocol": 6,
                "TrafficDirection": "ingress",
                "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
                "RuleAction": "accept",
                "TrafficMirrorFilterRuleId": "tmf-04812ff784EXAMPLE"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的修改流量镜像筛选条件网络服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTrafficMirrorFilterNetworkServices。

modify-traffic-mirror-filter-rule

以下代码示例演示了如何使用 modify-traffic-mirror-filter-rule。

Amazon CLI

修改流量镜像筛选条件规则

以下 modify-traffic-mirror-filter-rule 示例修改指定流量镜像筛选条件规则的描述。

```
aws ec2 modify-traffic-mirror-filter-rule \
    --traffic-mirror-filter-rule-id tmfr-0ca76e0e08EXAMPLE \
    --description "TCP Rule"
```

输出：

```
{
    "TrafficMirrorFilterRule": {
```

```
        "TrafficMirrorFilterRuleId": "tmfr-0ca76e0e08EXAMPLE",
        "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",
        "TrafficDirection": "ingress",
        "RuleNumber": 100,
        "RuleAction": "accept",
        "Protocol": 6,
        "DestinationCidrBlock": "10.0.0.0/24",
        "SourceCidrBlock": "10.0.0.0/24",
        "Description": "TCP Rule"
    }
}
```

有关更多信息，请参阅《Amazon 流量镜像指南》中的修改流量镜像筛选条件规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTrafficMirrorFilterRule。

modify-traffic-mirror-session

以下代码示例演示了如何使用 modify-traffic-mirror-session。

Amazon CLI

修改流量镜像会话

以下 modify-traffic-mirror-session 示例更改流量镜像会话描述和要镜像的数据包数量。

```
aws ec2 modify-traffic-mirror-session \
    --description "Change packet length" \
    --traffic-mirror-session-id tms-08a33b1214EXAMPLE \
    --remove-fields "packet-length"
```

输出：

```
{
    "TrafficMirrorSession": {
        "TrafficMirrorSessionId": "tms-08a33b1214EXAMPLE",
        "TrafficMirrorTargetId": "tmt-07f75d8feeEXAMPLE",
        "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
        "NetworkInterfaceId": "eni-070203f901EXAMPLE",
        "OwnerId": "111122223333",
        "SessionNumber": 1,
        "VirtualNetworkId": 7159709,
```

```
        "Description": "Change packet length",
        "Tags": []
    }
}
```

有关更多信息，请参阅《流量镜像指南》中的修改流量镜像会话。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTrafficMirrorSession。

modify-transit-gateway-prefix-list-reference

以下代码示例演示了如何使用 modify-transit-gateway-prefix-list-reference。

Amazon CLI

修改对前缀列表的引用

以下 modify-transit-gateway-prefix-list-reference 示例通过更改将流量路由到的连接来修改指定路由表中的前缀列表引用。

```
aws ec2 modify-transit-gateway-prefix-list-reference \
    --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
    --prefix-list-id pl-1111112222222333 \
    --transit-gateway-attachment-id tgw-attach-aabbccddaabbccaab
```

输出：

```
{
    "TransitGatewayPrefixListReference": {
        "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
        "PrefixListId": "pl-1111112222222333",
        "PrefixListOwnerId": "123456789012",
        "State": "modifying",
        "Blackhole": false,
        "TransitGatewayAttachment": {
            "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",
            "ResourceType": "vpc",
            "ResourceId": "vpc-112233445566aabbc"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的前缀列表引用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTransitGatewayPrefixListReference。

modify-transit-gateway-vpc-attachment

以下代码示例演示了如何使用 modify-transit-gateway-vpc-attachment。

Amazon CLI

修改中转网关 VPC 连接

以下 modify-transit-gateway-vpc-attachment 示例将子网添加到指定的中转网关 VPC 连接。

```
aws ec2 modify-transit-gateway-vpc-attachment \
    --transit-gateway-attachment-id tgw-attach-09fbd47ddfEXAMPLE \
    --add-subnet-ids subnet-0e51f45802EXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-09fbd47ddfEXAMPLE",
        "TransitGatewayId": "tgw-0560315ccfEXAMPLE",
        "VpcId": "vpc-5eccc927",
        "VpcOwnerId": "111122223333",
        "State": "modifying",
        "SubnetIds": [
            "subnet-0e51f45802EXAMPLE",
            "subnet-1EXAMPLE"
        ],
        "CreationTime": "2019-08-08T16:47:38.000Z",
        "Options": {
            "DnsSupport": "enable",
            "Ipv6Support": "disable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 VPC 的中转网关连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTransitGatewayVpcAttachment。

`modify-transit-gateway`

以下代码示例演示了如何使用 `modify-transit-gateway`。

Amazon CLI

修改中转网关

以下 `modify-transit-gateway` 示例通过启用 ECMP 对 VPN 连接的支持来修改指定的中转网关。

```
aws ec2 modify-transit-gateway \
    --transit-gateway-id tgw-111111222222aaaaa \
    --options VpnEcmpSupport=enable
```

输出：

```
{
    "TransitGateway": {
        "TransitGatewayId": "tgw-111111222222aaaaa",
        "TransitGatewayArn": "64512",
        "State": "modifying",
        "OwnerId": "123456789012",
        "CreationTime": "2020-04-30T08:41:37.000Z",
        "Options": {
            "AmazonSideAsn": 64512,
            "AutoAcceptSharedAttachments": "disable",
            "DefaultRouteTableAssociation": "enable",
            "AssociationDefaultRouteTableId": "tgw-rtb-0123456789abcd123",
            "DefaultRouteTablePropagation": "enable",
            "PropagationDefaultRouteTableId": "tgw-rtb-0123456789abcd123",
            "VpnEcmpSupport": "enable",
            "DnsSupport": "enable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTransitGateway。

modify-verified-access-endpoint-policy

以下代码示例演示了如何使用 modify-verified-access-endpoint-policy。

Amazon CLI

为端点配置已验证访问策略

以下 modify-verified-access-endpoint-policy 示例将指定的已验证访问策略添加到指定的已验证访问端点。

```
aws ec2 modify-verified-access-endpoint-policy \
    --verified-access-endpoint-id vae-066fac616d4d546f2 \
    --policy-enabled \
    --policy-document file://policy.txt
```

的内容policy.txt：

```
permit(principal,action,resource)
when {
    context.identity.groups.contains("finance") &&
    context.identity.email.verified == true
};
```

输出：

```
{
    "PolicyEnabled": true,
    "PolicyDocument": "permit(principal,action,resource)\nwhen
 {\n    context.identity.groups.contains(\"finance\") &&\n
 context.identity.email_verified == true\n};"
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessEndpointPolicy。

modify-verified-access-endpoint

以下代码示例演示了如何使用 modify-verified-access-endpoint。

Amazon CLI

修改已验证访问端点的配置

以下 modify-verified-access-endpoint 示例将指定的描述添加到指定的已验证访问端点。

```
aws ec2 modify-verified-access-endpoint \
    --verified-access-endpoint-id vae-066fac616d4d546f2 \
    --description 'Testing Verified Access'
```

输出：

```
{
    "VerifiedAccessEndpoint": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
        "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
        "ApplicationDomain": "example.com",
        "EndpointType": "network-interface",
        "AttachmentType": "vpc",
        "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
        "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
        "SecurityGroupIds": [
            "sg-004915970c4c8f13a"
        ],
        "NetworkInterfaceOptions": {
            "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
            "Protocol": "https",
            "Port": 443
        },
        "Status": {
            "Code": "updating"
        },
        "Description": "Testing Verified Access",
        "CreationTime": "2023-08-25T20:54:43",
        "LastUpdatedTime": "2023-08-25T22:46:32"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessEndpoint。

modify-verified-access-group-policy

以下代码示例演示了如何使用 modify-verified-access-group-policy。

Amazon CLI

为组配置已验证访问策略

以下 modify-verified-access-group-policy 示例将指定的已验证访问策略添加到指定的已验证访问组。

```
aws ec2 modify-verified-access-group-policy \
    --verified-access-group-id vagr-0dbe967baf14b7235 \
    --policy-enabled \
    --policy-document file://policy.txt
```

的内容policy.txt：

```
permit(principal,action,resource)
when {
    context.identity.groups.contains("finance") &&
    context.identity.email.verified == true
};
```

输出：

```
{
    "PolicyEnabled": true,
    "PolicyDocument": "permit(principal,action,resource)\nwhen
 {\n    context.identity.groups.contains(\"finance\") &&\n
 context.identity.email_verified == true\n};"
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessGroupPolicy。

modify-verified-access-group

以下代码示例演示了如何使用 modify-verified-access-group。

Amazon CLI

修改已验证访问组的配置

以下 modify-verified-access-group 示例将指定的描述添加到指定的已验证访问组。

```
aws ec2 modify-verified-access-group \
    --verified-access-group-id vagr-0dbe967baf14b7235 \
    --description "Testing Verified Access"
```

输出：

```
{
    "VerifiedAccessGroup": {
        "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "Testing Verified Access",
        "Owner": "123456789012",
        "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
        "CreationTime": "2023-08-25T19:55:19",
        "LastUpdatedTime": "2023-08-25T22:17:25"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessGroup。

modify-verified-access-instance-logging-configuration

以下代码示例演示了如何使用 modify-verified-access-instance-logging-configuration。

Amazon CLI

为已验证访问实例启用日志记录

以下 modify-verified-access-instance-logging-configuration 示例为指定的已验证访问实例启用访问日志记录。日志将传输到指定的 CloudWatch Logs 日志组。

```
aws ec2 modify-verified-access-instance-logging-configuration \
```

```
    --verified-access-instance-id vai-0ce000c0b7643abea \
    --access-logs CloudWatchLogs={Enabled=true,LogGroup=my-log-group}
```

输出：

```
{
    "LoggingConfiguration": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "AccessLogs": {
            "S3": {
                "Enabled": false
            },
            "CloudWatchLogs": {
                "Enabled": true,
                "DeliveryStatus": {
                    "Code": "success"
                },
                "LogGroup": "my-log-group"
            },
            "KinesisDataFirehose": {
                "Enabled": false
            },
            "LogVersion": "ocsf-1.0.0-rc.2",
            "IncludeTrustContext": false
        }
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessInstanceLoggingConfiguration。

modify-verified-access-instance

以下代码示例演示了如何使用 modify-verified-access-instance。

Amazon CLI

修改已验证访问实例的配置

以下 modify-verified-access-instance 示例将指定的描述添加到指定的已验证访问实例。

```
aws ec2 modify-verified-access-instance \
    --verified-access-instance-id vai-0ce000c0b7643abea \
    --description "Testing Verified Access"
```

输出：

```
{
    "VerifiedAccessInstance": {
        "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
        "Description": "Testing Verified Access",
        "VerifiedAccessTrustProviders": [
            {
                "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
                "TrustProviderType": "user",
                "UserTrustProviderType": "iam-identity-center"
            }
        ],
        "CreationTime": "2023-08-25T18:27:56",
        "LastUpdatedTime": "2023-08-25T22:41:04"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessInstance。

modify-verified-access-trust-provider

以下代码示例演示了如何使用 modify-verified-access-trust-provider。

Amazon CLI

修改已验证访问信任提供商的配置

以下 modify-verified-access-trust-provider 示例将指定的描述添加到指定的已验证访问信任提供商。

```
aws ec2 modify-verified-access-trust-provider \
    --verified-access-trust-provider-id vatp-0bb32de759a3e19e7 \
    --description "Testing Verified Access"
```

输出：

```
{
    "VerifiedAccessTrustProvider": {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "Description": "Testing Verified Access",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center",
        "PolicyReferenceName": "idc",
        "CreationTime": "2023-08-25T19:00:38",
        "LastUpdatedTime": "2023-08-25T19:18:21"
    }
}
```

有关更多信息，请参阅《Amazon 已验证访问用户指南》中的已验证访问的信任提供商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVerifiedAccessTrustProvider。

modify-volume-attribute

以下代码示例演示了如何使用 modify-volume-attribute。

Amazon CLI

修改卷属性

此示例将 ID 为 vol-1234567890abcdef0 的卷的 autoEnableIo 属性设置为 true。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-io
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVolumeAttribute。

modify-volume

以下代码示例演示了如何使用 modify-volume。

Amazon CLI

示例 1：通过更改卷大小来修改卷

以下 modify-volume 示例将指定卷的大小更改为 150 GB。

命令:

```
aws ec2 modify-volume --size 150 --volume-id vol-1234567890abcdef0
```

输出：

```
{
    "VolumeModification": {
        "TargetSize": 150,
        "TargetVolumeType": "io1",
        "ModificationState": "modifying",
        "VolumeId": " vol-1234567890abcdef0",
        "TargetIops": 100,
        "StartTime": "2019-05-17T11:27:19.000Z",
        "Progress": 0,
        "OriginalVolumeType": "io1",
        "OriginalIops": 100,
        "OriginalSize": 100
    }
}
```

示例 2：通过更改卷的类型、大小和 IOPS 值来修改卷

以下 modify-volume 示例将卷类型更改为预置 IOPS SSD，将目标 IOPS 速率设置为 10000，并将卷大小设置为 350 GB。

```
aws ec2 modify-volume \
    --volume-type io1 \
    --iops 10000 \
    --size 350 \
    --volume-id vol-1234567890abcdef0
```

输出：

```
{
    "VolumeModification": {
        "TargetSize": 350,
        "TargetVolumeType": "io1",
        "ModificationState": "modifying",
```

```
        "VolumeId": "vol-0721c1a9d08c93bf6",
        "TargetIops": 10000,
        "StartTime": "2019-05-17T11:38:57.000Z",
        "Progress": 0,
        "OriginalVolumeType": "gp2",
        "OriginalIops": 150,
        "OriginalSize": 50
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVolume。

modify-vpc-attribute

以下代码示例演示了如何使用 modify-vpc-attribute。

Amazon CLI

修改 enableDnsSupport 属性

此示例修改 enableDnsSupport 属性。此属性指示是否为 VPC 启用 DNS 解析。如果该属性为 true，则 Amazon DNS 服务器会将您实例的 DNS 主机名称解析为相应的 IP 地址，否则不会解析。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value
\":false}"
```

修改 enableDnsHostnames 属性

此示例修改 enableDnsHostnames 属性。此属性指示在 VPC 中启动的实例是否可获得 DNS 主机名。如果该属性为 true，则 VPC 内的实例可获得 DNS 主机名称，否则将无法获得。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames "{\"Value
\":false}"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcAttribute。

modify-vpc-endpoint-connection-notification

以下代码示例演示了如何使用 modify-vpc-endpoint-connection-notification。

Amazon CLI

修改端点连接通知

此示例更改指定端点连接通知的 SNS 主题。

命令:

```
aws ec2 modify-vpc-endpoint-connection-notification --connection-notification-
id vpce-nfn-008776de7e03f5abc --connection-events Accept Reject --connection-
notification-arn arn:aws:sns:us-east-2:123456789012:mytopic
```

输出：

```
{
   "ReturnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcEndpointConnectionNotification。

modify-vpc-endpoint-service-configuration

以下代码示例演示了如何使用 modify-vpc-endpoint-service-configuration。

Amazon CLI

修改端点服务配置

此示例更改指定端点服务的接受要求。

命令:

```
aws ec2 modify-vpc-endpoint-service-configuration --service-id vpce-
svc-09222513e6e77dc86 --no-acceptance-required
```

输出：

```
{
   "ReturnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcEndpointServiceConfiguration。

modify-vpc-endpoint-service-payer-responsibility

以下代码示例演示了如何使用 modify-vpc-endpoint-service-payer-responsibility。

Amazon CLI

修改付款人责任

以下 modify-vpc-endpoint-service-payer-responsibility 示例修改指定端点服务的付款人责任。

```
aws ec2 modify-vpc-endpoint-service-payer-responsibility \
    --service-id vpce-svc-071afff70666e61e0 \
    --payer-responsibility ServiceOwner
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcEndpointServicePayerResponsibility。

modify-vpc-endpoint-service-permissions

以下代码示例演示了如何使用 modify-vpc-endpoint-service-permissions。

Amazon CLI

修改端点服务权限

此示例为 Amazon 账户添加了连接到指定端点服务的权限。

命令：

```
aws ec2 modify-vpc-endpoint-service-permissions --service-id vpce-
svc-03d5ebb7d9579a2b3 --add-allowed-principals '["arn:aws:iam::123456789012:root"]'
```

输出：

```
{
   "ReturnValue": true
}
```

此示例为特定 IAM 用户 (admin) 添加了连接到指定端点服务的权限。

命令:

```
aws ec2 modify-vpc-endpoint-service-permissions --service-id vpce-
svc-03d5ebb7d9579a2b3 --add-allowed-principals '["arn:aws:iam::123456789012:user/
admin"]'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcEndpointServicePermissions。

modify-vpc-endpoint

以下代码示例演示了如何使用 modify-vpc-endpoint。

Amazon CLI

修改网关端点

此示例通过将路由表 rtb-aaa222bb 与端点相关联并重置策略文档来修改网关端点 vpce-1a2b3c4d。

命令:

```
aws ec2 modify-vpc-endpoint --vpc-endpoint-id vpce-1a2b3c4d --add-route-table-
ids rtb-aaa222bb --reset-policy
```

输出：

```
{
  "Return": true
```

```
}
```

修改接口端点

此示例通过向端点添加子网 subnet-d6fcaa8d 来修改接口端点 vpce-0fe5b17a0707d6fa5。

命令:

```
aws ec2 modify-vpc-endpoint --vpc-endpoint-id vpce-0fe5b17a0707d6fa5 --add-subnet-
id subnet-d6fcaa8d
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcEndpoint。

modify-vpc-peering-connection-options

以下代码示例演示了如何使用 modify-vpc-peering-connection-options。

Amazon CLI

通过 VPC 对等连接启用从本地 ClassicLink 连接进行通信

在此示例中，对于对等连接 pcx-aaaabbb，请求者 VPC 的所有者修改 VPC 对等连接选项，以使本地 ClassicLink 连接能够与对等 VPC 进行通信。

命令:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-
connection-id pcx-aaaabbbb --requester-peering-connection-
options AllowEgressFromLocalClassicLinkToRemoteVpc=true
```

输出：

```
{
  "RequesterPeeringConnectionOptions": {
```

```
      "AllowEgressFromLocalClassicLinkToRemoteVpc": true
  }
}
```

通过 VPC 对等连接实现从本地 VPC 到远程 ClassicLink 连接的通信

在此示例中，接受者 VPC 的所有者修改 VPC 对等连接选项，使本地 VPC 能够与对等 VPC 中的 ClassicLink 连接进行通信。

命令:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-
connection-id pcx-aaaabbbb --accepter-peering-connection-
options AllowEgressFromLocalVpcToRemoteClassicLink=true
```

输出：

```
{
  "AccepterPeeringConnectionOptions": {
    "AllowEgressFromLocalVpcToRemoteClassicLink": true
  }
}
```

实现对 VPC 对等连接的 DNS 解析支持

在此示例中，请求者 VPC 的所有者修改 pcx-aaaabbbb 的 VPC 对等连接选项，以使本地 VPC 能够在接到对等 VPC 中的实例的查询时将公有 DNS 主机名解析为私有 IP 地址。

命令:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-
aaaabbbb --requester-peering-connection-options AllowDnsResolutionFromRemoteVpc=true
```

输出：

```
{
  "RequesterPeeringConnectionOptions": {
      "AllowDnsResolutionFromRemoteVpc": true
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcPeeringConnectionOptions。

modify-vpc-tenancy

以下代码示例演示了如何使用 modify-vpc-tenancy。

Amazon CLI

修改 VPC 的租赁

此示例将 VPC vpc-1a2b3c4d 的租赁修改为 default。

命令:

```
aws ec2 modify-vpc-tenancy --vpc-id vpc-1a2b3c4d --instance-tenancy default
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpcTenancy。

modify-vpn-connection-options

以下代码示例演示了如何使用 modify-vpn-connection-options。

Amazon CLI

修改 VPN 连接选项

以下 modify-vpn-connection-options 示例修改指定 VPN 连接的客户网关端的本地 IPv4 CIDR。

```
aws ec2 modify-vpn-connection-options \
    --vpn-connection-id vpn-1122334455aabbccd \
    --local-ipv4-network-cidr 10.0.0.0/16
```

输出：

```
{
    "VpnConnections": [
        {
            "CustomerGatewayConfiguration": "...configuration information...",
            "CustomerGatewayId": "cgw-01234567abcde1234",
            "Category": "VPN",
            "State": "modifying",
            "Type": "ipsec.1",
            "VpnConnectionId": "vpn-1122334455aabbccd",
            "TransitGatewayId": "tgw-00112233445566aab",
            "Options": {
                "EnableAcceleration": false,
                "StaticRoutesOnly": true,
                "LocalIpv4NetworkCidr": "10.0.0.0/16",
                "RemoteIpv4NetworkCidr": "0.0.0.0/0",
                "TunnelInsideIpVersion": "ipv4"
            },
            "Routes": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "CanadaVPN"
                }
            ],
            "VgwTelemetry": [
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-07-29T10:35:11.000Z",
                    "OutsideIpAddress": "203.0.113.3",
                    "Status": "DOWN",
                    "StatusMessage": ""
                },
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-09-02T09:09:33.000Z",
                    "OutsideIpAddress": "203.0.113.5",
                    "Status": "UP",
                    "StatusMessage": ""
                }
            ]
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Site-to-Site VPN 用户指南》中的修改 Site-to-Site VPN 连接选项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpnConnectionOptions。

`modify-vpn-connection`

以下代码示例演示了如何使用 `modify-vpn-connection`。

Amazon CLI

修改 VPN 连接

以下 `modify-vpn-connection` 示例将 VPN 连接 `vpn-12345678901234567` 的目标网关更改为虚拟私有网关 `vgw-11223344556677889`：

```
aws ec2 modify-vpn-connection \
    --vpn-connection-id vpn-12345678901234567 \
    --vpn-gateway-id vgw-11223344556677889
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "...configuration information...",
        "CustomerGatewayId": "cgw-aabbccddee1122334",
        "Category": "VPN",
        "State": "modifying",
        "Type": "ipsec.1",
        "VpnConnectionId": "vpn-12345678901234567",
        "VpnGatewayId": "vgw-11223344556677889",
        "Options": {
            "StaticRoutesOnly": false
        },
        "VgwTelemetry": [
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-07-17T07:34:00.000Z",
                "OutsideIpAddress": "18.210.3.222",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN"
```

```
            },
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-07-20T21:20:16.000Z",
                "OutsideIpAddress": "34.193.129.33",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpnConnection。

modify-vpn-tunnel-certificate

以下代码示例演示了如何使用 modify-vpn-tunnel-certificate。

Amazon CLI

轮换 VPN 隧道证书

以下 modify-vpn-tunnel-certificate 示例轮换 VPN 连接的指定隧道的证书。

```
aws ec2 modify-vpn-tunnel-certificate \
    --vpn-tunnel-outside-ip-address 203.0.113.17 \
    --vpn-connection-id vpn-12345678901234567
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": ...configuration information...,
        "CustomerGatewayId": "cgw-aabbccddee1122334",
        "Category": "VPN",
        "State": "modifying",
        "Type": "ipsec.1",
        "VpnConnectionId": "vpn-12345678901234567",
        "VpnGatewayId": "vgw-11223344556677889",
        "Options": {
            "StaticRoutesOnly": false
        },
```

```
        "VgwTelemetry": [
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-09-11T17:27:14.000Z",
                "OutsideIpAddress": "203.0.113.17",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN",
                "CertificateArn": "arn:aws:acm:us-east-1:123456789101:certificate/
c544d8ce-20b8-4fff-98b0-example"
            },
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-09-11T17:26:47.000Z",
                "OutsideIpAddress": "203.0.114.18",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN",
                "CertificateArn": "arn:aws:acm:us-
east-1:123456789101:certificate/5ab64566-761b-4ad3-b259-example"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpnTunnelCertificate。

`modify-vpn-tunnel-options`

以下代码示例演示了如何使用 `modify-vpn-tunnel-options`。

Amazon CLI

修改 VPN 连接的隧道选项

以下 `modify-vpn-tunnel-options` 示例将更新指定隧道和 VPN 连接允许的 Diffie-Hellman 组。

```
aws ec2 modify-vpn-tunnel-options \
    --vpn-connection-id vpn-12345678901234567 \
    --vpn-tunnel-outside-ip-address 203.0.113.17 \
    --tunnel-options Phase1DHGroupNumbers=[{Value=14},{Value=15},{Value=16},
{Value=17},{Value=18}],Phase2DHGroupNumbers=[{Value=14},{Value=15},{Value=16},
{Value=17},{Value=18}]
```

输出：

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "...configuration information...",
        "CustomerGatewayId": "cgw-aabbccddee1122334",
        "Category": "VPN",
        "State": "available",
        "Type": "ipsec.1",
        "VpnConnectionId": "vpn-12345678901234567",
        "VpnGatewayId": "vgw-11223344556677889",
        "Options": {
            "StaticRoutesOnly": false,
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.17",
                    "Phase1DHGroupNumbers": [
                        {
                            "Value": 14
                        },
                        {
                            "Value": 15
                        },
                        {
                            "Value": 16
                        },
                        {
                            "Value": 17
                        },
                        {
                            "Value": 18
                        }
                    ],
                    "Phase2DHGroupNumbers": [
                        {
                            "Value": 14
                        },
                        {
                            "Value": 15
                        },
                        {
                            "Value": 16
                        },
                        {
```

```
                            "Value": 17
                        },
                        {
                            "Value": 18
                        }
                    ]
                },
                {
                    "OutsideIpAddress": "203.0.114.19"
                }
            ]
        },
        "VgwTelemetry": [
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-09-10T21:56:54.000Z",
                "OutsideIpAddress": "203.0.113.17",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN"
            },
            {
                "AcceptedRouteCount": 0,
                "LastStatusChange": "2019-09-10T21:56:43.000Z",
                "OutsideIpAddress": "203.0.114.19",
                "Status": "DOWN",
                "StatusMessage": "IPSEC IS DOWN"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyVpnTunnelOptions。

monitor-instances

以下代码示例演示了如何使用 monitor-instances。

Amazon CLI

启用对实例的详细监控

本示例命令启用对指定实例的详细监控。

命令:

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

输出：

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "pending"
          }
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MonitorInstances。

move-address-to-vpc

以下代码示例演示了如何使用 move-address-to-vpc。

Amazon CLI

将地址移动到 EC2-VPC

此示例将弹性 IP 地址 54.123.4.56 移动到 EC2-VPC 平台。

命令:

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

输出：

```
{
  "Status": "MoveInProgress"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MoveAddressToVpc。

move-byoip-cidr-to-ipam

以下代码示例演示了如何使用 move-byoip-cidr-to-ipam。

Amazon CLI

将 BYOIP CIDR 传输到 IPAM

以下 move-byoip-cidr-to-ipam 示例将 BYOIP CIDR 传输到 IPAM。

（Linux）：

```
aws ec2 move-byoip-cidr-to-ipam \
    --region us-west-2 \
    --ipam-pool-id ipam-pool-0a03d430ca3f5c035 \
    --ipam-pool-owner 111111111111 \
    --cidr 130.137.249.0/24
```

（Windows）：

```
aws ec2 move-byoip-cidr-to-ipam ^
    --region us-west-2 ^
    --ipam-pool-id ipam-pool-0a03d430ca3f5c035 ^
    --ipam-pool-owner 111111111111 ^
    --cidr 130.137.249.0/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "130.137.249.0/24",
        "State": "pending-transfer"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的教程：将现有的 BYOIP IPv4 CIDR 传输到 IPAM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MoveByoipCidrToIpam。

network-insights-access-scope

以下代码示例演示了如何使用 network-insights-access-scope。

Amazon CLI

创建 Network Insights 访问范围

以下 `create-network-insights-access-scope` 示例在您的 Amazon 账户中创建 Network Insights 访问范围。

```
aws ec2 create-network-insights-access-scope \
    --cli-input-json file://access-scope-file.json
```

的内容`access-scope-file.json`：

```
{
    {
        "MatchPaths": [
            {
                "Source": {
                    "ResourceStatement": {
                         "Resources": [
                            "vpc-abcd12e3"
                        ]
                    }
                }
            }
        ],
        "ExcludePaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "ResourceTypes": [
                            "AWS::EC2::InternetGateway"
                        ]
                    }
                }
            }
        ]
    }
}
```

输出：

```
{
```

```
    "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789111"
    }{
    "NetworkInsightsAccessScope": {
        "NetworkInsightsAccessScopeId": "nis-123456789222",
        "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-
east-1:123456789222:network-insights-access-scope/nis-123456789222",
        "CreatedDate": "2022-01-25T19:20:28.796000+00:00",
        "UpdatedDate": "2022-01-25T19:20:28.797000+00:00"
    },
    "NetworkInsightsAccessScopeContent": {
        "NetworkInsightsAccessScopeId": "nis-04c0c0fbca737c404",
        "MatchPaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "Resources": [
                            "vpc-abcd12e3"
                        ]
                    }
                }
            }
        ],
        "ExcludePaths": [
            {
                "Source": {
                    "ResourceStatement": {
                        "ResourceTypes": [
                            "AWS::EC2::InternetGateway"
                        ]
                    }
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 NetworkInsightsAccessScope。

provision-byoip-cidr

以下代码示例演示了如何使用 provision-byoip-cidr。

Amazon CLI

预置地址范围

以下 provision-byoip-cidr 示例预置了一个公有 IP 地址范围以用于 Amazon。

```
aws ec2 provision-byoip-cidr \
    --cidr 203.0.113.25/24 \
    --cidr-authorization-context Message="$text_message",Signature="$signed_message"
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "203.0.113.25/24",
        "State": "pending-provision"
    }
}
```

有关为授权上下文创建消息字符串的更多信息，请参阅《Amazon EC2 用户指南》中的自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ProvisionByoipCidr。

provision-ipam-pool-cidr

以下代码示例演示了如何使用 provision-ipam-pool-cidr。

Amazon CLI

将 CIDR 预置到 IPAM 池

以下 provision-ipam-pool-cidr 示例将 CIDR 预置到 IPAM 池。

（Linux）：

```
aws ec2 provision-ipam-pool-cidr \
    --ipam-pool-id ipam-pool-0533048da7d823723 \
    --cidr 10.0.0.0/24
```

（Windows）：

```
aws ec2 provision-ipam-pool-cidr ^
    --ipam-pool-id ipam-pool-0533048da7d823723 ^
    --cidr 10.0.0.0/24
```

输出：

```
{
    "IpamPoolCidr": {
        "Cidr": "10.0.0.0/24",
        "State": "pending-provision"
    }
}
```

有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的将 CIDR 预置到池。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ProvisionIpamPoolCidr。

purchase-host-reservation

以下代码示例演示了如何使用 purchase-host-reservation。

Amazon CLI

购买专属主机预留

此示例为您的账户中的指定专属主机购买指定的专属主机预留产品。

命令:

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-
set h-013abcd2a00cbd123
```

输出：

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
```

```
            "HostIdSet": [
                "h-013abcd2a00cbd123"
            ],
            "HostReservationId": "hr-0d418a3a4ffc669ae",
            "UpfrontPrice": "0.000",
            "Duration": 31536000
        }
    ],
    "TotalUpfrontPrice": "0.000"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseHostReservation。

purchase-reserved-instances-offering

以下代码示例演示了如何使用 purchase-reserved-instances-offering。

Amazon CLI

购买预留实例产品

此示例命令阐明了预留实例产品的购买，并指定了产品 ID 和实例计数。

命令:

```
aws ec2 purchase-reserved-instances-offering --reserved-instances-offering-
id ec06327e-dd07-46ee-9398-75b5fexample --instance-count 3
```

输出：

```
{
    "ReservedInstancesId": "af9f760e-6f91-4559-85f7-4980eexample"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseReservedInstancesOffering。

purchase-scheduled-instances

以下代码示例演示了如何使用 purchase-scheduled-instances。

Amazon CLI

购买计划实例

此示例将购买计划实例。

命令:

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-request.json
```

Purchase-request.json：

```
[
    {
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "InstanceCount": 1
    }
]
```

输出：

```
{
  "ScheduledInstanceSet": [
      {
          "AvailabilityZone": "us-west-2b",
          "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
          "HourlyPrice": "0.095",
          "CreateDate": "2016-01-25T21:43:38.612Z",
          "Recurrence": {
              "OccurrenceDaySet": [
                  1
              ],
              "Interval": 1,
              "Frequency": "Weekly",
              "OccurrenceRelativeToEnd": false,
              "OccurrenceUnit": ""
          },
          "Platform": "Linux/UNIX",
          "TermEndDate": "2017-01-31T09:00:00Z",
          "InstanceCount": 1,
          "SlotDurationInHours": 32,
```

```
            "TermStartDate": "2016-01-31T09:00:00Z",
            "NetworkPlatform": "EC2-VPC",
            "TotalScheduledInstanceHours": 1696,
            "NextSlotStartTime": "2016-01-31T09:00:00Z",
            "InstanceType": "c4.large"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseScheduledInstances。

reboot-instances

以下代码示例演示了如何使用 reboot-instances。

Amazon CLI

重启 Amazon EC2 实例

本示例将重启指定的实例。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“重启实例”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootInstances。

register-image

以下代码示例演示了如何使用 register-image。

Amazon CLI

示例 1：使用清单文件注册 AMI

以下 register-image 示例在 Amazon S3 中使用指定的清单文件注册 AMI。

```
aws ec2 register-image \
    --name my-image \
    --image-location amzn-s3-demo-bucket/myimage/image.manifest.xml
```

输出：

```
{
    "ImageId": "ami-1234567890EXAMPLE"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的亚马逊机器映像（AMI）。

示例 2：使用根设备的快照注册 AMI

以下 `register-image` 示例使用 EBS 根卷的指定快照作为设备 `/dev/xvda` 注册 AMI。块设备映射还包括一个空的 100 GiB EBS 卷作为设备 `/dev/xvdf`。

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0db2cf683925d191f} DeviceName=/dev/
xvdf,Ebs={VolumeSize=100}
```

输出：

```
{
    "ImageId": "ami-1a2b3c4d5eEXAMPLE"
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的亚马逊机器映像（AMI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterImage。

register-instance-event-notification-attributes

以下代码示例演示了如何使用 `register-instance-event-notification-attributes`。

Amazon CLI

示例 1：在事件通知中包含所有标签

以下 `register-instance-event-notification-attributes` 示例在事件通知中包含所有标签。

```
aws ec2 register-instance-event-notification-attributes \
```

```
    --instance-tag-attribute IncludeAllTagsOfInstance=true
```

输出：

```
{
    "InstanceTagAttribute": {
        "InstanceTagKeys": [],
        "IncludeAllTagsOfInstance": true
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例的计划事件。

示例 2：在事件通知中包含特定标签

以下 register-instance-event-notification-attributes 示例在事件通知中包含指定标签。如果 IncludeAllTagsOfInstance 为 true，则无法指定标签。

```
aws ec2 register-instance-event-notification-attributes \
    --instance-tag-attribute InstanceTagKeys="tag-key1","tag-key2"
```

输出：

```
{
    "InstanceTagAttribute": {
        "InstanceTagKeys": [
            "tag-key1",
            "tag-key2"
        ],
        "IncludeAllTagsOfInstance": false
    }
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的实例的计划事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterInstanceEventNotificationAttributes。

register-transit-gateway-multicase-group-sources

以下代码示例演示了如何使用 register-transit-gateway-multicase-group-sources。

Amazon CLI

将源注册到中转网关多播组

以下 register-transit-gateway-multicast-group-sources 示例将指定的网络接口组源注册到多播组。

```
aws ec2 register-transit-gateway-multicast-group-sources \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
    --group-ip-address 224.0.1.0 \
    --network-interface-ids eni-07f290fc3c090cbae
```

输出：

```
{
    "RegisteredMulticastGroupSources": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
        "RegisteredNetworkInterfaceIds": [
            "eni-07f290fc3c090cbae"
        ],
        "GroupIpAddress": "224.0.1.0"
    }
}
```

有关更多信息，请参阅《Amazon 中转网关用户指南》中的将源注册到多播组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTransitGatewayMulticaseGroupSources。

register-transit-gateway-multicast-group-members

以下代码示例演示了如何使用 register-transit-gateway-multicast-group-members。

Amazon CLI

查看有关中转网关组播域关联的信息

以下 register-transit-gateway-multicast-group-members 示例返回指定组播域的关联。

```
aws ec2 register-transit-gateway-multicast-group-members \
```

```
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
    --group-ip-address 224.0.1.0 \
    --network-interface-ids eni-0e246d32695012e81
```

输出：

```
{
    "RegisteredMulticastGroupMembers": {
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
        "RegisteredNetworkInterfaceIds": [
            "eni-0e246d32695012e81"
        ],
        "GroupIpAddress": "224.0.1.0"
    }
}
```

有关更多信息，请参阅《Transit Gateways User Guide》中的 Multicast domains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTransitGatewayMulticastGroupMembers。

register-transit-gateway-multicast-group-sources

以下代码示例演示了如何使用 register-transit-gateway-multicast-group-sources。

Amazon CLI

将源注册到中转网关多播组

以下 register-transit-gateway-multicast-group-sources 示例将指定的网络接口组源注册到多播组。

```
aws ec2 register-transit-gateway-multicast-group-sources \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
    --group-ip-address 224.0.1.0 \
    --network-interface-ids eni-07f290fc3c090cbae
```

输出：

```
{
    "RegisteredMulticastGroupSources": {
```

```
        "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
        "RegisteredNetworkInterfaceIds": [
            "eni-07f290fc3c090cbae"
        ],
        "GroupIpAddress": "224.0.1.0"
    }
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Multicast domains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTransitGatewayMulticastGroupSources。

reject-transit-gateway-peering-attachment

以下代码示例演示了如何使用 reject-transit-gateway-peering-attachment。

Amazon CLI

拒绝中转网关对等连接

以下 reject-transit-gateway-peering-attachment 示例拒绝指定的中转网关对等连接请求。--region 参数指定接受方中转网关所在的区域。

```
aws ec2 reject-transit-gateway-peering-attachment \
    --transit-gateway-attachment-id tgw-attach-4455667788aabbccd \
    --region us-east-2
```

输出：

```
{
    "TransitGatewayPeeringAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
        "RequesterTgwInfo": {
            "TransitGatewayId": "tgw-123abc05e04123abc",
            "OwnerId": "123456789012",
            "Region": "us-west-2"
        },
        "AccepterTgwInfo": {
            "TransitGatewayId": "tgw-11223344aabbcc112",
            "OwnerId": "123456789012",
            "Region": "us-east-2"
```

```
        },
        "State": "rejecting",
        "CreationTime": "2019-12-09T11:50:31.000Z"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关对等连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectTransitGatewayPeeringAttachment。

reject-transit-gateway-vpc-attachment

以下代码示例演示了如何使用 reject-transit-gateway-vpc-attachment。

Amazon CLI

拒绝中转网关 VPC 连接

以下 reject-transit-gateway-vpc-attachment 示例拒绝指定的中转网关 VPC 连接。

```
aws ec2 reject-transit-gateway-vpc-attachment \
    --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
        "VpcId": "vpc-07e8ffd50fEXAMPLE",
        "VpcOwnerId": "111122223333",
        "State": "pending",
        "SubnetIds": [
            "subnet-0752213d59EXAMPLE"
        ],
        "CreationTime": "2019-07-10T17:33:46.000Z",
        "Options": {
            "DnsSupport": "enable",
            "Ipv6Support": "disable"
        }
    }
```

```
}
```

有关更多信息，请参阅《中转网关指南》中的 VPC 的中转网关连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectTransitGatewayVpcAttachment。

reject-transit-gateway-vpc-attachments

以下代码示例演示了如何使用 reject-transit-gateway-vpc-attachments。

Amazon CLI

拒绝中转网关 VPC 连接

以下 reject-transit-gateway-vpc-attachment 示例拒绝指定的中转网关 VPC 连接。

```
aws ec2 reject-transit-gateway-vpc-attachment \
    --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE
```

输出：

```
{
    "TransitGatewayVpcAttachment": {
        "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
        "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
        "VpcId": "vpc-07e8ffd50fEXAMPLE",
        "VpcOwnerId": "111122223333",
        "State": "pending",
        "SubnetIds": [
            "subnet-0752213d59EXAMPLE"
        ],
        "CreationTime": "2019-07-10T17:33:46.000Z",
        "Options": {
            "DnsSupport": "enable",
            "Ipv6Support": "disable"
        }
    }
}
```

有关更多信息，请参阅《中转网关指南》中的 VPC 的中转网关连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectTransitGatewayVpcAttachments。

reject-vpc-endpoint-connections

以下代码示例演示了如何使用 reject-vpc-endpoint-connections。

Amazon CLI

拒绝接口端点连接请求

此示例拒绝指定端点服务的指定端点连接请求。

命令:

```
aws ec2 reject-vpc-endpoint-connections --service-id vpce-svc-03d5ebb7d9579a2b3 --
vpc-endpoint-ids vpce-0c1308d7312217abc
```

输出：

```
{
  "Unsuccessful": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectVpcEndpointConnections。

reject-vpc-peering-connection

以下代码示例演示了如何使用 reject-vpc-peering-connection。

Amazon CLI

拒绝 VPC 对等连接

此示例拒绝指定的 VPC 对等连接请求。

命令:

```
aws ec2 reject-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

输出：

```
{
    "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectVpcPeeringConnection。

release-address

以下代码示例演示了如何使用 release-address。

Amazon CLI

为 EC2-Classic 释放弹性 IP 地址

本示例将释放一个弹性 IP 地址，供 EC2-Classic 中的实例使用。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 release-address --public-ip 198.51.100.0
```

为 EC2-VPC 释放弹性 IP 地址

本示例将释放一个弹性 IP 地址，供 VPC 中的实例使用。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReleaseAddress。

release-hosts

以下代码示例演示了如何使用 release-hosts。

Amazon CLI

从账户中释放专属主机

从您的账户中释放专属主机。必须先停止或终止主机上的实例，然后才能释放主机。

命令:

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

输出：

```
{
    "Successful":  [
        "h-0029d6e3cacf1b3da"
         ],
  "Unsuccessful": []

 }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReleaseHosts。

release-ipam-pool-allocation

以下代码示例演示了如何使用 release-ipam-pool-allocation。

Amazon CLI

释放 IPAM 池分配

在此示例中，您是 IPAM 委派管理员，尝试删除 IPAM 池，但收到一条错误，指出当池具有分配时无法删除该池。您正在使用此命令释放池分配。

请注意以下几点：

此命令仅可用于自定义分配。要在不删除资源的情况下删除资源的分配，请使用 modify-ipam-resource-cidr 将其受监控状态设置为 false。要完成此请求，您需要 IPAM 池 ID，您可以使用 describe-ipam-pools 获取该 ID。您还需要分配 ID，您可以使用 get-ipam-pool-allocations 获取该 ID。如果您不想逐一删除分配，可以在删除 IPAM 池时使用 --cascade option 自动释放池中的任何分配，然后再将其删除。在运行此命令之前，有许多先决条件。有关更多信息，请参阅《Amazon VPC IPAM 用户指南》中的释放分配。在其中运行此命令的 --region 必须是分配所在的 IPAM 池的区域设置。

以下 release-ipam-pool-allocation 示例释放 IPAM 池分配。

```
aws ec2 release-ipam-pool-allocation \
    --ipam-pool-id ipam-pool-07bdd12d7c94e4693 \
    --cidr 10.0.0.0/23 \
    --ipam-pool-allocation-id ipam-pool-alloc-0e66a1f730da54791b99465b79e7d1e89 \
    --region us-west-1
```

输出：

```
{
    "Success": true
}
```

释放分配后，您可能需要运行 delete-ipam-pool。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReleaseIpamPoolAllocation。

replace-iam-instance-profile-association

以下代码示例演示了如何使用 replace-iam-instance-profile-association。

Amazon CLI

替换实例的 IAM 实例配置文件

本示例将关联 iip-assoc-060bae234aac2e7fa 表示的 IAM 实例配置文件，替换为名为 AdminRole 的 IAM 实例配置文件。

```
aws ec2 replace-iam-instance-profile-association \
    --iam-instance-profile Name=AdminRole \
    --association-id iip-assoc-060bae234aac2e7fa
```

输出：

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-087711ddaf98f9489",
        "State": "associating",
        "AssociationId": "iip-assoc-0b215292fab192820",
        "IamInstanceProfile": {
            "Id": "AIPAJLNLDX3AMYZNWYYAY",
            "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceIamInstanceProfileAssociation。

replace-network-acl-association

以下代码示例演示了如何使用 replace-network-acl-association。

Amazon CLI

替换与子网关联的网络 ACL

此示例将指定的网络 ACL 与指定网络 ACL 关联的子网相关联。

命令:

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --
network-acl-id acl-5fb85d36
```

输出：

```
{
    "NewAssociationId": "aclassoc-3999875b"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceNetworkAclAssociation。

replace-network-acl-entry

以下代码示例演示了如何使用 replace-network-acl-entry。

Amazon CLI

替换网络 ACL 条目

此示例将替换指定网络 ACL 的一个条目。新规则 100 允许来自 UDP 端口 53（DNS）上的 203.0.113.12/24 的入口流量进入任何关联的子网。

命令:

```
aws ec2 replace-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100 --protocol udp --port-range From=53,To=53 --cidr-block 203.0.113.12/24 --rule-action allow
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceNetworkAclEntry。

replace-route-table-association

以下代码示例演示了如何使用 replace-route-table-association。

Amazon CLI

替换与子网关联的路由表

此示例将指定的路由表与指定路由表关联的子网相关联。

命令:

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --route-table-id rtb-22574640
```

输出：

```
{
    "NewAssociationId": "rtbassoc-3a1f0f58"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceRouteTableAssociation。

replace-route

以下代码示例演示了如何使用 replace-route。

Amazon CLI

替换路由

此示例将替换指定路由表中的指定路由。新路由与指定的 CIDR 匹配，并将流量发送到指定的虚拟私有网关。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-
block 10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceRoute。

replace-transit-gateway-route

以下代码示例演示了如何使用 replace-transit-gateway-route。

Amazon CLI

替换指定中转网关路由表中的指定路由

以下 replace-transit-gateway-route 示例将替换指定中转网关路由表中的路由。

```
aws ec2 replace-transit-gateway-route \
    --destination-cidr-block 10.0.2.0/24 \
    --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE \
    --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "10.0.2.0/24",
        "TransitGatewayAttachments": [
            {
                "ResourceId": "vpc-4EXAMPLE",
                "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
                "ResourceType": "vpc"
            }
        ],
        "Type": "static",
        "State": "active"
    }
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceTransitGatewayRoute。

report-instance-status

以下代码示例演示了如何使用 report-instance-status。

Amazon CLI

报告实例的状态反馈

此示例命令报告指定实例的状态反馈。

命令:

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired --
reason-codes unresponsive
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReportInstanceStatus。

request-spot-fleet

以下代码示例演示了如何使用 request-spot-fleet。

Amazon CLI

请求子网中价格最低的竞价型实例集

此示例命令创建一个竞价型实例集请求，其中包含两个仅因子网而异的启动规范。竞价型实例集启动指定子网中价格最低的实例。如果在默认 VPC 中启动实例，则实例在默认情况下会收到一个公有 IP 地址。如果在非默认 VPC 中启动实例，则默认情况下它们不会收到一个公有 IP 地址。

请注意，您无法在竞价型实例集请求中指定来自相同可用区的不同子网。

命令:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json：

```
{
```

```
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "SubnetId": "subnet-1a2b3c4d, subnet-3c4d5e6f",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

输出：

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

请求可用区中价格最低的竞价型实例集

此示例命令创建一个竞价型实例集请求，其中包含两个仅因可用区而异的启动规范。竞价型实例集启动指定可用区中价格最低的实例。如果您的账户仅支持 EC2-VPC，则 Amazon EC2 会在可用区的默认子网中启动竞价型实例。如果您的账户支持 EC2-Classic，则 Amazon EC2 会在可用区中的 EC2-Classic 中启动实例。

命令:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json：

```
{
```

```
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "Placement": {
              "AvailabilityZone": "us-west-2a, us-west-2b"
          },
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

在子网中启动竞价型实例并为其分配公有 IP 地址

此示例命令将公有地址分配给在非默认 VPC 中启动的实例。请注意，指定网络接口时，您必须包括使用网络接口的子网 ID 和安全组 ID。

命令:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json：

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
```

```
            "InstanceType": "m3.medium",
            "NetworkInterfaces": [
                {
                    "DeviceIndex": 0,
                    "SubnetId": "subnet-1a2b3c4d",
                    "Groups": [ "sg-1a2b3c4d" ],
                    "AssociatePublicIpAddress": true
                }
            ],
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::880185128111:instance-profile/my-iam-role"
            }
        }
    ]
}
```

使用多样化分配策略请求竞价型实例集

此示例命令创建一个竞价型实例集请求，该请求使用多样化分配策略启动 30 个实例。启动规范因实例类型而异。竞价型实例集在启动规范间分配实例，以便每种类型有 10 个实例。

命令:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json：

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      },
```

```
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“竞价型实例集请求”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestSpotFleet。

request-spot-instances

以下代码示例演示了如何使用 request-spot-instances。

Amazon CLI

请求竞价型实例

此示例命令为指定可用区中的五个实例创建一次性竞价型实例请求。如果您的账户仅支持 EC2-VPC，则 Amazon EC2 会在指定可用区的默认子网中启动实例。如果您的账户支持 EC2-Classic，则 Amazon EC2 会在指定可用区中的 EC2-Classic 中启动实例。

命令:

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --type "one-time" --launch-specification file://specification.json
```

Specification.json：

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
```

```
}
```

输出：

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
              "UpdateTime": "2014-03-25T20:54:21.000Z",
              "Code": "pending-evaluation",
              "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
          "SpotInstanceRequestId": "sir-df6f405d",
          "State": "open",
          "LaunchSpecification": {
              "Placement": {
                  "AvailabilityZone": "us-west-2a"
              },
              "ImageId": "ami-1a2b3c4d",
              "KeyName": "my-key-pair",
              "SecurityGroups": [
                  {
                      "GroupName": "my-security-group",
                      "GroupId": "sg-1a2b3c4d"
                  }
              ],
              "Monitoring": {
                  "Enabled": false
              },
              "IamInstanceProfile": {
                  "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
              },
              "InstanceType": "m3.medium"
          },
          "Type": "one-time",
          "CreateTime": "2014-03-25T20:54:20.000Z",
          "SpotPrice": "0.050000"
      },
      ...
  ]
}
```

此示例命令为指定子网中的五个实例创建一次性竞价型实例请求。Amazon EC2 会在指定子网中启动实例。如果 VPC 是一个非默认 VPC，则默认情况下，实例不会收到公有 IP 地址。

命令:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --type "one-time" --launch-specification file://specification.json
```

Specification.json：

```
{
  "ImageId": "ami-1a2b3c4d",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

输出：

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
             "UpdateTime": "2014-03-25T22:21:58.000Z",
             "Code": "pending-evaluation",
             "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
          "SpotInstanceRequestId": "sir-df6f405d",
          "State": "open",
          "LaunchSpecification": {
             "Placement": {
                 "AvailabilityZone": "us-west-2a"
             }
             "ImageId": "ami-1a2b3c4d"
             "SecurityGroups": [
                 {
                     "GroupName": "my-security-group",
```

```
                    "GroupID": "sg-1a2b3c4d"
                }
            ]
            "SubnetId": "subnet-1a2b3c4d",
            "Monitoring": {
                "Enabled": false
            },
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
            },
            "InstanceType": "m3.medium",
        },
        "Type": "one-time",
        "CreateTime": "2014-03-25T22:21:58.000Z",
        "SpotPrice": "0.050000"
    },
    ...
  ]
}
```

此示例为您在非默认 VPC 中启动的竞价型实例分配公有 IP 地址。请注意，指定网络接口时，您必须包括使用网络接口的子网 ID 和安全组 ID。

命令:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --type "one-time" --launch-specification file://specification.json
```

Specification.json：

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "InstanceType": "m3.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d",
      "Groups": [ "sg-1a2b3c4d" ],
      "AssociatePublicIpAddress": true
    }
  ],
```

```
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestSpotInstances。

reset-address-attribute

以下代码示例演示了如何使用 reset-address-attribute。

Amazon CLI

重置与弹性 IP 地址关联的域名属性

以下 reset-address-attribute 示例重置弹性 IP 地址的域名属性。

Linux：

```
aws ec2 reset-address-attribute \
    --allocation-id eipalloc-abcdef01234567890 \
    --attribute domain-name
```

Windows:

```
aws ec2 reset-address-attribute ^
    --allocation-id eipalloc-abcdef01234567890 ^
    --attribute domain-name
```

输出：

```
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.com."
            "PtrRecordUpdate": {
                "Value": "example.net.",
                "Status": "PENDING"
        }
```

```
    ]
}
```

要监控待定更改，请参阅《Amazon CLI 命令参考》中的 describe-addresses-attribute。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetAddressAttribute。

reset-ebs-default-kms-key-id

以下代码示例演示了如何使用 reset-ebs-default-kms-key-id。

Amazon CLI

重置 EBS 加密的默认 CMK

以下 reset-ebs-default-kms-key-id 示例重置当前区域中 Amazon 账户的 EBS 加密的默认 CMK。

```
aws ec2 reset-ebs-default-kms-key-id
```

输出：

```
{
   "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/8c5b2c63-b9bc-45a3-
a87a-5513eEXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetEbsDefaultKmsKeyId。

reset-fpga-image-attribute

以下代码示例演示了如何使用 reset-fpga-image-attribute。

Amazon CLI

重置 Amazon FPGA 映像的属性

此示例重置指定 AFI 的加载权限。

命令:

```
aws ec2 reset-fpga-image-attribute --fpga-image-id afi-0d123e123bfc85abc --
attribute loadPermission
```

输出：

```
{
  "Return": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetFpgaImageAttribute。

reset-image-attribute

以下代码示例演示了如何使用 reset-image-attribute。

Amazon CLI

重置 launchPermission 属性

此示例将指定 AMI 的 launchPermission 属性重置为其默认值。默认情况下，AMI 是私有的。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reset-image-attribute --image-id ami-5731123e --attribute launchPermission
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetImageAttribute。

reset-instance-attribute

以下代码示例演示了如何使用 reset-instance-attribute。

Amazon CLI

重置 sourceDestCheck 属性

此示例重置指定实例的 sourceDestCheck 属性。该实例必须在 VPC 中。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --
attribute sourceDestCheck
```

重置内核属性

此示例重置指定实例的 kernel 属性。该实例必须处于 stopped 状态。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --
attribute kernel
```

重置虚拟磁盘属性

此示例重置指定实例的 ramdisk 属性。该实例必须处于 stopped 状态。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --
attribute ramdisk
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetInstanceAttribute。

reset-network-interface-attribute

以下代码示例演示了如何使用 reset-network-interface-attribute。

Amazon CLI

重置网络接口属性

以下 reset-network-interface-attribute 示例将源/目标检查属性的值重置为 true。

```
aws ec2 reset-network-interface-attribute \
    --network-interface-id eni-686ea200 \
    --source-dest-check
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetNetworkInterfaceAttribute。

reset-snapshot-attribute

以下代码示例演示了如何使用 reset-snapshot-attribute。

Amazon CLI

重置快照属性

此示例重置快照 snap-1234567890abcdef0 的创建卷权限。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --
attribute createVolumePermission
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetSnapshotAttribute。

restore-address-to-classic

以下代码示例演示了如何使用 restore-address-to-classic。

Amazon CLI

将地址还原到 EC2-Classic

此示例将弹性 IP 地址 198.51.100.0 还原到 EC2-Classic 平台。

命令:

```
aws ec2 restore-address-to-classic --public-ip 198.51.100.0
```

输出：

```
{
  "Status": "MoveInProgress",
  "PublicIp": "198.51.100.0"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreAddressToClassic。

`restore-image-from-recycle-bin`

以下代码示例演示了如何使用 `restore-image-from-recycle-bin`。

Amazon CLI

从回收站中恢复映像

以下 `restore-image-from-recycle-bin` 示例从回收站中恢复 AMI ami-0111222333444abcd。

```
aws ec2 restore-image-from-recycle-bin \
    --image-id ami-0111222333444abcd
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EBS User Guide》中的 Recover deleted AMIs from the Recycle Bin。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreImageFromRecycleBin。

`restore-managed-prefix-list-version`

以下代码示例演示了如何使用 `restore-managed-prefix-list-version`。

Amazon CLI

us-west-2**恢复前缀列表版本**

以下 `restore-managed-prefix-list-version` 从指定前缀列表的版本 1 中恢复条目。

```
aws ec2 restore-managed-prefix-list-version \
    --prefix-list-id pl-0123456abcabcabc1 \
    --current-version 2 \
```

```
    --previous-version 1
```

输出：

```
{
    "PrefixList": {
        "PrefixListId": "pl-0123456abcabcabc1",
        "AddressFamily": "IPv4",
        "State": "restore-in-progress",
        "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
        "PrefixListName": "vpc-cidrs",
        "MaxEntries": 10,
        "Version": 2,
        "OwnerId": "123456789012"
    }
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的托管前缀列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreManagedPrefixListVersion。

`restore-snapshot-from-recycle-bin`

以下代码示例演示了如何使用 `restore-snapshot-from-recycle-bin`。

Amazon CLI

从回收站中还原快照

以下 `restore-snapshot-from-recycle-bin` 示例从回收站中还原快照。从回收站还原快照时，该快照立即可供使用，回收站会将其删除。还原快照后，您可以像使用账户中任何其它快照一样使用它。

```
aws ec2 restore-snapshot-from-recycle-bin \
    --snapshot-id snap-01234567890abcdef
```

此命令不生成任何输出。

有关回收站的更多信息，请参阅《Amazon EBS User Guide》中的 Recover deleted snapshots from the Recycle Bin。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreSnapshotFromRecycleBin。

`restore-snapshot-tier`

以下代码示例演示了如何使用 `restore-snapshot-tier`。

Amazon CLI

示例 1：永久还原已归档的快照

以下 `restore-snapshot-tier` 示例永久还原指定的快照。指定 `--snapshot-id` 并包括 `permanent-restore` 选项。

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-01234567890abcedf \
    --permanent-restore
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "IsPermanentRestore": true
}
```

有关快照归档的更多信息，请参阅《Amazon EBS User Guide》中的 Archive Amazon EBS snapshots。

示例 2：临时还原已归档的快照

以下 `restore-snapshot-tier` 示例临时还原指定的快照。忽略 `--permanent-restore` 选项。指定 `--snapshot-id`，对于 `temporary-restore-days`，指定还原快照的天数。`temporary-restore-days` 必须以天为单位指定。允许的范围是 1 到 180。如果没有指定一个值，默认为 1 天。

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-01234567890abcedf \
    --temporary-restore-days 5
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "RestoreDuration": 5,
    "IsPermanentRestore": false
}
```

有关快照归档的更多信息，请参阅《Amazon EBS User Guide》中的 Archive Amazon EBS snapshots。

示例 3：修改还原期

以下 `restore-snapshot-tier` 示例将指定快照的还原期更改为 10 天。

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-01234567890abcedf
    --temporary-restore-days 10
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "RestoreDuration": 10,
    "IsPermanentRestore": false
}
```

有关快照归档的更多信息，请参阅《Amazon EBS User Guide》中的 Archive Amazon EBS snapshots。

示例 4：修改还原类型

以下 `restore-snapshot-tier` 示例将指定快照的还原类型从临时更改为永久。

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-01234567890abcedf
    --permanent-restore
```

输出：

```
{
    "SnapshotId": "snap-01234567890abcedf",
```

```
    "IsPermanentRestore": true
}
```

有关快照归档的更多信息，请参阅《Amazon EBS User Guide》中的 Archive Amazon EBS snapshots。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreSnapshotTier。

`revoke-client-vpn-ingress`

以下代码示例演示了如何使用 `revoke-client-vpn-ingress`。

Amazon CLI

撤销 Client VPN 端点的授权规则

以下 `revoke-client-vpn-ingress` 示例撤销所有组的互联网访问权限 (`0.0.0.0/0`) 规则。

```
aws ec2 revoke-client-vpn-ingress \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
    --target-network-cidr 0.0.0.0/0 --revoke-all-groups
```

输出：

```
{
    "Status": {
        "Code": "revoking"
    }
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的授权规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeClientVpnIngress。

`revoke-security-group-egress`

以下代码示例演示了如何使用 `revoke-security-group-egress`。

Amazon CLI

示例 1：删除允许指向特定地址范围的出站流量的规则

以下 revoke-security-group-egress 示例命令删除授予对 TCP 端口 80 上指定地址范围的访问权限的规则。

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-
permissions [{IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组。

示例 2：删除允许指向特定安全组的出站流量的规则

以下 revoke-security-group-egress 示例命令删除授予对 TCP 端口 80 上指定安全组的访问权限的规则。

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort":
 443,"UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeSecurityGroupEgress。

revoke-security-group-ingress

以下代码示例演示了如何使用 revoke-security-group-ingress。

Amazon CLI

示例 1：从安全组中删除规则

以下 revoke-security-group-ingress 示例从默认 VPC 的指定安全组中删除 203.0.113.0/24 地址范围的 TCP 端口 22 访问权限。

```
aws ec2 revoke-security-group-ingress \
    --group-name mySecurityGroup
```

```
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组。

示例 2：使用 IP 权限集删除规则

以下 `revoke-security-group-ingress` 示例使用 `ip-permissions` 参数删除允许 ICMP 消息 `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set`（类型 3，代码 4）的入站规则。

```
aws ec2 revoke-security-group-ingress \
    --group-id sg-026c12253ce15eff7 \
    --ip-
permissions IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeSecurityGroupIngress。

`run-instances`

以下代码示例演示了如何使用 `run-instances`。

Amazon CLI

示例 1：将实例启动到默认子网

以下 `run-instances` 示例将类型 `t2.micro` 的单个实例启动到当前区域的默认子网中，并将其与该区域默认 VPC 的默认子网相关联。如果不打算使用 SSH（Linux）或 RDP（Windows）连接到实例，则密钥对是可选的。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --key-name MyKeyPair
```

输出：


```
{
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0abcdef1234567890",
            "InstanceId": "i-1231231230abcdef0",
            "InstanceType": "t2.micro",
            "KeyName": "MyKeyPair",
            "LaunchTime": "2018-05-10T08:05:20.000Z",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-2a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
            "PrivateIpAddress": "10.0.0.157",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-04a636d18e83cfacb",
            "VpcId": "vpc-1234567890abcdef0",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "",
            "EbsOptimized": false,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2018-05-10T08:05:20.000Z",
                        "AttachmentId": "eni-attach-0e325c07e928a0405",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching"
                    },
```

```
                    "Description": "",
                    "Groups": [
                        {
                            "GroupName": "MySecurityGroup",
                            "GroupId": "sg-0598c7d356eba48d7"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:ab:58:e0:67:e2",
                    "NetworkInterfaceId": "eni-0c0a29997760baee7",
                    "OwnerId": "123456789012",
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10.0.0.157",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                            "PrivateIpAddress": "10.0.0.157"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "InterfaceType": "interface"
                }
            ],
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SecurityGroups": [
                {
                    "GroupName": "MySecurityGroup",
                    "GroupId": "sg-0598c7d356eba48d7"
                }
            ],
            "SourceDestCheck": true,
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
            },
            "Tags": [],
            "VirtualizationType": "hvm",
            "CpuOptions": {
```

```
                "CoreCount": 1,
                "ThreadsPerCore": 1
            },
            "CapacityReservationSpecification": {
                "CapacityReservationPreference": "open"
            },
            "MetadataOptions": {
                "State": "pending",
                "HttpTokens": "optional",
                "HttpPutResponseHopLimit": 1,
                "HttpEndpoint": "enabled"
            }
        }
    ],
    "OwnerId": "123456789012",
    "ReservationId": "r-02a3f596d91211712"
}
```

示例 2：将实例启动到非默认子网，并添加一个公有 IP 地址

以下 run-instances 示例为要启动到非默认子网的实例请求一个公有 IP 地址。实例与指定的安全组相关联。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --associate-public-ip-address \
    --key-name MyKeyPair
```

有关 run-instances 的输出示例，请参阅示例 1。

示例 3：启动具有附加卷的实例

以下 run-instances 示例使用 mapping.json 中指定的块设备映射，在启动时对附加卷进行附加。块设备映射可以指定 EBS 卷、实例存储卷，也可以同时指定 EBS 卷和实例存储卷。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
```

```
    --key-name MyKeyPair \
    --block-device-mappings file://mapping.json
```

mapping.json 的内容。本示例添加了 /dev/sdh，这是一个大小为 100GiB 的空 EBS 卷。

```
[
    {
        "DeviceName": "/dev/sdh",
        "Ebs": {
            "VolumeSize": 100
        }
    }
]
```

mapping.json 的内容。本示例添加了 ephemeral1，作为实例存储卷。

```
[
    {
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral1"
    }
]
```

有关 run-instances 的输出示例，请参阅示例 1。

有关块设备映射的更多信息，请参阅《Amazon EC2 用户指南》中的块设备映射。

示例 4：启动实例并在创建时添加标签

以下 run-instances 示例向实例中添加了一个键为 webserver、值为 production 的标签。该命令还向创建的任何 EBS 卷 (此示例中为根卷) 应用键为 cost-center、值为 cc123 的标签。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --tag-specifications
 'ResourceType=instance,Tags=[{Key=webserver,Value=production}]'
 'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

有关 run-instances 的输出示例，请参阅示例 1。

示例 5：启动包含用户数据的实例

以下 run-instances 示例在名为 my_script.txt 的文件中传递用户数据，该文件包含实例的配置脚本。该脚本在启动时运行。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --user-data file://my_script.txt
```

有关 run-instances 的输出示例，请参阅示例 1。

有关实例用户数据的更多信息，请参阅《Amazon EC2 用户指南》中的使用实例用户数据。

示例 6：启动可突增性能实例

以下 run-instances 示例启动带有 unlimited 服务抵扣金选项的 t2.micro 实例。当启动 T2 实例时，如果未指定 --credit-specification，则默认为 standard 服务抵扣金选项。当启动 T3 实例时，默认为 unlimited 服务抵扣金选项。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --credit-specification CpuCredits=unlimited
```

有关 run-instances 的输出示例，请参阅示例 1。

有关可突增性能实例的更多信息，请参阅《Amazon EC2 用户指南》中的可突增性能实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RunInstances。

run-scheduled-instances

以下代码示例演示了如何使用 run-scheduled-instances。

Amazon CLI

启动计划实例

此示例在 VPC 中启动指定的计划实例。

命令:

```
aws ec2 run-scheduled-instances --scheduled-instance-
id sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-
specification file://launch-specification.json
```

Launch-specification.json：

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "InstanceType": "c4.large",
  "NetworkInterfaces": [
    {
        "DeviceIndex": 0,
        "SubnetId": "subnet-12345678",
        "AssociatePublicIpAddress": true,
        "Groups": ["sg-12345678"]
    }
  ],
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

输出：

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

此示例在 EC2-Classic 中启动指定的计划实例。

命令:

```
aws ec2 run-scheduled-instances --scheduled-instance-
id sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-
specification file://launch-specification.json
```

Launch-specification.json：

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
  "InstanceType": "c4.large",
  "Placement": {
    "AvailabilityZone": "us-west-2b"
  }
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

输出：

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RunScheduledInstances。

search-local-gateway-routes

以下代码示例演示了如何使用 search-local-gateway-routes。

Amazon CLI

在本地网关路由表中搜索路由

以下 search-local-gateway-routes 示例在指定的本地网关路由表中搜索静态路由。

```
aws ec2 search-local-gateway-routes \
    --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE \
    --filters "Name=type,Values=static"
```

输出：

```
{
    "Route": {
        "DestinationCidrBlock": "0.0.0.0/0",
        "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
        "Type": "static",
        "State": "deleted",
        "LocalGatewayRouteTableId": "lgw-rtb-059615ef7EXAMPLE"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchLocalGatewayRoutes。

search-transit-gateway-multicast-groups

以下代码示例演示了如何使用 search-transit-gateway-multicast-groups。

Amazon CLI

搜索一个或多个中转网关多播组并返回组成员资格信息

以下 search-transit-gateway-multicast-groups 示例返回指定多播组的组成员资格。

```
aws ec2 search-transit-gateway-multicast-groups \
    --transit-gateway-multicast-domain-id tgw-mcast-domain-000fb24d04EXAMPLE
```

输出：

```
{
    "MulticastGroups": [
        {
            "GroupIpAddress": "224.0.1.0",
            "TransitGatewayAttachmentId": "tgw-attach-0372e72386EXAMPLE",
            "SubnetId": "subnet-0187aff814EXAMPLE",
            "ResourceId": "vpc-0065acced4EXAMPLE",
            "ResourceType": "vpc",
            "NetworkInterfaceId": "eni-03847706f6EXAMPLE",
```

```
            "GroupMember": false,
            "GroupSource": true,
            "SourceType": "static"
        }
    ]
}
```

有关更多信息，请参阅《Transit Gateways Guide》中的 Multicast on transit gateways。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchTransitGatewayMulticastGroups。

search-transit-gateway-routes

以下代码示例演示了如何使用 search-transit-gateway-routes。

Amazon CLI

在指定的中转网关路由表中搜索路由

以下 search-transit-gateway-routes 示例返回指定路由表中 static 类型的所有路由。

```
aws ec2 search-transit-gateway-routes \
    --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE \
    --filters "Name=type,Values=static"
```

输出：

```
{
    "Routes": [
        {
            "DestinationCidrBlock": "10.0.2.0/24",
            "TransitGatewayAttachments": [
                {
                    "ResourceId": "vpc-4EXAMPLE",
                    "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
                    "ResourceType": "vpc"
                }
            ],
            "Type": "static",
            "State": "active"
        },
        {
```

```
            "DestinationCidrBlock": "10.1.0.0/24",
            "TransitGatewayAttachments": [
                {
                    "ResourceId": "vpc-4EXAMPLE",
                    "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
                    "ResourceType": "vpc"
                }
            ],
            "Type": "static",
            "State": "active"
        }
    ],
    "AdditionalRoutesAvailable": false
}
```

有关更多信息，请参阅《中转网关指南》中的中转网关路由表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchTransitGatewayRoutes。

send-diagnostic-interrupt

以下代码示例演示了如何使用 send-diagnostic-interrupt。

Amazon CLI

发送诊断中断

以下 send-diagnostic-interrupt 示例向指定实例发送诊断中断。

```
aws ec2 send-diagnostic-interrupt \
    --instance-id i-1234567890abcdef0
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendDiagnosticInterrupt。

start-instances

以下代码示例演示了如何使用 start-instances。

Amazon CLI

启动 Amazon EC2 实例

本示例启动指定的受 Amazon EBS 支持的实例。

命令:

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "StartingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的“停止和启动实例”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartInstances。

start-network-insights-access-scope-analysis

以下代码示例演示了如何使用 start-network-insights-access-scope-analysis。

Amazon CLI

启动 Network Insights 访问范围分析

以下 start-network-insights-access-scope-analysis 示例在您的 Amazon 账户中启动范围分析。

```
aws ec2 start-network-insights-access-scope-analysis \
    --region us-east-1 \
    --network-insights-access-scope-id nis-123456789111
```

输出：

```
{
    "NetworkInsightsAccessScopeAnalysis": {
        "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
        "NetworkInsightsAccessScopeAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope-analysis/nisa-123456789222",
        "NetworkInsightsAccessScopeId": "nis-123456789111",
        "Status": "running",
        "StartDate": "2022-01-26T00:47:06.814000+00:00"
    }
}
```

有关更多信息，请参阅《网络访问分析器指南》中的使用 Amazon CLI 的网络访问分析器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartNetworkInsightsAccessScopeAnalysis。

start-network-insights-analysis

以下代码示例演示了如何使用 start-network-insights-analysis。

Amazon CLI

分析路径

以下 start-network-insights-analysis 示例分析源和目标之间的路径。要查看路径分析的结果，请使用 describe-network-insights-analyses 命令。

```
aws ec2 start-network-insights-analysis \
    --network-insights-path-id nip-0b26f224f1d131fa8
```

输出：

```
{
    "NetworkInsightsAnalysis": {
        "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a",
        "NetworkInsightsAnalysisArn": "arn:aws:ec2:us-east-1:123456789012:network-
insights-analysis/nia-02207aa13eb480c7a",
        "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
        "StartDate": "2021-01-20T22:58:37.495Z",
```

```
        "Status": "running"
    }
}
```

有关更多信息，请参阅《Reachability Analyzer 指南》中的使用 Amazon CLI 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartNetworkInsightsAnalysis。

`start-vpc-endpoint-service-private-dns-verification`

以下代码示例演示了如何使用 `start-vpc-endpoint-service-private-dns-verification`。

Amazon CLI

启动 DNS 验证流程

以下 `start-vpc-endpoint-service-private-dns-verification` 示例启动指定端点服务的 DNS 验证流程。

```
aws ec2 start-vpc-endpoint-service-private-dns-verification \
    --service-id vpce-svc-071afff70666e61e0
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon PrivateLink 用户指南》中的管理 DNS 名称。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartVpcEndpointServicePrivateDnsVerification。

`stop-instances`

以下代码示例演示了如何使用 `stop-instances`。

Amazon CLI

示例 1：停止 Amazon EC2 实例

以下 `stop-instances` 示例将停止指定的受 Amazon EBS 支持的实例。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "StoppingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的停止和启动实例。

示例 2：使 Amazon EC2 实例进入休眠状态

以下 `stop-instances` 示例将让受 Amazon EBS 支持的实例进入休眠，前提是实例已启用休眠并且满足休眠先决条件。实例进入休眠状态后，实例将停止。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --hibernate
```

输出：

```
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-1234567890abcdef0",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
```

```
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic Cloud Compute 用户指南》中的休眠按需 Linux 实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopInstances。

`terminate-client-vpn-connections`

以下代码示例演示了如何使用 `terminate-client-vpn-connections`。

Amazon CLI

终止到 Client VPN 端点的连接

以下 `terminate-client-vpn-connections` 示例终止到 Client VPN 端点的指定连接。

```
aws ec2 terminate-client-vpn-connections \
    --client-vpn-endpoint-id vpn-endpoint-123456789123abcde \
    --connection-id cvpn-connection-04edd76f5201e0cb8
```

输出：

```
{
    "ClientVpnEndpointId": "vpn-endpoint-123456789123abcde",
    "ConnectionStatuses": [
        {
            "ConnectionId": "cvpn-connection-04edd76f5201e0cb8",
            "PreviousStatus": {
                "Code": "active"
            },
            "CurrentStatus": {
                "Code": "terminating"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Client VPN 管理员指南》中的客户端连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateClientVpnConnections。

terminate-instances

以下代码示例演示了如何使用 terminate-instances。

Amazon CLI

终止 Amazon EC2 实例

本示例将终止指定的实例。

命令:

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

输出：

```
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon 命令行界面用户指南》中的“使用 Amazon EC2 实例”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateInstances。

unassign-ipv6-addresses

以下代码示例演示了如何使用 unassign-ipv6-addresses。

Amazon CLI

取消分配给网络接口的 IPv6 地址

此示例取消分配给指定网络接口的指定 IPv6 地址。

命令:

```
aws ec2 unassign-ipv6-addresses --ipv6-
addresses 2001:db8:1234:1a00:3304:8879:34cf:4071 --network-interface-id eni-23c49b68
```

输出：

```
{
  "NetworkInterfaceId": "eni-23c49b68",
  "UnassignedIpv6Addresses": [
      "2001:db8:1234:1a00:3304:8879:34cf:4071"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnassignIpv6Addresses。

unassign-private-ip-addresses

以下代码示例演示了如何使用 unassign-private-ip-addresses。

Amazon CLI

取消分配给网络接口的辅助私有 IP 地址

此示例取消分配给指定网络接口的指定私有 IP 地址。如果命令成功，则不返回任何输出。

命令:

```
aws ec2 unassign-private-ip-addresses --network-interface-id eni-e5aa89a3 --private-
ip-addresses 10.0.0.82
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnassignPrivateIpAddresses。

unassign-private-nat-gateway-address

以下代码示例演示了如何使用 unassign-private-nat-gateway-address。

Amazon CLI

取消分配给私有 NAT 网关的私有 IP 地址

以下 unassign-private-nat-gateway-address 示例取消分配给指定私有 NAT 网关的指定 IP 地址。

```
aws ec2 unassign-private-nat-gateway-address \
    --nat-gateway-id nat-1234567890abcdef0 \
    --private-ip-addresses 10.0.20.197
```

输出：

```
{
    "NatGatewayId": "nat-0ee3edd182361f662",
    "NatGatewayAddresses": [
        {
            "NetworkInterfaceId": "eni-0065a61b324d1897a",
            "PrivateIp": "10.0.20.197",
            "IsPrimary": false,
            "Status": "unassigning"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC 用户指南》中的 NAT 网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnassignPrivateNatGatewayAddress。

unlock-snapshot

以下代码示例演示了如何使用 unlock-snapshot。

Amazon CLI

解锁快照

以下 unlock-snapshot 示例解锁指定的快照。

```
aws ec2 unlock-snapshot \
    --snapshot-id snap-0b5e733b4a8df6e0d
```

输出：

```
{
```

```
    "SnapshotId": "snap-0b5e733b4a8df6e0d"
}
```

有关更多信息，请参阅《Amazon EBS 用户指南》中的快照锁定。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnlockSnapshot。

unmonitor-instances

以下代码示例演示了如何使用 unmonitor-instances。

Amazon CLI

禁用对实例的详细监控

本示例命令禁用对指定实例的详细监控。

命令:

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

输出：

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "disabling"
          }
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnmonitorInstances。

update-security-group-rule-descriptions-egress

以下代码示例演示了如何使用 update-security-group-rule-descriptions-egress。

Amazon CLI

更新出站安全组规则的描述

以下 update-security-group-rule-descriptions-egress 示例更新指定端口和 IPv4 地址范围的安全组规则的描述。描述“Outbound HTTP access to server 2”取代该规则的任何现有描述。

```
aws ec2 update-security-group-rule-descriptions-egress \
    --group-id sg-02f0d35a850ba727f \
    --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=203.0.113.0/24,Description="Outbound
 HTTP access to server 2"}]
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecurityGroupRuleDescriptionsEgress。

update-security-group-rule-descriptions-ingress

以下代码示例演示了如何使用 update-security-group-rule-descriptions-ingress。

Amazon CLI

示例 1：使用 CIDR 源更新入站安全组规则的描述

以下 update-security-group-rule-descriptions-ingress 示例更新指定端口和 IPv4 地址范围的安全组规则的描述。描述“SSH access from ABC office”取代该规则的任何现有描述。

```
aws ec2 update-security-group-rule-descriptions-ingress \
    --group-id sg-02f0d35a850ba727f \
    --ip-permissions
 IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges='[{CidrIp=203.0.113.0/16,Description="SSH
 access from corpnet"}]'
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组规则。

示例 2：使用前缀列表源更新入站安全组规则的描述

以下 update-security-group-rule-descriptions-ingress 示例更新指定端口和前缀列表的安全组规则的描述。描述“SSH access from ABC office”取代该规则的任何现有描述。

```
aws ec2 update-security-group-rule-descriptions-ingress \
    --group-id sg-02f0d35a850ba727f \
    --ip-permissions
 IpProtocol=tcp,FromPort=22,ToPort=22,PrefixListIds='[{PrefixListId=pl-12345678,Description=
 access from corpnet"}]'
```

输出：

```
{
    "Return": true
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecurityGroupRuleDescriptionsIngress。

withdraw-byoip-cidr

以下代码示例演示了如何使用 withdraw-byoip-cidr。

Amazon CLI

停止公告地址范围

以下 withdraw-byoip-cidr 示例停止公告指定的地址范围。

```
aws ec2 withdraw-byoip-cidr
    --cidr 203.0.113.25/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "203.0.113.25/24",
        "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",
        "State": "advertised"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 WithdrawByoipCidr。

使用 Amazon CLI 的 Amazon EC2 Instance Connect 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon EC2 Instance Connect 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`send-ssh-public-key`

以下代码示例演示了如何使用 `send-ssh-public-key`。

Amazon CLI

将 SSH 公有密钥发送到实例

以下 `send-ssh-public-key` 示例将指定的 SSH 公有密钥发送到指定实例。该密钥用于对指定用户进行身份验证。

```
aws ec2-instance-connect send-ssh-public-key \
    --instance-id i-1234567890abcdef0 \
    --instance-os-user ec2-user \
    --availability-zone us-east-2b \
```

```
    --ssh-public-key file://path/my-rsa-key.pub
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendSshPublicKey。

使用 Amazon CLI 的 Amazon ECR 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon ECR 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-check-layer-availability`

以下代码示例演示了如何使用 `batch-check-layer-availability`。

Amazon CLI

检查层的可用性

以下 `batch-check-layer-availability` 示例检查 `cluster-autoscaler` 存储库中具有摘要 `sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed` 的层的可用性。

```
aws ecr batch-check-layer-availability \
    --repository-name cluster-autoscaler \
    --layer-
digests sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
```

输出：

```
{
    "layers": [
        {
            "layerDigest":
 "sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed",
            "layerAvailability": "AVAILABLE",
            "layerSize": 2777,
            "mediaType": "application/vnd.docker.container.image.v1+json"
        }
    ],
    "failures": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchCheckLayerAvailability。

batch-delete-image

以下代码示例演示了如何使用 batch-delete-image。

Amazon CLI

示例 1：删除单个映像

以下 batch-delete-image 示例删除账户默认注册表中指定存储库中带有标签 precise 的映像。

```
aws ecr batch-delete-image \
    --repository-name ubuntu \
    --image-ids imageTag=precise
```

输出：

```
{
    "failures": [],
    "imageIds": [
        {
            "imageTag": "precise",
            "imageDigest":
 "sha256:19665f1e6d1e504117a1743c0a3d3753086354a38375961f2e665416ef4b1b2f"
        }
    ]
```

```
}
```

示例 2：删除多个映像

以下 batch-delete-image 示例删除指定存储库中标记为 prod 和 team1 的所有映像。

```
aws ecr batch-delete-image \
    --repository-name MyRepository \
    --image-ids imageTag=prod imageTag=team1
```

输出：

```
{
    "imageIds": [
        {
            "imageDigest": "sha256:123456789012",
            "imageTag": "prod"
        },
        {
            "imageDigest": "sha256:567890121234",
            "imageTag": "team1"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的删除映像。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteImage。

batch-get-image

以下代码示例演示了如何使用 batch-get-image。

Amazon CLI

示例 1：获取单个映像

以下 batch-get-image 示例在账户的默认注册表中名为 cluster-autoscaler 的存储库中获取带有标签 v1.13.6 的映像。

```
aws ecr batch-get-image \
    --repository-name cluster-autoscaler \
```

```
    --image-ids imageTag=v1.13.6
```

输出：

```
{
    "images": [
        {
            "registryId": "012345678910",
            "repositoryName": "cluster-autoscaler",
            "imageId": {
                "imageDigest":
 "sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
                "imageTag": "v1.13.6"
            },
            "imageManifest": "{\n   \"schemaVersion\": 2,\n
 \"mediaType\": \"application/vnd.docker.distribution.manifest.v2+json
\",\n   \"config\": {\n      \"mediaType\": \"application/
vnd.docker.container.image.v1+json\",\n      \"size\": 2777,\n      \"digest
\": \"sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
\"\n   },\n   \"layers\": [\n      {\n         \"mediaType
\": \"application/vnd.docker.image.rootfs.diff.tar.gzip
\",\n         \"size\": 17743696,\n         \"digest\":
 \"sha256:39fafc05754f195f134ca11ecdb1c9a691ab0848c697fffeb5a85f900caaf6e1\"\n
      },\n      {\n         \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n         \"size\": 2565026,\n
 \"digest\":
 \"sha256:8c8a779d3a537b767ae1091fe6e00c2590afd16767aa6096d1b318d75494819f
\"\n      },\n      {\n         \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n         \"size\": 28005981,\n
 \"digest\":
 \"sha256:c44ba47496991c9982ee493b47fd25c252caabf2b4ae7dd679c9a27b6a3c8fb7\"\n
      },\n      {\n         \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n         \"size\": 775,\n         \"digest
\": \"sha256:e2c388b44226544363ca007be7b896bcce1baebea04da23cbd165eac30be650f\"\n
   }\n   ]\n}"
        }
    ],
    "failures": []
}
```

示例 2：获取多个映像

以下 batch-get-image 示例显示指定存储库中标记为 prod 和 team1 的所有映像的详细信息。

```
aws ecr batch-get-image \
    --repository-name MyRepository \
    --image-ids imageTag=prod imageTag=team1
```

输出：

```
{
    "images": [
        {
            "registryId": "123456789012",
            "repositoryName": "MyRepository",
            "imageId": {
                "imageDigest": "sha256:123456789012",
                "imageTag": "prod"
            },
            "imageManifest": "manifestExample1"
        },
        {
            "registryId": "567890121234",
            "repositoryName": "MyRepository",
            "imageId": {
                "imageDigest": "sha256:123456789012",
                "imageTag": "team1"
            },
            "imageManifest": "manifestExample2"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetImage。

complete-layer-upload

以下代码示例演示了如何使用 complete-layer-upload。

Amazon CLI

完成映像层上传

以下 complete-layer-upload 示例完成到 layer-test 存储库的映像层上传。

```
aws ecr complete-layer-upload \
    --repository-name layer-test \
    --upload-id 6cb64b8a-9378-0e33-2ab1-b780fab8a9e9 \
    --layer-digests 6cb64b8a-9378-0e33-2ab1-
b780fab8a9e9:48074e6d3a68b39aad8ccc002cdad912d4148c0f92b3729323e
```

输出：

```
{
    "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9",
    "layerDigest":
 "sha256:9a77f85878aa1906f2020a0ecdf7a7e962d57e882250acd773383224b3fe9a02",
    "repositoryName": "layer-test",
    "registryId": "130757420319"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteLayerUpload。

create-repository

以下代码示例演示了如何使用 create-repository。

Amazon CLI

示例 1：创建存储库

以下 create-repository 示例将在账户默认注册表中的指定命名空间内创建存储库。

```
aws ecr create-repository \
    --repository-name project-a/sample-repo
```

输出：

```
{
    "repository": {
        "registryId": "123456789012",
        "repositoryName": "project-a/sample-repo",
        "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-a/
sample-repo"
    }
```

```
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的创建存储库。

示例 2：创建配置了映像标签不可变性的存储库

以下 create-repository 示例将在账户的默认注册表中创建已配置标签不可变性的存储库。

```
aws ecr create-repository \
    --repository-name project-a/sample-repo \
    --image-tag-mutability IMMUTABLE
```

输出：

```
{
    "repository": {
        "registryId": "123456789012",
        "repositoryName": "project-a/sample-repo",
        "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-a/
sample-repo",
        "imageTagMutability": "IMMUTABLE"
    }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像标签可变性。

示例 3：创建已配置扫描配置的存储库

以下 create-repository 示例将在账户的默认注册表中创建配置为在映像推送中执行漏洞扫描的存储库。

```
aws ecr create-repository \
    --repository-name project-a/sample-repo \
    --image-scanning-configuration scanOnPush=true
```

输出：

```
{
    "repository": {
        "registryId": "123456789012",
        "repositoryName": "project-a/sample-repo",
```

```
        "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-a/
sample-repo",
        "imageScanningConfiguration": {
            "scanOnPush": true
        }
    }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRepository。

delete-lifecycle-policy

以下代码示例演示了如何使用 delete-lifecycle-policy。

Amazon CLI

删除存储库的生命周期策略

以下 delete-lifecycle-policy 示例删除 hello-world 存储库的生命周期策略。

```
aws ecr delete-lifecycle-policy \
    --repository-name hello-world
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "hello-world",
    "lifecyclePolicyText": "{\"rules\":[{\"rulePriority\":1,\"description\":\"Remove
 untagged images.\",\"selection\":{\"tagStatus\":\"untagged\",\"countType\":
\"sinceImagePushed\",\"countUnit\":\"days\",\"countNumber\":10},\"action\":{\"type
\":\"expire\"}}]}",
    "lastEvaluatedAt": 0.0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLifecyclePolicy。

delete-repository-policy

以下代码示例演示了如何使用 delete-repository-policy。

Amazon CLI

删除存储库的存储库策略

以下 delete-repository-policy 示例删除 cluster-autoscaler 存储库的存储库策略。

```
aws ecr delete-repository-policy \
    --repository-name cluster-autoscaler
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "cluster-autoscaler",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
 \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :
 \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage
\", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepositoryPolicy。

delete-repository

以下代码示例演示了如何使用 delete-repository。

Amazon CLI

删除存储库（ ）

以下 delete-repository 示例命令将强制删除账户默认注册表中指定的存储库。如果存储库包含映像，则需要 --force 标志。

```
aws ecr delete-repository \
    --repository-name ubuntu \
    --force
```

输出：

```
{
    "repository": {
        "registryId": "123456789012",
```

```
        "repositoryName": "ubuntu",
        "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/ubuntu"
    }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的删除存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepository。

describe-image-scan-findings

以下代码示例演示了如何使用 describe-image-scan-findings。

Amazon CLI

描述映像的扫描结果

以下 describe-image-scan-findings 示例使用账户默认注册表中指定存储库中的映像摘要返回映像的映像扫描结果。

```
aws ecr describe-image-scan-findings \
    --repository-name sample-repo \
    --image-
id imageDigest=sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6
```

输出：

```
{
    "imageScanFindings": {
      "findings": [
          {
              "name": "CVE-2019-5188",
              "description": "A code execution vulnerability exists in the directory
 rehashing functionality of E2fsprogs e2fsck 1.45.4. A specially crafted ext4
 directory can cause an out-of-bounds write on the stack, resulting in code
 execution. An attacker can corrupt a partition to trigger this vulnerability.",
              "uri": "http://people.ubuntu.com/~ubuntu-security/cve/CVE-2019-5188",
              "severity": "MEDIUM",
              "attributes": [
                  {
                      "key": "package_version",
                      "value": "1.44.1-1ubuntu1.1"
                  },
```

```
                    {
                        "key": "package_name",
                        "value": "e2fsprogs"
                    },
                    {
                        "key": "CVSS2_VECTOR",
                        "value": "AV:L/AC:L/Au:N/C:P/I:P/A:P"
                    },
                    {
                        "key": "CVSS2_SCORE",
                        "value": "4.6"
                    }
                ]
            }
        ],
        "imageScanCompletedAt": 1579839105.0,
        "vulnerabilitySourceUpdatedAt": 1579811117.0,
        "findingSeverityCounts": {
            "MEDIUM": 1
        }
    },
    "registryId": "123456789012",
    "repositoryName": "sample-repo",
    "imageId": {
        "imageDigest":
 "sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6"
    },
    "imageScanStatus": {
        "status": "COMPLETE",
        "description": "The scan was completed successfully."
    }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImageScanFindings。

describe-images

以下代码示例演示了如何使用 describe-images。

Amazon CLI

描述存储库中的映像

以下 describe-images 示例显示具有标签 v1.13.6 的 cluster-autoscaler 存储库中某个映像的相关详细信息。

```
aws ecr describe-images \
    --repository-name cluster-autoscaler \
    --image-ids imageTag=v1.13.6
```

输出：

```
{
    "imageDetails": [
        {
            "registryId": "012345678910",
            "repositoryName": "cluster-autoscaler",
            "imageDigest":
 "sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
            "imageTags": [
                "v1.13.6"
            ],
            "imageSizeInBytes": 48318255,
            "imagePushedAt": 1565128275.0
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImages。

describe-repositories

以下代码示例演示了如何使用 describe-repositories。

Amazon CLI

描述注册表中的存储库

此示例将描述账户默认注册表中的存储库。

命令:

```
aws ecr describe-repositories
```

输出：

```
{
    "repositories": [
        {
            "registryId": "012345678910",
            "repositoryName": "ubuntu",
            "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/ubuntu"
        },
        {
            "registryId": "012345678910",
            "repositoryName": "test",
            "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/test"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRepositories。

get-authorization-token

以下代码示例演示了如何使用 get-authorization-token。

Amazon CLI

获取默认注册表的授权令牌

以下 get-authorization-token 示例命令可获取默认注册表的授权令牌。

```
aws ecr get-authorization-token
```

输出：

```
{
    "authorizationData": [
        {
            "authorizationToken": "QVdTOkN...",
            "expiresAt": 1448875853.241,
            "proxyEndpoint": "https://123456789012.dkr.ecr.us-west-2.amazonaws.com"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizationToken。

get-download-url-for-layer

以下代码示例演示了如何使用 get-download-url-for-layer。

Amazon CLI

获取图层的下载 URL

以下 get-download-url-for-layer 示例显示 cluster-autoscaler 存储库中具有摘要 sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed 的层的下载 URL。

```
aws ecr get-download-url-for-layer \
    --repository-name cluster-autoscaler  \
    --layer-
digest sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
```

输出：

```
{
    "downloadUrl": "https://prod-us-west-2-starport-layer-bucket.s3.us-
west-2.amazonaws.com/e501-012345678910-9cb60dc0-7284-5643-3987-
da6dac0465f0/04620aac-66a5-4167-8232-55ee7ef6d565?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Date=20190814T220617Z&X-Amz-SignedHeaders=host&X-Amz-Expires=3600&X-
Amz-Credential=AKIA32P3D2JDNMVAJLGF%2F20190814%2Fus-west-2%2Fs3%2Faws4_request&X-
Amz-Signature=9161345894947a1672467a0da7a1550f2f7157318312fe4941b59976239c3337",
    "layerDigest":
 "sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDownloadUrlForLayer。

get-lifecycle-policy-preview

以下代码示例演示了如何使用 get-lifecycle-policy-preview。

Amazon CLI

检索生命周期策略预览的详细信息

以下 get-lifecycle-policy-preview 示例检索账户的默认注册表中指定存储库的生命周期策略预览结果。

命令:

```
aws ecr get-lifecycle-policy-preview \
    --repository-name "project-a/amazon-ecs-sample"
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "project-a/amazon-ecs-sample",
    "lifecyclePolicyText": "{\n    \"rules\": [\n        {\n
 \"rulePriority\": 1,\n            \"description\": \"Expire images older than 14
 days\",\n            \"selection\": {\n                \"tagStatus\": \"untagged\",
\n                \"countType\": \"sinceImagePushed\",\n                \"countUnit
\": \"days\",\n                \"countNumber\": 14\n            },\n
 \"action\": {\n                \"type\": \"expire\"\n            }\n        }\n
  ]\n}\n",
    "status": "COMPLETE",
    "previewResults": [],
    "summary": {
        "expiringImageTotalCount": 0
    }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的生命周期策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLifecyclePolicyPreview。

get-lifecycle-policy

以下代码示例演示了如何使用 get-lifecycle-policy。

Amazon CLI

检索生命周期策略

以下 get-lifecycle-policy 示例显示账户的默认注册表中指定存储库生命周期策略的详细信息。

```
aws ecr get-lifecycle-policy \
    --repository-name "project-a/amazon-ecs-sample"
```

输出：

```
{
     "registryId": "123456789012",
     "repositoryName": "project-a/amazon-ecs-sample",
     "lifecyclePolicyText": "{\"rules\":[{\"rulePriority\":1,\"description\":
\"Expire images older than 14 days\",\"selection\":{\"tagStatus\":\"untagged\",
\"countType\":\"sinceImagePushed\",\"countUnit\":\"days\",\"countNumber\":14},
\"action\":{\"type\":\"expire\"}}]}",
     "lastEvaluatedAt": 1504295007.0
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的生命周期策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLifecyclePolicy。

get-login-password

以下代码示例演示了如何使用 get-login-password。

Amazon CLI

检索密码以向注册表进行身份验证

以下 get-login-password 显示一个密码，您可以将其与您选择的容器客户端一起使用，以对您的 IAM 主体有权访问的任何 Amazon ECR 注册表进行身份验证。

```
aws ecr get-login-password
```

输出：

```
<password>
```

要与 Docker CLI 一起使用，请将 get-login-password 命令的输出通过管道传输到 docker login 命令。在检索密码时，请确保您指定的区域与 Amazon ECR 注册表所在的区域相同。

```
aws ecr get-login-password \
    --region <region>
 \
| docker login \
```

```
    --username AWS \
    --password-stdin <aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的注册表身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoginPassword。

get-login

以下代码示例演示了如何使用 get-login。

Amazon CLI

将 Docker 登录命令检索到默认注册表

此示例打印一个命令，您可以使用该命令登录默认 Amazon ECR 注册表。

命令:

```
aws ecr get-login
```

输出：

```
docker login -u AWS -p <password> -e none https://
<aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

登录其他账户的注册表

此示例打印一个或多个命令，您可以使用这些命令登录与其他账户关联的 Amazon ECR 注册表。

命令:

```
aws ecr get-login --registry-ids 012345678910 023456789012
```

输出：

```
docker login -u <username> -p <token-1> -e none <endpoint-1>
docker login -u <username> -p <token-2> -e none <endpoint-2>
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLogin。

get-repository-policy

以下代码示例演示了如何使用 get-repository-policy。

Amazon CLI

检索存储库的存储库策略

以下 get-repository-policy 示例显示 cluster-autoscaler 存储库的存储库策略的相关详细信息。

```
aws ecr get-repository-policy \
    --repository-name cluster-autoscaler
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "cluster-autoscaler",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
 \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :
 \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage
\", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepositoryPolicy。

initiate-layer-upload

以下代码示例演示了如何使用 initiate-layer-upload。

Amazon CLI

启动映像层上传

以下 initiate-layer-upload 示例启动到 layer-test 存储库的映像层上传。

```
aws ecr initiate-layer-upload \
    --repository-name layer-test
```

输出：

```
{
    "partSize": 10485760,
    "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InitiateLayerUpload。

`list-images`

以下代码示例演示了如何使用 `list-images`。

Amazon CLI

列出存储库的映像

以下 `list-images` 示例将显示 `cluster-autoscaler` 存储库的映像列表。

```
aws ecr list-images \
    --repository-name cluster-autoscaler
```

输出：

```
{
    "imageIds": [
        {
            "imageDigest":
 "sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",
            "imageTag": "v1.13.8"
        },
        {
            "imageDigest":
 "sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",
            "imageTag": "v1.13.7"
        },
        {
            "imageDigest":
 "sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
            "imageTag": "v1.13.6"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImages。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出存储库的标签

以下 list-tags-for-resource 示例将显示与 hello-world 存储库关联的标签列表。

```
aws ecr list-tags-for-resource \
    --resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world
```

输出：

```
{
    "tags": [
        {
            "Key": "Stage",
            "Value": "Integ"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-image-scanning-configuration

以下代码示例演示了如何使用 put-image-scanning-configuration。

Amazon CLI

更新存储库的映像扫描配置

以下 put-image-scanning-configuration 示例将更新指定存储库的映像扫描配置。

```
aws ecr put-image-scanning-configuration \
    --repository-name sample-repo \
```

```
    --image-scanning-configuration scanOnPush=true
```

输出：

```
{
   "registryId": "012345678910",
   "repositoryName": "sample-repo",
   "imageScanningConfiguration": {
     "scanOnPush": true
   }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutImageScanningConfiguration。

put-image-tag-mutability

以下代码示例演示了如何使用 put-image-tag-mutability。

Amazon CLI

更新存储库的映像标签可变性设置

以下 put-image-tag-mutability 示例为标签不可变性配置了指定存储库。这样可以防止存储库中的所有映像标签都被覆盖。

```
aws ecr put-image-tag-mutability \
    --repository-name hello-repository \
    --image-tag-mutability IMMUTABLE
```

输出：

```
{
   "registryId": "012345678910",
   "repositoryName": "sample-repo",
   "imageTagMutability": "IMMUTABLE"
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像标签可变性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutImageTagMutability。

put-image

以下代码示例演示了如何使用 put-image。

Amazon CLI

使用映像清单重新为映像添加标签

以下 put-image 示例使用现有映像清单在 hello-world 存储库中创建新标签。

```
aws ecr put-image \
    --repository-name hello-world \
    --image-tag 2019.08 \
    --image-manifest file://hello-world.manifest.json
```

的内容hello-world.manifest.json：

```
{
    "schemaVersion": 2,
    "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
    "config": {
        "mediaType": "application/vnd.docker.container.image.v1+json",
        "size": 5695,
        "digest":
 "sha256:cea5fe7701b7db3dd1c372f3cea6f43cdda444fcc488f530829145e426d8b980"
    },
    "layers": [
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 39096921,
            "digest":
 "sha256:d8868e50ac4c7104d2200d42f432b661b2da8c1e417ccfae217e6a1e04bb9295"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 57938,
            "digest":
 "sha256:83251ac64627fc331584f6c498b3aba5badc01574e2c70b2499af3af16630eed"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 423,
            "digest":
 "sha256:589bba2f1b36ae56f0152c246e2541c5aa604b058febfcf2be32e9a304fec610"
```

```
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 680,
            "digest":
"sha256:d62ecaceda3964b735cdd2af613d6bb136a52c1da0838b2ff4b4dab4212bcb1c"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 162,
            "digest":
"sha256:6d93b41cfc6bf0d2522b7cf61588de4cd045065b36c52bd3aec2ba0622b2b22b"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 28268840,
            "digest":
"sha256:6986b4d4c07932c680b3587f2eac8b0e013568c003cc23b04044628a5c5e599f"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 35369152,
            "digest":
"sha256:8c5ec60f10102dc8da0649d866c7c2f706e459d0bdc25c83ad2de86f4996c276"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 155,
            "digest":
"sha256:cde50b1c594539c5f67cbede9aef95c9ae321ccfb857f7b251b45b84198adc85"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 28737,
            "digest":
"sha256:2e102807ab72a73fc9abf53e8c50e421bdc337a0a8afcb242176edeec65977e4"
        },
        {
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 190,
            "digest":
"sha256:fc379bbd5ed37808772bef016553a297356c59b8f134659e6ee4ecb563c2f5a7"
        },
        {
```

```
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 28748,
            "digest":
 "sha256:021db240dfccf5a1aff19507d17c0177e5888e518acf295b52204b1825e8b7ee"
        }
    ]
}
```

输出：

```
{
    "image": {
        "registryId": "130757420319",
        "repositoryName": "hello-world",
        "imageId": {
            "imageDigest":
 "sha256:8ece96b74f87652876199d83bd107d0435a196133af383ac54cb82b6cc5283ae",
            "imageTag": "2019.08"
        },
        "imageManifest": "{\n  \"schemaVersion\": 2,\n  \"mediaType
\": \"application/vnd.docker.distribution.manifest.v2+json
\",\n  \"config\": {\n    \"mediaType\": \"application/
vnd.docker.container.image.v1+json\",\n    \"size\": 5695,\n    \"digest\":
 \"sha256:cea5fe7701b7db3dd1c372f3cea6f43cdda444fcc488f530829145e426d8b980\"\n
  },\n  \"layers\": [\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 39096921,\n      \"digest
\": \"sha256:d8868e50ac4c7104d2200d42f432b661b2da8c1e417ccfae217e6a1e04bb9295\"\n
  },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 57938,\n      \"digest
\": \"sha256:83251ac64627fc331584f6c498b3aba5badc01574e2c70b2499af3af16630eed
\"\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 423,\n      \"digest\":
 \"sha256:589bba2f1b36ae56f0152c246e2541c5aa604b058febfcf2be32e9a304fec610\"\n    },
\n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 680,\n      \"digest\":
 \"sha256:d62ecaceda3964b735cdd2af613d6bb136a52c1da0838b2ff4b4dab4212bcb1c
\"\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 162,\n      \"digest
\": \"sha256:6d93b41cfc6bf0d2522b7cf61588de4cd045065b36c52bd3aec2ba0622b2b22b
\"\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 28268840,\n      \"digest
\": \"sha256:6986b4d4c07932c680b3587f2eac8b0e013568c003cc23b04044628a5c5e599f
\"\n    },\n    {\n      \"mediaType\": \"application/
```

```
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 35369152,\n      \"digest
\": \"sha256:8c5ec60f10102dc8da0649d866c7c2f706e459d0bdc25c83ad2de86f4996c276\"\n
  },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 155,\n      \"digest\":
 \"sha256:cde50b1c594539c5f67cbede9aef95c9ae321ccfb857f7b251b45b84198adc85\"\n    },
\n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 28737,\n      \"digest\":
 \"sha256:2e102807ab72a73fc9abf53e8c50e421bdc337a0a8afcb242176edeec65977e4\"\n    },
\n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 190,\n      \"digest\":
 \"sha256:fc379bbd5ed37808772bef016553a297356c59b8f134659e6ee4ecb563c2f5a7\"\n    },
\n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 28748,\n      \"digest\":
 \"sha256:021db240dfccf5a1aff19507d17c0177e5888e518acf295b52204b1825e8b7ee\"\n
  }\n  ]\n}\n"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutImage。

put-lifecycle-policy

以下代码示例演示了如何使用 put-lifecycle-policy。

Amazon CLI

创建生命周期策略

以下 put-lifecycle-policy 示例为账户的默认注册表中的指定存储库创建生命周期策略。

```
aws ecr put-lifecycle-policy \
    --repository-name "project-a/amazon-ecs-sample" \
    --lifecycle-policy-text "file://policy.json"
```

的内容policy.json：

```
{
   "rules": [
       {
           "rulePriority": 1,
           "description": "Expire images older than 14 days",
           "selection": {
               "tagStatus": "untagged",
```

```
                "countType": "sinceImagePushed",
                "countUnit": "days",
                "countNumber": 14
            },
            "action": {
                "type": "expire"
            }
        }
    ]
}
```

输出：

```
{
    "registryId": "<aws_account_id>",
    "repositoryName": "project-a/amazon-ecs-sample",
    "lifecyclePolicyText": "{\"rules\":[{\"rulePriority\":1,\"description\":\"Expire
 images older than 14 days\",\"selection\":{\"tagStatus\":\"untagged\",\"countType
\":\"sinceImagePushed\",\"countUnit\":\"days\",\"countNumber\":14},\"action\":
{\"type\":\"expire\"}}]}"
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的生命周期策略。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutLifeCyclePolicy。

set-repository-policy

以下代码示例演示了如何使用 set-repository-policy。

Amazon CLI

为存储库设置存储库策略

以下 set-repository-policy 示例将文件中包含的存储库策略附加到 cluster-autoscaler 存储库。

```
aws ecr set-repository-policy \
    --repository-name cluster-autoscaler \
    --policy-text file://my-policy.json
```

的内容my-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Sid" : "allow public pull",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer"
            ]
        }
    ]
}
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "cluster-autoscaler",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
 \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :
 \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage
\", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetRepositoryPolicy。

start-image-scan

以下代码示例演示了如何使用 start-image-scan。

Amazon CLI

启动映像漏洞扫描

以下 start-image-scan 示例为指定存储库中的映像摘要启动映像扫描，并启动由该映像摘要指定的映像扫描。

```
aws ecr start-image-scan \
    --repository-name sample-repo \
```

```
    --image-
id imageDigest=sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6
```

输出：

```
{
   "registryId": "012345678910",
   "repositoryName": "sample-repo",
   "imageId": {
       "imageDigest":
 "sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6"
   },
   "imageScanStatus": {
       "status": "IN_PROGRESS"
   }
}
```

有关更多信息，请参阅《Amazon ECR 用户指南》中的映像扫描。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartImageScan。

start-lifecycle-policy-preview

以下代码示例演示了如何使用 start-lifecycle-policy-preview。

Amazon CLI

创建生命周期策略预览

以下 start-lifecycle-policy-preview 示例创建了由指定存储库的 JSON 文件定义的生命周期策略预览。

```
aws ecr start-lifecycle-policy-preview \
    --repository-name "project-a/amazon-ecs-sample" \
    --lifecycle-policy-text "file://policy.json"
```

的内容policy.json：

```
{
   "rules": [
       {
           "rulePriority": 1,
```

```
            "description": "Expire images older than 14 days",
            "selection": {
                "tagStatus": "untagged",
                "countType": "sinceImagePushed",
                "countUnit": "days",
                "countNumber": 14
            },
            "action": {
                "type": "expire"
            }
        }
    ]
}
```

输出：

```
{
    "registryId": "012345678910",
    "repositoryName": "project-a/amazon-ecs-sample",
    "lifecyclePolicyText": "{\n    \"rules\": [\n        {\n
 \"rulePriority\": 1,\n            \"description\": \"Expire images older than 14
 days\",\n            \"selection\": {\n                \"tagStatus\": \"untagged\",
\n                \"countType\": \"sinceImagePushed\",\n                \"countUnit
\": \"days\",\n                \"countNumber\": 14\n            },\n
 \"action\": {\n                \"type\": \"expire\"\n            }\n        }\n
  ]\n}\n",
    "status": "IN_PROGRESS"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartLifecyclePolicyPreview。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记存储库

以下 tag-resource 示例在 hello-world 存储库上设置键为 Stage 且值为 Integ 的标签。

```
aws ecr tag-resource \
    --resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world \
```

```
    --tags Key=Stage,Value=Integ
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

取消标记存储库

以下 untag-resource 示例从 hello-world 存储库中删除键为 Stage 的标签。

```
aws ecr untag-resource \
    --resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world \
    --tag-keys Stage
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

upload-layer-part

以下代码示例演示了如何使用 upload-layer-part。

Amazon CLI

上传图层部分

以下 upload-layer-part 将映像层部分上传到 layer-test 存储库。

```
aws ecr upload-layer-part \
    --repository-name layer-test \
    --upload-id 6cb64b8a-9378-0e33-2ab1-b780fab8a9e9 \
    --part-first-byte 0 \
    --part-last-byte 8323314 \
    --layer-part-blob file:///var/lib/docker/image/overlay2/layerdb/sha256/
ff986b10a018b48074e6d3a68b39aad8ccc002cdad912d4148c0f92b3729323e/layer.b64
```

输出：

```
{
    "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9",
    "registryId": "012345678910",
    "lastByteReceived": 8323314,
    "repositoryName": "layer-test"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadLayerPart。

使用 Amazon CLI 的 Amazon ECR Public 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon ECR Public 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-delete-image`

以下代码示例演示了如何使用 `batch-delete-image`。

Amazon CLI

示例 1：要使用映像摘要 ID 来删除映像，需要从公共注册表中删除存储库中的映像及其所有标签

以下 `batch-delete-image` 示例通过指定映像摘要来删除映像。

```
aws ecr-public batch-delete-image \
    --repository-name project-a/nginx-web-app \
    --image-
ids imageDigest=sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2
```

输出：

```
{
"imageIds": [
    {
        "imageDigest":
 "sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2",
        "imageTag": "latest"
    }
],
"failures": []
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Deleting an image in a public repository。

示例 2：通过指定与要从存储库中删除的任何映像关联的标签来删除该映像。

以下 batch-delete-image 示例通过在公共注册表中指定与名为 project-a/nginx-web-app 的映像存储库关联的标签来删除映像。如果您只有一个标签并执行此命令，它将移除映像。否则，如果同一个映像有多个标签，请指定一个标签，并且只会从存储库而非映像中移除该标签。

```
aws ecr-public batch-delete-image \
    --repository-name project-a/nginx-web-app \
    --image-ids imageTag=_temp
```

输出：

```
{
    "imageIds": [
        {
            "imageDigest":
 "sha256:f7a86a0760e2f8d7eff07e515fc87bf4bac45c35376c06f9a280f15ecad6d7e0",
            "imageTag": "_temp"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Deleting an image in a public repository。

示例 3：要删除多个映像，您可以在公共注册表中的存储库请求中指定多个映像标签或映像摘要。

以下 batch-delete-image 示例通过在请求中指定多个映像标签或映像摘要，从名为 project-a/nginx-web-app 的存储库中删除多个映像。

```
aws ecr-public batch-delete-image \
    --repository-name project-a/nginx-web-app \
    --image-ids imageTag=temp2.0
  imageDigest=sha256:47ba980bc055353d9c0af89b1894f68faa43ca93856917b8406316be86f01278
```

输出：

```
{
     "imageIds": [
         {
             "imageDigest":
 "sha256:47ba980bc055353d9c0af89b1894f68faa43ca93856917b8406316be86f01278"
         },
         {
             "imageDigest":
 "sha256:f7a86a0760e2f8d7eff07e515fc87bf4bac45c35376c06f9a280f15ecad6d7e0",
             "imageTag": "temp2.0"
         }
     ],
     "failures": []
 }
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Deleting an image in a public repository。

示例 4：要使用 registry-id 和 imagedigest id 跨 Amazon 账户删除映像，需要从公共注册表中删除存储库中的映像及其所有标签

以下 batch-delete-image 示例通过跨 Amazon 账户指定映像摘要来删除映像。

```
aws ecr-public batch-delete-image \
    --registry-id 123456789098 \
    --repository-name project-a/nginx-web-app \
    --image-
ids imageDigest=sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2
 \
    --region us-east-1
```

输出：

```
{
    "imageIds": [
        {
            "imageDigest":
 "sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2",
            "imageTag": "temp2.0"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Deleting an image in a public repository。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteImage。

`create-repository`

以下代码示例演示了如何使用 `create-repository`。

Amazon CLI

示例 1：在公有注册表中创建存储库

以下 `create-repository` 示例在公共注册表中创建一个名为 project-a/nginx-web-app 的存储库。

```
aws ecr-public create-repository \
    --repository-name project-a/nginx-web-app
```

输出：

```
{
    "repository": {
        "repositoryArn": "arn:aws:ecr-public::123456789012:repository/project-a/
nginx-web-app",
        "registryId": "123456789012",
        "repositoryName": "project-a/nginx-web-app",
        "repositoryUri": "public.ecr.aws/public-registry-custom-alias/project-a/
nginx-web-app",
```

```
        "createdAt": "2024-07-01T21:08:55.131000+00:00"
    },
    "catalogData": {}
}
```

有关更多信息，请参阅《Amazon ECR Public 用户指南》中的创建公有存储库。

示例 2：在公有注册表中创建存储库，并简要描述存储库的内容、存储库中的映像与之兼容的系统和操作架构

以下 `create-repository` 示例在公共注册表中创建一个名为 project-a/nginx-web-app 的存储库，并简要描述存储库的内容、存储库中的映像与之兼容的系统和操作架构。

```
aws ecr-public create-repository \
    --repository-name project-a/nginx-web-app \
    --catalog-data 'description=My project-a ECR Public
 Repository,architectures=ARM,ARM 64,x86,x86-64,operatingSystems=Linux'
```

输出：

```
{
    "repository": {
        "repositoryArn": "arn:aws:ecr-public::123456789012:repository/project-a/
nginx-web-app",
        "registryId": "123456789012",
        "repositoryName": "project-a/nginx-web-app",
        "repositoryUri": "public.ecr.aws/public-registry-custom-alias/project-a/
nginx-web-app",
        "createdAt": "2024-07-01T21:23:20.455000+00:00"
    },
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ]
    }
```

```
}
```

有关更多信息，请参阅《Amazon ECR Public 用户指南》中的创建公有存储库。

示例 3：在公有注册表中创建存储库，以及 logoImageBlob、aboutText、usageText 和标签信息

以下 `create-repository` 示例在公有注册表中创建一个名为 project-a/nginx-web-app 的存储库，以及 logoImageBlob、aboutText、usageText 和标签信息。

```
aws ecr-public create-repository \
    --cli-input-json file://myfile.json
```

的内容`myfile.json`：

```
{
    "repositoryName": "project-a/nginx-web-app",
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ],
        "logoImageBlob": "iVBORw0KGgoA<<truncated-for-better-reading>>ErkJggg==",
        "aboutText": "## Quick reference\n\nMaintained by: [the Amazon Linux Team]
(https://github.com/aws/amazon-linux-docker-images)\n\nWhere to get help: [the
 Docker Community Forums](https://forums.docker.com/), [the Docker Community Slack]
(https://dockr.ly/slack), or [Stack Overflow](https://stackoverflow.com/search?
tab=newest&q=docker)\n\n## Supported tags and respective `dockerfile` links\n\n*
 [`2.0.20200722.0`, `2`, `latest`](https://github.com/amazonlinux/container-images/
blob/03d54f8c4d522bf712cffd6c8f9aafba0a875e78/Dockerfile)\n* [`2.0.20200722.0-
with-sources`, `2-with-sources`, `with-sources`](https://github.com/
amazonlinux/container-images/blob/1e7349845e029a2e6afe6dc473ef17d052e3546f/
Dockerfile)\n* [`2018.03.0.20200602.1`, `2018.03`, `1`](https://github.com/
amazonlinux/container-images/blob/f10932e08c75457eeb372bf1cc47ea2a4b8e98c8/
Dockerfile)\n* [`2018.03.0.20200602.1-with-sources`, `2018.03-with-sources`,
 `1-with-sources`](https://github.com/amazonlinux/container-images/
blob/8c9ee491689d901aa72719be0ec12087a5fa8faf/Dockerfile)\n\n## What is Amazon
 Linux?\n\nAmazon Linux is provided by Amazon Web Services (AWS). It is designed
```

```
 to provide a stable, secure, and high-performance execution environment for
 applications running on Amazon EC2. The full distribution includes packages that
 enable easy integration with AWS, including launch configuration tools and many
 popular AWS libraries and tools. AWS provides ongoing security and maintenance
 updates to all instances running Amazon Linux.\n\nThe Amazon Linux container image
 contains a minimal set of packages. To install additional packages, [use `yum`]
(https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/managing-software.html).\n\nAWS
 provides two versions of Amazon Linux: [Amazon Linux 2](https://aws.amazon.com/
amazon-linux-2/) and [Amazon Linux AMI](https://aws.amazon.com/amazon-linux-ami/).
\n\nFor information on security updates for Amazon Linux, please refer to [Amazon
 Linux 2 Security Advisories](https://alas.aws.amazon.com/alas2.html) and [Amazon
 Linux AMI Security Advisories](https://alas.aws.amazon.com/). Note that Docker
 Hub's vulnerability scanning for Amazon Linux is currently based on RPM versions,
 which does not reflect the state of backported patches for vulnerabilities.\n
\n## Where can I run Amazon Linux container images?\n\nYou can run Amazon Linux
 container images in any Docker based environment. Examples include, your laptop,
 in Amazon EC2 instances, and Amazon ECS clusters.\n\n## License\n\nAmazon Linux is
 available under the [GNU General Public License, version 2.0](https://github.com/
aws/amazon-linux-docker-images/blob/master/LICENSE). Individual software packages
 are available under their own licenses; run `rpm -qi [package name]` or check
 `/usr/share/doc/[package name]-*` and `/usr/share/licenses/[package name]-*` for
 details.\n\nAs with all Docker images, these likely also contain other software
 which may be under other licenses (such as Bash, etc from the base distribution,
 along with any direct or indirect dependencies of the primary software being
 contained).\n\nSome additional license information which was able to be auto-
detected might be found in [the `repo-info` repository's `amazonlinux/` directory]
(https://github.com/docker-library/repo-info/tree/master/repos/amazonlinux).\n\n##
 Security\n\nFor information on security updates for Amazon Linux, please refer
 to [Amazon Linux 2 Security Advisories](https://alas.aws.amazon.com/alas2.html)
 and [Amazon Linux AMI Security Advisories](https://alas.aws.amazon.com/). Note
 that Docker Hub's vulnerability scanning for Amazon Linux is currently based
 on RPM versions, which does not reflect the state of backported patches for
 vulnerabilities.",
       "usageText": "## Supported architectures\n\namd64, arm64v8\n\n## Where
 can I run Amazon Linux container images?\n\nYou can run Amazon Linux container
 images in any Docker based environment. Examples include, your laptop, in Amazon
 EC2 instances, and ECS clusters.\n\n## How do I install a software package from
 Extras repository in Amazon Linux 2?\n\nAvailable packages can be listed with the
 `amazon-linux-extras` command. Packages can be installed with the `amazon-linux-
extras install <package>` command. Example: `amazon-linux-extras install rust1`\n
\n## Will updates be available for Amazon Linux containers?\n\nSimilar to the Amazon
 Linux images for Amazon EC2 and on-premises use, Amazon Linux container images will
 get ongoing updates from Amazon in the form of security updates, bug fix updates,
 and other enhancements. Security bulletins for Amazon Linux are available at
```

```
 https://alas.aws.amazon.com/\n\n## Will AWS Support the current version of Amazon
 Linux going forward?\n\nYes; in order to avoid any disruption to your existing
 applications and to facilitate migration to Amazon Linux 2, AWS will provide
 regular security updates for Amazon Linux 2018.03 AMI and container image for 2
 years after the final LTS build is announced. You can also use all your existing
 support channels such as AWS Support and Amazon Linux Discussion Forum to continue
 to submit support requests."
    },
    "tags": [
        {
            "Key": "Name",
            "Value": "project-a/nginx-web-app"
        },
        {
            "Key": "Environment",
            "Value": "Prod"
        }
    ]
}
```

输出：

```
{
    "repository": {
        "repositoryArn": "arn:aws:ecr-public::123456789012:repository/project-a/
nginx-web-app",
        "registryId": "123456789012",
        "repositoryName": "project-a/nginx-web-app",
        "repositoryUri": "public.ecr.aws/public-registry-custom-alias/project-a/
nginx-web-app",
        "createdAt": "2024-07-01T21:53:05.749000+00:00"
    },
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ],
```

```
        "logoUrl": "https://d3g9o9u8re44ak.cloudfront.net/
logo/23861450-4b9b-403c-9a4c-7aa0ef140bb8/2f9bf5a7-a32f-45b4-b5cd-c5770a35e6d7.png",
        "aboutText": "## Quick reference\n\nMaintained by: [the Amazon Linux Team]
(https://github.com/aws/amazon-linux-docker-images)\n\nWhere to get help: [the
 Docker Community Forums](https://forums.docker.com/), [the Docker Community Slack]
(https://dockr.ly/slack), or [Stack Overflow](https://stackoverflow.com/search?
tab=newest&q=docker)\n\n## Supported tags and respective `dockerfile` links\n\n*
 [`2.0.20200722.0`, `2`, `latest`](https://github.com/amazonlinux/container-images/
blob/03d54f8c4d522bf712cffd6c8f9aafba0a875e78/Dockerfile)\n* [`2.0.20200722.0-
with-sources`, `2-with-sources`, `with-sources`](https://github.com/
amazonlinux/container-images/blob/1e7349845e029a2e6afe6dc473ef17d052e3546f/
Dockerfile)\n* [`2018.03.0.20200602.1`, `2018.03`, `1`](https://github.com/
amazonlinux/container-images/blob/f10932e08c75457eeb372bf1cc47ea2a4b8e98c8/
Dockerfile)\n* [`2018.03.0.20200602.1-with-sources`, `2018.03-with-sources`,
 `1-with-sources`](https://github.com/amazonlinux/container-images/
blob/8c9ee491689d901aa72719be0ec12087a5fa8faf/Dockerfile)\n\n## What is Amazon
 Linux?\n\nAmazon Linux is provided by Amazon Web Services (AWS). It is designed
 to provide a stable, secure, and high-performance execution environment for
 applications running on Amazon EC2. The full distribution includes packages that
 enable easy integration with AWS, including launch configuration tools and many
 popular AWS libraries and tools. AWS provides ongoing security and maintenance
 updates to all instances running Amazon Linux.\n\nThe Amazon Linux container image
 contains a minimal set of packages. To install additional packages, [use `yum`]
(https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/managing-software.html).\n\nAWS
 provides two versions of Amazon Linux: [Amazon Linux 2](https://aws.amazon.com/
amazon-linux-2/) and [Amazon Linux AMI](https://aws.amazon.com/amazon-linux-ami/).
\n\nFor information on security updates for Amazon Linux, please refer to [Amazon
 Linux 2 Security Advisories](https://alas.aws.amazon.com/alas2.html) and [Amazon
 Linux AMI Security Advisories](https://alas.aws.amazon.com/). Note that Docker
 Hub's vulnerability scanning for Amazon Linux is currently based on RPM versions,
 which does not reflect the state of backported patches for vulnerabilities.\n
\n## Where can I run Amazon Linux container images?\n\nYou can run Amazon Linux
 container images in any Docker based environment. Examples include, your laptop,
 in Amazon EC2 instances, and Amazon ECS clusters.\n\n## License\n\nAmazon Linux is
 available under the [GNU General Public License, version 2.0](https://github.com/
aws/amazon-linux-docker-images/blob/master/LICENSE). Individual software packages
 are available under their own licenses; run `rpm -qi [package name]` or check
 `/usr/share/doc/[package name]-*` and `/usr/share/licenses/[package name]-*` for
 details.\n\nAs with all Docker images, these likely also contain other software
 which may be under other licenses (such as Bash, etc from the base distribution,
 along with any direct or indirect dependencies of the primary software being
 contained).\n\nSome additional license information which was able to be auto-
detected might be found in [the `repo-info` repository's `amazonlinux/` directory]
(https://github.com/docker-library/repo-info/tree/master/repos/amazonlinux).\n\n##
```

```
 Security\n\nFor information on security updates for Amazon Linux, please refer
 to [Amazon Linux 2 Security Advisories](https://alas.aws.amazon.com/alas2.html)
 and [Amazon Linux AMI Security Advisories](https://alas.aws.amazon.com/). Note
 that Docker Hub's vulnerability scanning for Amazon Linux is currently based
 on RPM versions, which does not reflect the state of backported patches for
 vulnerabilities.",
        "usageText": "## Supported architectures\n\namd64, arm64v8\n\n## Where
 can I run Amazon Linux container images?\n\nYou can run Amazon Linux container
 images in any Docker based environment. Examples include, your laptop, in Amazon
 EC2 instances, and ECS clusters.\n\n## How do I install a software package from
 Extras repository in Amazon Linux 2?\n\nAvailable packages can be listed with the
 `amazon-linux-extras` command. Packages can be installed with the `amazon-linux-
extras install <package>` command. Example: `amazon-linux-extras install rust1`\n
\n## Will updates be available for Amazon Linux containers?\n\nSimilar to the Amazon
 Linux images for Amazon EC2 and on-premises use, Amazon Linux container images will
 get ongoing updates from Amazon in the form of security updates, bug fix updates,
 and other enhancements. Security bulletins for Amazon Linux are available at
 https://alas.aws.amazon.com/\n\n## Will AWS Support the current version of Amazon
 Linux going forward?\n\nYes; in order to avoid any disruption to your existing
 applications and to facilitate migration to Amazon Linux 2, AWS will provide
 regular security updates for Amazon Linux 2018.03 AMI and container image for 2
 years after the final LTS build is announced. You can also use all your existing
 support channels such as AWS Support and Amazon Linux Discussion Forum to continue
 to submit support requests."
    }
}
```

有关更多信息，请参阅《Amazon ECR Public 用户指南》中的创建公有存储库，以及《Amazon ECR Public 用户指南》中的存储库目录数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRepository。

delete-repository-policy

以下代码示例演示了如何使用 delete-repository-policy。

Amazon CLI

在公共注册表中删除存储库策略

以下 delete-repository-policy 示例从 Amazon 账户中删除 ECR Public 存储库的存储库策略。

```
aws ecr-public delete-repository-policy \
     --repository-name project-a/nginx-web-app \
     --region us-east-1
```

输出：

```
{
    "registryId": "123456789012",
    "repositoryName": "project-a/nginx-web-app",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
    \"Sid\" : \"AllowPush\",\n    \"Effect\" : \"Allow\",\n    \"Principal
\" : {\n      \"AWS\" : [ \"arn:aws:iam:"123456789012":user/eksuser1\",
 \"arn:aws:iam:"123456789012":user/admin\" ]\n    },\n    \"Action\" :
 [ \"ecr-public:BatchCheckLayerAvailability\", \"ecr-public:PutImage\",
 \"ecr-public:InitiateLayerUpload\", \"ecr-public:UploadLayerPart\", \"ecr-
public:CompleteLayerUpload\" ]\n  } ]\n}"
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Deleting a public repository policy statement。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepositoryPolicy。

delete-repository

以下代码示例演示了如何使用 delete-repository。

Amazon CLI

删除公有注册表中的存储库

以下 delete-repository 示例从公有注册表中删除名为 project-a/nginx-web-app 的存储库。

```
aws ecr-public delete-repository \
    --repository-name project-a/nginx-web-app
```

输出：

```
{
    "repository": {
```

```
        "repositoryArn": "arn:aws:ecr-public::123456789012:repository/project-a/
nginx-web-app",
        "registryId": "123456789012",
        "repositoryName": "project-a/nginx-web-app",
        "repositoryUri": "public.ecr.aws/public-registry-custom-alias/project-a/
nginx-web-app",
        "createdAt": "2024-07-01T22:14:50.103000+00:00"
    }
}
```

有关更多信息，请参阅 Amazon ECR Public 中的删除公共存储库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRepository。

describe-image-tags

以下代码示例演示了如何使用 describe-image-tags。

Amazon CLI

示例 1：描述公共存储库中的映像标签的详细信息

以下 describe-image-tags 示例描述 project-a/nginx-web-app 示例存储库中的 imagetags。

```
aws ecr-public describe-image-tags \
    --repository-name project-a/nginx-web-app \
    --region us-east-1
```

输出：

```
{
    "imageTagDetails": [
        {
            "imageTag": "latest",
            "createdAt": "2024-07-10T22:29:00-05:00",
            "imageDetail": {
                "imageDigest":
 "sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2",
                "imageSizeInBytes": 121956548,
                "imagePushedAt": "2024-07-10T22:29:00-05:00",
                "imageManifestMediaType": "application/
vnd.docker.distribution.manifest.v2+json",
```

```
                "artifactMediaType": "application/
vnd.docker.container.image.v1+json"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeImageTags。

`describe-images`

以下代码示例演示了如何使用 `describe-images`。

Amazon CLI

示例 1：描述公共注册表存储库中的映像

以下 `describe-images` 示例描述公共注册表中名为 `project-a/nginx-web-app` 的存储库中的 imagesDetails。

```
aws ecr-public describe-images \
    --repository-name project-a/nginx-web-app \
    --region us-east-1
```

输出：

```
{
    "imageDetails": [
        {
            "registryId": "123456789012",
            "repositoryName": "project-a/nginx-web-app",
            "imageDigest":
 "sha256:0d8c93e72e82fa070d49565c00af32abbe8ddfd7f75e39f4306771ae0628c7e8",
            "imageTags": [
                "temp1.0"
            ],
            "imageSizeInBytes": 123184716,
            "imagePushedAt": "2024-07-23T11:32:49-05:00",
            "imageManifestMediaType": "application/
vnd.docker.distribution.manifest.v2+json",
            "artifactMediaType": "application/vnd.docker.container.image.v1+json"
        },
```

```
        {
            "registryId": "123456789012",
            "repositoryName": "project-a/nginx-web-app",
            "imageDigest":
 "sha256:b1f9deb5fe3711a3278379ebbcaefbc5d70a2263135db86bd27a0dae150546c2",
            "imageTags": [
                "temp2.0"
            ],
            "imageSizeInBytes": 121956548,
            "imagePushedAt": "2024-07-23T11:39:38-05:00",
            "imageManifestMediaType": "application/
vnd.docker.distribution.manifest.v2+json",
            "artifactMediaType": "application/vnd.docker.container.image.v1+json"
        },
        {
            "registryId": "123456789012",
            "repositoryName": "project-a/nginx-web-app",
            "imageDigest":
 "sha256:f7a86a0760e2f8d7eff07e515fc87bf4bac45c35376c06f9a280f15ecad6d7e0",
            "imageTags": [
                "temp3.0",
                "latest"
            ],
            "imageSizeInBytes": 232108879,
            "imagePushedAt": "2024-07-22T00:54:34-05:00",
            "imageManifestMediaType": "application/
vnd.docker.distribution.manifest.v2+json",
            "artifactMediaType": "application/vnd.docker.container.image.v1+json"
        }
    ]
}
```

有关更多信息，请参阅 Amazon ECR Public 中的描述公共存储库中的映像。

示例 2：按排序 imageTags 和 imagePushedAt 描述存储库中的映像

以下 `describe-images` 示例描述公共注册表中名为 project-a/nginx-web-app 的存储库中的映像。

```
aws ecr-public describe-images \
    --repository-name project-a/nginx-web-app \
    --query 'sort_by(imageDetails,& imagePushedAt)[*].imageTags[*]' \
    --output text
```

输出：

```
temp3.0 latest
temp1.0
temp2.0
```

示例 3：描述存储库中的映像以生成推送到存储库中的最后 2 个映像标签

以下 describe-images 示例从公共注册表中名为 project-a/nginx-web-app 的存储库中获取 imagetags 详细信息，并查询结果以仅显示前两条记录。

```
aws ecr-public describe-images \
    --repository-name project-a/nginx-web-app  \
    --query 'sort_by(imageDetails,& imagePushedAt)[*].imageTags[*] | [0:2]' \
    --output text
```

输出：

```
temp3.0 latest
temp1.0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImages。

describe-registries

以下代码示例演示了如何使用 describe-registries。

Amazon CLI

描述公共注册表中的所有注册表项

以下 describe-registries 示例描述账户中的所有注册表项。

```
aws ecr-public describe-registries
```

输出：

```
{
 "registries": [
     {
         "registryId": "123456789012",
```

```
        "registryArn": "arn:aws:ecr-public::123456789012:registry/123456789012",
        "registryUri": "public.ecr.aws/publicregistrycustomalias",
        "verified": false,
        "aliases": [
            {
                "name": "publicregistrycustomalias",
                "status": "ACTIVE",
                "primaryRegistryAlias": true,
                "defaultRegistryAlias": true
            }
        ]
    }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeRegistries。

describe-repository

以下代码示例演示了如何使用 describe-repository。

Amazon CLI

示例 1：描述公共注册表中的存储库

以下 describe-repositories 示例描述公共注册表中名为 project-a/nginx-web-app 的存储库。

```
aws ecr-public describe-repositories \
    --repository-name project-a/nginx-web-app
```

输出：

```
{
    "repositories": [
        {
            "repositoryArn": "arn:aws:ecr-public::123456789012:repository/project-a/
nginx-web-app",
            "registryId": "123456789012",
            "repositoryName": "project-a/nginx-web-app",
            "repositoryUri": "public.ecr.aws/public-registry-custom-alias/project-a/
nginx-web-app",
```

```
            "createdAt": "2024-07-07T00:07:56.526000-05:00"
        }
    ]
}
```

示例 2：以表格式描述公共注册表中的所有存储库

以下 describe-repositories 示例描述公共注册表中的所有存储库，然后将存储库名称输出为表格式。

```
aws ecr-public describe-repositories \
    --region us-east-1 \
    --output table \
    --query "repositories[*].repositoryName"
```

输出：

```
----------------------------
|   DescribeRepositories    |
+---------------------------+
|  project-a/nginx-web-app  |
|  nginx                    |
|  myfirstrepo1             |
|  helm-test-chart          |
|  test-ecr-public          |
|  nginx-web-app            |
|  sample-repo              |
+---------------------------+
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRepository。

get-authorization-token

以下代码示例演示了如何使用 get-authorization-token。

Amazon CLI

示例 1：检索 IAM 主体有权访问的任何 Amazon ECR 公共注册表的授权令牌

以下 get-authorization-token 示例使用 Amazon CLI 获取授权令牌并将它设置为环境变量。

```
aws ecr-public get-authorization-token \
```

```
    --region us-east-1
```

输出：

```
{
    "authorizationData": {
        "authorizationToken":
 "QVdTOmV5SndZWGxzYjJKJFHDSFKJHERWUY65IOU36TRYEGFNSDLRIUOTUYTHJKLDFGOcmFUQk9OSFV2UVV4a0x6Sm1
        "expiresAt": "2024-07-25T21:37:26.301000-04:00"
    }
}
```

有关更多信息，请参阅 Amazon ECR Public 中的 Amazon ECR 公共注册表。

示例 2：检索 IAM 主体有权访问的任何 Amazon ECR 公共注册表的授权令牌

以下 `get-authorization-token` 示例使用 Amazon CLI 获取授权令牌并将它设置为环境变量。

```
aws ecr-public get-authorization-token \
    --region us-east-1 \
    --output=text \
    --query 'authorizationData.authorizationToken'
```

输出：

```
QVdTOmV5SndZWGxzYjJKJFHDSFKJHERWUY65IOU36TRYEGFNSDLRIUOTUYTHJKLDFGOcmFUQk9OSFV2UVV4a0x6Sm1ZV
```

有关更多信息，请参阅 Amazon ECR Public 中的 Amazon ECR 公共注册表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthorizationToken。

`get-login-password`

以下代码示例演示了如何使用 `get-login-password`。

Amazon CLI

示例 1：向 Amazon ECR 公共注册表验证 Docker

以下 `get-login-password` 示例使用 GetAuthorizationToken API 检索并显示身份验证令牌，该令牌可用于向 Amazon ECR 公共注册表进行身份验证。

```
aws ecr-public get-login-password \
    --region us-
east-1
| docker login \
    --username AWS \
    --password-stdin public.ecr.aws
```

此命令不会在终端生成任何输出，而是通过管道将输出传送到 Docker。

有关更多信息，请参阅 Amazon ECR Public 中的向公共注册表进行身份验证。

示例 2：向您自己的自定义 Amazon ECR 公共注册表验证 Docker

以下 `get-login-password` 示例使用 GetAuthorizationToken API 检索并显示身份验证令牌，该令牌可用于向您自己的自定义 Amazon ECR 公共注册表进行身份验证。

```
 aws ecr-public get-login-password \
    --region us-east-1 \
| docker login \
    --username AWS \
    --password-stdin public.ecr.aws/<your-public-registry-custom-alias>
```

此命令不会在终端生成任何输出，而是通过管道将输出传送到 Docker。

有关更多信息，请参阅 Amazon ECR Public 中的向您自己的 Amazon ECR Public 进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoginPassword。

get-registry-catalog-data

以下代码示例演示了如何使用 `get-registry-catalog-data`。

Amazon CLI

检索 ECR 公共注册表的目录元数据

以下 `get-registry-catalog-data` 检索 ECR 公共注册表的目录元数据。

```
aws ecr-public get-registry-catalog-data \
    --region us-east-1
```

输出：

```
{
    "registryCatalogData": {
        "displayName": "YourCustomPublicRepositoryalias"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetRegistryCatalogData。

get-repository-catalog-data

以下代码示例演示了如何使用 get-repository-catalog-data。

Amazon CLI

在公共注册表中检索存储库的目录元数据

以下 get-repository-catalog-data 示例在公共注册表中列出存储库 project-a/nginx-web-app 的目录元数据。

```
aws ecr-public get-repository-catalog-data \
    --repository-name project-a/nginx-web-app \
    --region us-east-1
```

输出：

```
{
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ],
        "logoUrl": "https://d3g9o9u8re44ak.cloudfront.net/logo/491d3846-8f33-4d8b-
a10c-c2ce271e6c0d/4f09d87c-2569-4916-a932-5c296bf6f88a.png",
        "aboutText": "## Quick reference\n\nMaintained <truncated>",
```

```
        "usageText": "## Supported architectures\n\namd64, arm64v8\n\n##
 <truncated>"
    }
}
```

有关更多信息，请参阅 Amazon ECR Public 中的存储库目录数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetRepositoryCatalogData。

get-repository-policy

以下代码示例演示了如何使用 get-repository-policy。

Amazon CLI

获取与存储库关联的存储库策略

以下 get-repository-policy 示例获取与存储库关联的存储库策略。

```
aws ecr-public get-repository-policy \
    --repository-name project-a/nginx-web-app \
    --region us-east-1
```

输出：

```
{
    "registryId": "123456789012",
    "repositoryName": "project-a/nginx-web-app",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
    \"Sid\" : \"AllowPush\",\n    \"Effect\" : \"Allow\",\n    \"Principal
\" : {\n      \"AWS\" : [ \"arn:aws:iam::123456789012:user/eksuser1\",
 \"arn:aws:iam::123456789012:user/admin\" ]\n    },\n    \"Action\" :
 [ \"ecr-public:BatchCheckLayerAvailability\", \"ecr-public:PutImage\",
 \"ecr-public:InitiateLayerUpload\", \"ecr-public:UploadLayerPart\", \"ecr-
public:CompleteLayerUpload\" ]\n  } ]\n}"
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Use GetRepositoryPolicy with an Amazon SDK or CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRepositoryPolicy。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

在公共注册表中列出公共存储库的标签

以下 list-tags-for-resource 示例在公共注册表中列出名为 project-a/nginx-web-app 的资源的标签。

```
aws ecr-public list-tags-for-resource \
    --resource-arn arn:aws:ecr-public::123456789012:repository/project-a/nginx-web-app \
    --region us-east-1
```

输出：

```
{
    "tags": [
        {
            "Key": "Environment",
            "Value": "Prod"
        },
        {
            "Key": "stack",
            "Value": "dev1"
        },
        {
            "Key": "Name",
            "Value": "project-a/nginx-web-app"
        }
    ]
}
```

有关更多信息，请参阅 Amazon ECR Public 中的列出公共存储库的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-registry-catalog-data

以下代码示例演示了如何使用 put-registry-catalog-data。

Amazon CLI

为 ECR 公共注册表创建或更新目录元数据

以下 put-registry-catalog-data 创建或更新 ECR 公共注册表的目录元数据。仅拥有经过验证的账户徽章的账户才能获得注册表显示名称。

```
aws ecr-public put-registry-catalog-data \
    --region us-east-1 \
    --display-name <YourCustomPublicRepositoryalias>
```

输出：

```
{
    "registryCatalogData": {
        "displayName": "YourCustomPublicRepositoryalias"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutRegistryCatalogData。

put-repository-catalog-data

以下代码示例演示了如何使用 put-repository-catalog-data。

Amazon CLI

在公共注册表中创建或更新存储库的目录数据

以下 put-repository-catalog-data 示例在公共注册表中创建或更新名为 project-a/nginx-web-app 的存储库的目录数据，以及 logoImageBlob、aboutText、usageText 和标签信息。

```
aws ecr-public put-repository-catalog-data \
    --repository-name project-a/nginx-web-app \
    --cli-input-json file://repository-catalog-data.json \
    --region us-east-1
```

的内容repository-catalog-data.json：

```
{
```

```
    "repositoryName": "project-a/nginx-web-app",
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ],
        "logoImageBlob": "iVBORw0KGgoA<<truncated-for-better-reading>>ErkJggg==",
        "aboutText": "## Quick reference.",
        "usageText": "## Supported architectures are as follows."
    }
}
```

输出：

```
{
    "catalogData": {
        "description": "My project-a ECR Public Repository",
        "architectures": [
            "ARM",
            "ARM 64",
            "x86",
            "x86-64"
        ],
        "operatingSystems": [
            "Linux"
        ],
        "logoUrl": "https://d3g9o9u8re44ak.cloudfront.net/logo/df86cf58-ee60-4061-
b804-0be24d97ccb1/4a9ed9b2-69e4-4ede-b924-461462d20ef0.png",
        "aboutText": "## Quick reference.",
        "usageText": "## Supported architectures are as follows."
    }
}
```

有关更多信息，请参阅 Amazon ECR Public 中的存储库目录数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutRepositoryCatalogData。

set-repository-policy

以下代码示例演示了如何使用 set-repository-policy。

Amazon CLI

示例 1：将存储库策略设置为允许拉取存储库

以下 set-repository-policy 示例将 ECR 公共存储库策略应用于指定的存储库以控制访问权限。

```
aws ecr-public set-repository-policy \
    --repository-name project-a/nginx-web-app \
    --policy-text file://my-repository-policy.json
```

的内容my-repository-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Sid" : "allow public pull",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer"
            ]
        }
    ]
}
```

输出：

```
{
    "registryId": "12345678901",
    "repositoryName": "project-a/nginx-web-app",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
 \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :
 \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage
\", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
```

```
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Setting a repository policy statement。

示例 2：设置存储库策略以允许您账户中的 IAM 用户推送映像

以下 `set-repository-policy` 示例允许您账户中的 IAM 用户通过使用名为 `file://my-repository-policy.json` 输入文件作为策略文本，将映像推送到您的 Amazon 账户中的 ECR 存储库。

```
aws ecr-public set-repository-policy \
    --repository-name project-a/nginx-web-app \
    --policy-text file://my-repository-policy.json
```

的内容`my-repository-policy.json`：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPush",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::account-id:user/push-pull-user-1",
                    "arn:aws:iam::account-id:user/push-pull-user-2"
                ]
            },
            "Action": [
                "ecr-public:BatchCheckLayerAvailability",
                "ecr-public:PutImage",
                "ecr-public:InitiateLayerUpload",
                "ecr-public:UploadLayerPart",
                "ecr-public:CompleteLayerUpload"
            ]
        }
    ]
}
```

输出：

```
{
    "registryId": "12345678901",
    "repositoryName": "project-a/nginx-web-app",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" :
 [ {\n    \"Sid\" : \"AllowPush\",\n    \"Effect\" : \"Allow\",\n
 \"Principal\" : {\n      \"AWS\" : [ \"arn:aws:iam::12345678901:user/admin
\", \"arn:aws:iam::12345678901:user/eksuser1\" ]\n    },\n    \"Action\" :
 [ \"ecr-public:BatchCheckLayerAvailability\", \"ecr-public:PutImage\",
 \"ecr-public:InitiateLayerUpload\", \"ecr-public:UploadLayerPart\", \"ecr-
public:CompleteLayerUpload\" ]\n  } ]\n}"
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Setting a repository policy statement。

示例 3：设置存储库策略以允许其他账户中的 IAM 用户推送映像

以下 `set-repository-policy` 示例允许特定账户在您的 Amazon 账户中使用 cli 输入 file://my-repository-policy.json 来推送映像。

```
aws ecr-public set-repository-policy \
    --repository-name project-a/nginx-web-app \
    --policy-text file://my-repository-policy.json
```

的内容`my-repository-policy.json`：

```
 {
     "Version":"2012-10-17",
     "Statement": [
         {
             "Sid": "AllowCrossAccountPush",
             "Effect": "Allow",
             "Principal": {
                 "AWS": "arn:aws:iam::other-or-same-account-id:role/RoleName"
             },
             "Action": [
                 "ecr-public:BatchCheckLayerAvailability",
                 "ecr-public:PutImage",
                 "ecr-public:InitiateLayerUpload",
                 "ecr-public:UploadLayerPart",
                 "ecr-public:CompleteLayerUpload"
             ]
```

```
        }
    ]
}
```

输出：

```
{
    "registryId": "12345678901",
    "repositoryName": "project-a/nginx-web-app",
    "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
 \"Sid\" : \"AllowCrossAccountPush\",\n    \"Effect\" : \"Allow\",\n    \"Principal
\" : {\n      \"AWS\" : \"arn:aws:iam::12345678901:role/RoleName\"\n    },\n
 \"Action\" : [ \"ecr-public:BatchCheckLayerAvailability\", \"ecr-public:PutImage
\", \"ecr-public:InitiateLayerUpload\", \"ecr-public:UploadLayerPart\", \"ecr-
public:CompleteLayerUpload\" ]\n  } ]\n}"
}
```

有关更多信息，请参阅《Amazon ECR Public User Guide》中的 Public repository policy examples。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetRepositoryPolicy。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

示例 1：在公共注册表中标记现有公共存储库

以下 tag-resource 示例在公共注册表中标记名为 project-a/nginx-web-app 的存储库。

```
aws ecr-public tag-resource \
    --resource-arn arn:aws:ecr-public::123456789012:repository/project-a/nginx-web-
app \
    --tags Key=stack,Value=dev \
    --region us-east-1
```

有关更多信息，请参阅 Amazon ECR Public 中的对公共存储库使用标签。

示例 2：在公共注册表中使用多个标签标记现有公共存储库。

以下 tag-resource 示例使用多个标签标记现有存储库。

```
aws ecr-public tag-resource \
    --resource-arn arn:aws:ecr-public::890517186334:repository/project-a/nginx-web-app  \
    --tags Key=key1,Value=value1 Key=key2,Value=value2 Key=key3,Value=value3 \
    --region us-east-1
```

有关更多信息，请参阅 Amazon ECR Public 中的对公共存储库使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：在公共注册表中取消标记现有公共存储库

以下 untag-resource 示例在公共注册表中标记名为 project-a/nginx-web-app 的存储库。

```
aws ecr-public untag-resource \
    --resource-arn arn:aws:ecr-public::123456789012:repository/project-a/nginx-web-app \
    --tag-keys stack \
    --region us-east-1
```

此命令不生成任何输出。

有关更多信息，请参阅 Amazon ECR Public 中的对公共存储库使用标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Amazon ECS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon ECS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`capacity-provider-update`

以下代码示例演示了如何使用 `capacity-provider-update`。

Amazon CLI

更新 ECS 集群中的容量提供程序

以下 `update-capacity-provider` 示例演示如何修改 ECS 集群中容量提供程序的参数。

```
aws ecs update-capacity-provider \
    --name Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt \
    --auto-scaling-group-
provider "managedScaling={status=DISABLED,targetCapacity=50,minimumScalingStepSize=2,maximum
```

输出：

```
{
    "capacityProvider": {
        "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt",
        "name": "Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:424941d1-b43f-4a17-
adbb-08b6a6e397e1:autoScalingGroupName/Infra-ECS-Cluster-ECS-project-update-cluster-
d6bb6d5b-ECSAutoScalingGroup-f44jrQHS2nRB",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 10000,
                "instanceWarmupPeriod": 300
```

```
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "ENABLED"
        },
        "updateStatus": "UPDATE_IN_PROGRESS",
        "tags": []
    }
}
```

有关容量提供程序的更多信息，请参阅《Amazon ECS 开发人员指南》中的 EC2 启动类型的 Amazon ECS 容量提供程序。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CapacityProviderUpdate。

`create-capacity-provider`

以下代码示例演示了如何使用 `create-capacity-provider`。

Amazon CLI

创建容量提供程序

以下 create-capacity-provider 示例创建一个容量提供程序，该容量提供程序使用名为 MyASG 的自动扩缩组，并启用了托管扩展和托管终止保护。此配置用于 Amazon ECS 集群自动扩缩。

```
aws ecs create-capacity-provider \
    --name "MyCapacityProvider" \
    --auto-scaling-group-provider "autoScalingGroupArn=arn:aws:autoscaling:us-
east-1:123456789012:autoScalingGroup:57ffcb94-11f0-4d6d-
bf60-3bac5EXAMPLE:autoScalingGroupName/
MyASG,managedScaling={status=ENABLED,targetCapacity=100},managedTerminationProtection=ENABLE
```

输出：

```
{
    "capacityProvider": {
    "capacityProviderArn": "arn:aws:ecs:us-east-1:123456789012:capacity-provider/
MyCapacityProvider",
    "name": "MyCapacityProvider",
    "status": "ACTIVE",
    "autoScalingGroupProvider": {
```

```
        "autoScalingGroupArn": "arn:aws:autoscaling:us-
east-1:132456789012:autoScalingGroup:57ffcb94-11f0-4d6d-
bf60-3bac5EXAMPLE:autoScalingGroupName/MyASG",
        "managedScaling": {
            "status": "ENABLED",
            "targetCapacity": 100,
            "minimumScalingStepSize": 1,
            "maximumScalingStepSize": 10000,
            "instanceWarmupPeriod": 300
        },
        "managedTerminationProtection": "ENABLED"
    },
    "tags": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 集群自动扩缩。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCapacityProvider。

create-cluster

以下代码示例演示了如何使用 create-cluster。

Amazon CLI

示例 1：创建新集群

以下 create-cluster 示例创建一个名为 MyCluster 的集群，并启用具有增强型可观测性的 CloudWatch Container Insights。

```
aws ecs create-cluster \
    --cluster-name MyCluster \
    --settings name=containerInsights,value=enhanced
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
```

```
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "enhanced"
            }
        ],
        "tags": []
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建集群。

示例 2：使用容量提供程序创建新集群

以下 create-cluster 示例将创建一个集群并将两个现有容量提供程序与该集群相关联。使用 create-capacity-provider 命令创建容量提供程序。指定默认容量提供程序策略是可选的，但建议您这样做。在此示例中，我们创建一个名为 MyCluster 的集群，并将 MyCapacityProvider1 和 MyCapacityProvider2 容量提供程序与其相关联。指定默认容量提供程序策略，将任务平均分散到两个容量提供程序。

```
aws ecs create-cluster \
    --cluster-name MyCluster \
    --capacity-providers MyCapacityProvider1 MyCapacityProvider2 \
    --default-capacity-provider-
strategy capacityProvider=MyCapacityProvider1,weight=1 capacityProvider=MyCapacityProvider2,
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "PROVISIONING",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "settings": [
```

```
            {
                "name": "containerInsights",
                "value": "enabled"
            }
        ],
        "capacityProviders": [
            "MyCapacityProvider1",
            "MyCapacityProvider2"
        ],
        "defaultCapacityProviderStrategy": [
            {
                "capacityProvider": "MyCapacityProvider1",
                "weight": 1,
                "base": 0
            },
            {
                "capacityProvider": "MyCapacityProvider2",
                "weight": 1,
                "base": 0
            }
        ],
        "attachments": [
           {
               "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
               "type": "asp",
               "status": "PRECREATED",
               "details": [
                   {
                       "name": "capacityProviderName",
                       "value": "MyCapacityProvider1"
                   },
                   {
                       "name": "scalingPlanName",
                       "value": "ECSManagedAutoScalingPlan-a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
                   }
               ]
            },
            {
                "id": "ae592060-2382-4663-9476-b015c685593c",
                "type": "asp",
                "status": "PRECREATED",
                "details": [
                    {
```

```
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider2"
                    },
                    {
                        "name": "scalingPlanName",
                        "value": "ECSManagedAutoScalingPlan-a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222"
                    }
                ]
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

示例 3：创建带有多个标签的新集群

以下 `create-cluster` 示例将创建一个带有多个标签的集群。有关使用速记语法添加标签的更多信息，请参阅《Amazon CLI 用户指南》中的通过 Amazon 命令行界面使用速记语法。

```
aws ecs create-cluster \
    --cluster-name MyCluster \
    --tags key=key1,value=value1 key=key2,value=value2
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [
            {
                "key": "key1",
                "value": "value1"
            },
```

```
            {
                "key": "key2",
                "value": "value2"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-service

以下代码示例演示了如何使用 create-service。

Amazon CLI

示例 1：使用 Fargate 任务创建服务

以下 create-service 示例演示了如何使用 Fargate 任务创建服务。

```
aws ecs create-service \
    --cluster MyCluster \
    --service-name MyService \
    --task-definition sample-fargate:1 \
    --desired-count 2 \
    --launch-type FARGATE \
    --platform-version LATEST \
    --network-configuration
 'awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp
 \
    --tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/
MyService",
        "serviceName": "MyService",
          "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "loadBalancers": [],
```

```
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sample-fargate:1",
        "deploymentConfiguration": {
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "deployments": [
            {
                "id": "ecs-svc/1234567890123456789",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/sample-fargate:1",
                "desiredCount": 2,
                "pendingCount": 0,
                "runningCount": 0,
                "createdAt": 1557119253.821,
                "updatedAt": 1557119253.821,
                "launchType": "FARGATE",
                "platformVersion": "1.3.0",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                }
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [],
        "createdAt": 1557119253.821,
        "placementConstraints": [],
```

```
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "schedulingStrategy": "REPLICA",
        "tags": [
            {
                "key": "key1",
                "value": "value1"
            },
            {
                "key": "key2",
                "value": "value2"
            },
            {
                "key": "key3",
                "value": "value3"
            }
        ],
        "enableECSManagedTags": false,
        "propagateTags": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建服务。

示例 2：使用 EC2 启动类型创建服务

以下 `create-service` 示例演示如何通过使用 EC2 启动类型的任务创建名为 `ecs-simple-service` 的服务。该服务使用 `sleep360` 作业定义并维护任务的 1 个实例化。

```
aws ecs create-service \
    --cluster MyCluster \
    --service-name ecs-simple-service \
    --task-definition sleep360:2 \
```

```
    --desired-count 1
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/ecs-
simple-service",
        "serviceName": "ecs-simple-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sleep360:2",
        "deploymentConfiguration": {
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "deployments": [
            {
                "id": "ecs-svc/1234567890123456789",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/sleep360:2",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 0,
                "createdAt": 1557206498.798,
                "updatedAt": 1557206498.798,
                "launchType": "EC2"
            }
        ],
        "events": [],
        "createdAt": 1557206498.798,
        "placementConstraints": [],
        "placementStrategy": [],
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": false,
```

```
        "propagateTags": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建服务。

示例 3：创建使用外部部署控制器的服务

以下 `create-service` 示例将创建使用外部部署控制器的服务。

```
aws ecs create-service \
    --cluster MyCluster \
    --service-name MyService \
    --deployment-controller type=EXTERNAL \
    --desired-count 1
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/
MyService",
        "serviceName": "MyService",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "deploymentConfiguration": {
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "taskSets": [],
        "deployments": [],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [],
        "createdAt": 1557128207.101,
        "placementConstraints": [],
        "placementStrategy": [],
```

```
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "EXTERNAL"
        },
        "enableECSManagedTags": false,
        "propagateTags": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建服务。

示例 4：在负载均衡器后面创建新服务

以下 `create-service` 示例将显示如何创建位于负载均衡器之后的服务。您必须将负载均衡器配置为您的容器实例所在的同一区域。此示例使用 `--cli-input-json` 选项和名为 `ecs-simple-service-elb.json` 的 JSON 输入文件，该文件包含以下内容。

```
aws ecs create-service \
    --cluster MyCluster \
    --service-name ecs-simple-service-elb \
    --cli-input-json file://ecs-simple-service-elb.json
```

的内容`ecs-simple-service-elb.json`：

```
 {
    "serviceName": "ecs-simple-service-elb",
    "taskDefinition": "ecs-demo",
    "loadBalancers": [
        {
            "loadBalancerName": "EC2Contai-EcsElast-123456789012",
            "containerName": "simple-demo",
            "containerPort": 80
        }
    ],
    "desiredCount": 10,
    "role": "ecsServiceRole"
}
```

输出：

```
{
    "service": {
```

```
        "status": "ACTIVE",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/ecs-
demo:1",
        "pendingCount": 0,
        "loadBalancers": [
            {
                "containerName": "ecs-demo",
                "containerPort": 80,
                "loadBalancerName": "EC2Contai-EcsElast-123456789012"
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/ecsServiceRole",
        "desiredCount": 10,
        "serviceName": "ecs-simple-service-elb",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/ecs-simple-
service-elb",
        "deployments": [
            {
                "status": "PRIMARY",
                "pendingCount": 0,
                "createdAt": 1428100239.123,
                "desiredCount": 10,
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/ecs-demo:1",
                "updatedAt": 1428100239.123,
                "id": "ecs-svc/1234567890123456789",
                "runningCount": 0
            }
        ],
        "events": [],
        "runningCount": 0
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用负载均衡分配 Amazon ECS 服务流量。

示例 5：在创建服务时配置 Amazon EBS 卷

以下 `create-service` 示例说明如何为服务管理的每个任务配置 Amazon EBS 卷。您必须配置了一个附有 `AmazonECSInfrastructureRolePolicyForVolumes` 托管式策略的 Amazon ECS 基础设施角色。您必须使用与 `create-service` 请求中相同的卷名称来指定任务定义。此示例使

用 --cli-input-json 选项和名为 ecs-simple-service-ebs.json 的 JSON 输入文件，该文件包含以下内容。

```
aws ecs create-service \
    --cli-input-json file://ecs-simple-service-ebs.json
```

的内容ecs-simple-service-ebs.json：

```
{
    "cluster": "mycluster",
    "taskDefinition": "mytaskdef",
    "serviceName": "ecs-simple-service-ebs",
    "desiredCount": 2,
    "launchType": "FARGATE",
    "networkConfiguration":{
        "awsvpcConfiguration":{
            "assignPublicIp": "ENABLED",
            "securityGroups": ["sg-12344321"],
            "subnets":["subnet-12344321"]
        }
    },
    "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
                "roleArn":"arn:aws:iam::123456789012:role/ecsInfrastructureRole",
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/mycluster/ecs-
simple-service-ebs",
```

```
        "serviceName": "ecs-simple-service-ebs",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/mycluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
mytaskdef:3",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
                "rollback": false
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "deployments": [
            {
                "id": "ecs-svc/7851020056849183687",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:3",
                "desiredCount": 0,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": "2025-01-21T11:32:38.034000-06:00",
                "updatedAt": "2025-01-21T11:32:38.034000-06:00",
                "launchType": "EC2",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "DISABLED"
                    }
                },
                "rolloutState": "IN_PROGRESS",
```

```
                "rolloutStateReason": "ECS deployment ecs-svc/7851020056849183687 in
 progress.",
                "volumeConfigurations": [
                    {
                        "name": "myEBSVolume",
                        "managedEBSVolume": {
                            "volumeType": "gp3",
                            "sizeInGiB": 100,
                            "iops": 3000,
                            "throughput": 125,
                            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
                            "filesystemType": "ext4"
                        }
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [],
        "createdAt": "2025-01-21T11:32:38.034000-06:00",
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "DISABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:user/AIDACKCEVSQ6C2EXAMPLE",
        "enableECSManagedTags": false,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
```

```
        "availabilityZoneRebalancing": "DISABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将 Amazon EBS 卷与 Amazon ECS 结合使用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateService。

create-task-set

以下代码示例演示了如何使用 create-task-set。

Amazon CLI

创建任务集

以下 create-task-set 示例在使用外部部署控制器的服务中创建一个任务集。

```
aws ecs create-task-set \
    --cluster MyCluster \
    --service MyService \
    --task-definition MyTaskDefinition:2 \
    --network-
configuration "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321]}"
```

输出：

```
{
    "taskSet": {
        "id": "ecs-svc/1234567890123456789",
        "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789",
        "status": "ACTIVE",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
MyTaskDefinition:2",
        "computedDesiredCount": 0,
        "pendingCount": 0,
        "runningCount": 0,
        "createdAt": 1557128360.711,
        "updatedAt": 1557128360.711,
        "launchType": "EC2",
```

```
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "DISABLED"
            }
        },
        "loadBalancers": [],
        "serviceRegistries": [],
        "scale": {
            "value": 0.0,
            "unit": "PERCENT"
        },
        "stabilityStatus": "STABILIZING",
        "stabilityStatusAt": 1557128360.711
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTaskSet。

delete-account-setting

以下代码示例演示了如何使用 delete-account-setting。

Amazon CLI

删除特定 IAM 用户或 IAM 角色的账户设置

以下 delete-account-setting 示例删除特定 IAM 用户或 IAM 角色的账户设置。

```
aws ecs delete-account-setting \
    --name serviceLongArnFormat \
    --principal-arn arn:aws:iam::123456789012:user/MyUser
```

输出：

```
{
    "setting": {
```

```
        "name": "serviceLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:user/MyUser"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon 资源名称（ARN）和 ID。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccountSetting。

delete-attributes

以下代码示例演示了如何使用 delete-attributes。

Amazon CLI

从 Amazon ECS 资源中删除一个或多个自定义属性

以下 delete-attributes 从容器实例中删除名为 stack 的属性。

```
aws ecs delete-attributes \
    --attributes name=stack,targetId=arn:aws:ecs:us-west-2:130757420319:container-instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34
```

输出：

```
{
    "attributes": [
        {
            "name": "stack",
            "targetId": "arn:aws:ecs:us-west-2:130757420319:container-instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",
            "value": "production"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAttributes。

delete-capacity-provider

以下代码示例演示了如何使用 delete-capacity-provider。

Amazon CLI

示例 1：使用 Amazon 资源名称（ARN）删除容量提供程序

以下 delete-capacity-provider 示例通过指定容量提供程序的 Amazon 资源名称（ARN）来删除该容量提供程序。可以使用 describe-capacity-providers 命令检索 ARN 以及容量提供程序删除的状态。

```
aws ecs delete-capacity-provider \
    --capacity-provider arn:aws:ecs:us-west-2:123456789012:capacity-provider/
ExampleCapacityProvider
```

输出：

```
{
    "capacityProvider": {
        "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/ExampleCapacityProvider",
        "name": "ExampleCapacityProvider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 10000
            },
            "managedTerminationProtection": "DISABLED"
        },
        "updateStatus": "DELETE_IN_PROGRESS",
        "tags": []
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

示例 2：使用名称删除容量提供程序

以下 delete-capacity-provider 示例通过指定容量提供程序的短名称来删除该容量提供程序。可以使用 describe-capacity-providers 命令检索容量提供程序删除的短名称及状态。

```
aws ecs delete-capacity-provider \
    --capacity-provider ExampleCapacityProvider
```

输出：

```
{
    "capacityProvider": {
        "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/ExampleCapacityProvider",
        "name": "ExampleCapacityProvider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 10000
            },
            "managedTerminationProtection": "DISABLED"
        },
        "updateStatus": "DELETE_IN_PROGRESS",
        "tags": []
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCapacityProvider。

delete-cluster

以下代码示例演示了如何使用 delete-cluster。

Amazon CLI

删除空集群

以下 delete-cluster 示例将删除指定的空集群。

```
aws ecs delete-cluster --cluster MyCluster
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "status": "INACTIVE",
        "clusterName": "MyCluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0
        "statistics": [],
        "tags": []
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的删除集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCluster。

delete-service

以下代码示例演示了如何使用 delete-service。

Amazon CLI

删除服务

以下 ecs delete-service 示例将从集群中删除指定的服务。您可以包含 --force 参数来删除服务，即使它尚未缩减至 0 个任务。

```
aws ecs delete-service --cluster MyCluster --service MyService1 --force
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的删除服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteService。

delete-task-definitions

以下代码示例演示了如何使用 delete-task-definitions。

Amazon CLI

删除作业定义

以下 delete-task-definitions 示例删除 INACTIVE 作业定义。

```
aws ecs delete-task-definitions \
    --task-definition curltest:1
```

输出：

```
{
"taskDefinitions": [
    {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
curltest:1",
        "containerDefinitions": [
            {
                "name": "ctest",
                "image": "mreferre/eksutils",
                "cpu": 0,
                "portMappings": [],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "curl ${ECS_CONTAINER_METADATA_URI_V4}/task"
                ],
                "environment": [],
                "mountPoints": [],
                "volumesFrom": [],
                "logConfiguration": {
                    "logDriver": "awslogs",
                    "options": {
                        "awslogs-create-group": "true",
                        "awslogs-group": "/ecs/curltest",
                        "awslogs-region": "us-east-1",
                        "awslogs-stream-prefix": "ecs"
                    }
                }
            }
```

```
        ],
        "family": "curltest",
        "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
        "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
        "networkMode": "awsvpc",
        "revision": 1,
        "volumes": [],
        "status": "DELETE_IN_PROGRESS",
        "compatibilities": [
            "EC2",
            "FARGATE"
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512",
        "registeredAt": "2021-09-10T12:56:24.704000+00:00",
        "deregisteredAt": "2023-03-14T15:20:59.419000+00:00",
        "registeredBy": "arn:aws:sts::123456789012:assumed-role/Admin/jdoe"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTaskDefinitions。

delete-task-set

以下代码示例演示了如何使用 delete-task-set。

Amazon CLI

删除任务集

以下 delete-task-set 示例演示如何删除任务集。即使任务集尚未缩减至零，您也可以包含 --force 参数来删除该任务集。

```
aws ecs delete-task-set \
    --cluster MyCluster \
    --service MyService \
```

```
    --task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-
svc/1234567890123456789 \
    --force
```

输出：

```
{
    "taskSet": {
        "id": "ecs-svc/1234567890123456789",
        "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789",
        "status": "DRAINING",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sample-fargate:2",
        "computedDesiredCount": 0,
        "pendingCount": 0,
        "runningCount": 0,
        "createdAt": 1557130260.276,
        "updatedAt": 1557130290.707,
        "launchType": "EC2",
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12345678"
                ],
                "securityGroups": [
                    "sg-12345678"
                ],
                "assignPublicIp": "DISABLED"
            }
        },
        "loadBalancers": [],
        "serviceRegistries": [],
        "scale": {
            "value": 0.0,
            "unit": "PERCENT"
        },
        "stabilityStatus": "STABILIZING",
        "stabilityStatusAt": 1557130290.707
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTaskSet。

deregister-container-instance

以下代码示例演示了如何使用 deregister-container-instance。

Amazon CLI

从集群中取消注册容器实例

以下 deregister-container-instance 示例从指定集群中取消注册容器实例。如果容器实例中仍有任务正在运行，则必须在取消注册之前停止这些任务，或使用 --force 选项。

```
aws ecs deregister-container-instance \
    --cluster arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \
    --container-instance arn:aws:ecs:us-west-2:123456789012:container-instance/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --force
```

输出：

```
{
    "containerInstance": {
        "remainingResources": [
            {
                "integerValue": 1024,
                "doubleValue": 0.0,
                "type": "INTEGER",
                "longValue": 0,
                "name": "CPU"
            },
            {
                "integerValue": 985,
                "doubleValue": 0.0,
                "type": "INTEGER",
                "longValue": 0,
                "name": "MEMORY"
            },
            {
                "type": "STRINGSET",
                "integerValue": 0,
                "name": "PORTS",
                "stringSetValue": [
                    "22",
                    "2376",
```

```
                    "2375",
                    "51678",
                    "51679"
                ],
                "longValue": 0,
                "doubleValue": 0.0
            },
            {
                "type": "STRINGSET",
                "integerValue": 0,
                "name": "PORTS_UDP",
                "stringSetValue": [],
                "longValue": 0,
                "doubleValue": 0.0
            }
        ],
        "agentConnected": true,
        "attributes": [
            {
                "name": "ecs.capability.secrets.asm.environment-variables"
            },
            {
                "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
            },
            {
                "value": "ami-01a82c3fce2c3ba58",
                "name": "ecs.ami-id"
            },
            {
                "name": "ecs.capability.secrets.asm.bootstrap.log-driver"
            },
            {
                "name": "com.amazonaws.ecs.capability.logging-driver.none"
            },
            {
                "name": "ecs.capability.ecr-endpoint"
            },
            {
                "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
            },
            {
                "value": "vpc-1234567890123467",
                "name": "ecs.vpc-id"
            },
```

```
            {
                "name": "ecs.capability.execution-role-awslogs"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
            },
            {
                "name": "ecs.capability.docker-plugin.local"
            },
            {
                "name": "ecs.capability.task-eni"
            },
            {
                "name": "ecs.capability.task-cpu-mem-limit"
            },
            {
                "name": "ecs.capability.secrets.ssm.bootstrap.log-driver"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.30"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.31"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.32"
            },
            {
                "name": "ecs.capability.execution-role-ecr-pull"
            },
            {
                "name": "ecs.capability.container-health-check"
            },
            {
                "value": "subnet-1234567890123467",
                "name": "ecs.subnet-id"
            },
            {
```

```
                "value": "us-west-2a",
                "name": "ecs.availability-zone"
            },
            {
                "value": "t2.micro",
                "name": "ecs.instance-type"
            },
            {
                "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
            },
            {
                "name": "ecs.capability.aws-appmesh"
            },
            {
                "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.24"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.25"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.26"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.27"
            },
            {
                "name": "com.amazonaws.ecs.capability.privileged-container"
            },
            {
                "name": "ecs.capability.container-ordering"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.28"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.29"
            },
            {
                "value": "x86_64",
                "name": "ecs.cpu-architecture"
            },
```

```
            {
                "value": "93f43776-2018.10.0",
                "name": "ecs.capability.cni-plugin-version"
            },
            {
                "name": "ecs.capability.secrets.ssm.environment-variables"
            },
            {
                "name": "ecs.capability.pid-ipc-namespace-sharing"
            },
            {
                "name": "com.amazonaws.ecs.capability.ecr-auth"
            },
            {
                "value": "linux",
                "name": "ecs.os-type"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
            },
            {
                "name": "ecs.capability.task-eia"
            },
            {
                "name": "ecs.capability.private-registry-
authentication.secretsmanager"
            },
            {
                "name": "com.amazonaws.ecs.capability.task-iam-role"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
            }
        ],
        "pendingTasksCount": 0,
        "tags": [],
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
```

```
        "registeredResources": [
            {
                "integerValue": 1024,
                "doubleValue": 0.0,
                "type": "INTEGER",
                "longValue": 0,
                "name": "CPU"
            },
            {
                "integerValue": 985,
                "doubleValue": 0.0,
                "type": "INTEGER",
                "longValue": 0,
                "name": "MEMORY"
            },
            {
                "type": "STRINGSET",
                "integerValue": 0,
                "name": "PORTS",
                "stringSetValue": [
                    "22",
                    "2376",
                    "2375",
                    "51678",
                    "51679"
                ],
                "longValue": 0,
                "doubleValue": 0.0
            },
            {
                "type": "STRINGSET",
                "integerValue": 0,
                "name": "PORTS_UDP",
                "stringSetValue": [],
                "longValue": 0,
                "doubleValue": 0.0
            }
        ],
        "status": "INACTIVE",
        "registeredAt": 1557768075.681,
        "version": 4,
        "versionInfo": {
            "agentVersion": "1.27.0",
            "agentHash": "aabe65ee",
```

```
            "dockerVersion": "DockerVersion: 18.06.1-ce"
        },
        "attachments": [],
        "runningTasksCount": 0,
        "ec2InstanceId": "i-12345678901234678"
    }
}
```

有关更多信息，请参阅《ECS 开发人员指南》中的取消注册容器实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterContainerInstance。

deregister-task-definition

以下代码示例演示了如何使用 deregister-task-definition。

Amazon CLI

取消注册作业定义

以下 deregister-task-definition 示例在您的默认区域中取消注册 curler 作业定义的第一个修订版。

```
aws ecs deregister-task-definition --task-definition curler:1
```

请注意，在生成的输出中，作业定义状态显示 INACTIVE：

```
{
    "taskDefinition": {
        "status": "INACTIVE",
        "family": "curler",
        "volumes": [],
        "taskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:task-definition/
curler:1",
        "containerDefinitions": [
            {
                "environment": [],
                "name": "curler",
                "mountPoints": [],
                "image": "curl:latest",
                "cpu": 100,
                "portMappings": [],
                "entryPoint": [],
```

```
                "memory": 256,
                "command": [
                    "curl -v http://example.com/"
                ],
                "essential": true,
                "volumesFrom": []
            }
        ],
        "revision": 1
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTaskDefinition。

describe-capacity-providers

以下代码示例演示了如何使用 describe-capacity-providers。

Amazon CLI

示例 1：描述所有容量提供程序

以下 describe-capacity-providers 示例检索有关所有容量提供程序的详细信息。

```
aws ecs describe-capacity-providers
```

输出：

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/MyCapacityProvider",
            "name": "MyCapacityProvider",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",
                "managedScaling": {
                    "status": "ENABLED",
```

```
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1000
                },
                "managedTerminationProtection": "ENABLED"
            },
            "tags": []
        },
        {
            "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/FARGATE",
            "name": "FARGATE",
            "status": "ACTIVE",
            "tags": []
        },
        {
            "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/FARGATE_SPOT",
            "name": "FARGATE_SPOT",
            "status": "ACTIVE",
            "tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

示例 2：描述特定容量提供程序

以下 describe-capacity-providers 示例检索有关特定容量提供程序的详细信息。使用 --include TAGS 参数会将与容量提供程序关联的标签添加到输出中。

```
aws ecs describe-capacity-providers \
    --capacity-providers MyCapacityProvider \
    --include TAGS
```

输出：

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/MyCapacityProvider",
```

```
            "name": "MyCapacityProvider",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1000
                },
                "managedTerminationProtection": "ENABLED"
            },
            "tags": [
                {
                    "key": "environment",
                    "value": "production"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCapacityProviders。

describe-clusters

以下代码示例演示了如何使用 describe-clusters。

Amazon CLI

示例 1：描述集群

以下 describe-clusters 示例将检索指定集群的详细信息。

```
aws ecs describe-clusters \
    --cluster default
```

输出：

```
{
    "clusters": [
        {
            "status": "ACTIVE",
            "clusterName": "default",
            "registeredContainerInstancesCount": 0,
            "pendingTasksCount": 0,
            "runningTasksCount": 0,
            "activeServicesCount": 1,
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 集群。

示例 2：描述带有附加选项的集群

以下 `describe-clusters` 示例将指定 ATTACHMENTS 选项。它以附件的形式检索有关指定集群的详细信息以及附加到集群的资源列表。在集群中使用容量提供程序时，资源（无论是 AutoScaling 计划还是扩展策略）都将以 asp 或 as_policy 附件的形式表示。

```
aws ecs describe-clusters \
    --include ATTACHMENTS \
    --clusters sampleCluster
```

输出：

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:af-south-1:123456789222:cluster/
sampleCluster",
            "clusterName": "sampleCluster",
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
```

```
            "settings": [],
            "capacityProviders": [
                "sampleCapacityProvider"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                {
                    "id": "a1b2c3d4-5678-901b-cdef-EXAMPLE22222",
                    "type": "as_policy",
                    "status": "CREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "sampleCapacityProvider"
                        },
                        {
                            "name": "scalingPolicyName",
                            "value": "ECSManagedAutoScalingPolicy-3048e262-
fe39-4eaf-826d-6f975d303188"
                        }
                    ]
                }
            ],
            "attachmentsStatus": "UPDATE_COMPLETE"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusters。

`describe-container-instances`

以下代码示例演示了如何使用 `describe-container-instances`。

Amazon CLI

描述容器实例

以下 `describe-container-instances` 示例使用容器实例 UUID 作为标识符来检索 `update` 集群中容器实例的详细信息。

```
aws ecs describe-container-instances \
    --cluster update \
    --container-instances a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 2048,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 3955,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "ec2InstanceId": "i-A1B2C3D4",
            "agentConnected": true,
```

```
            "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 2048,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 3955,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "runningTasksCount": 0,
            "versionInfo": {
                "agentVersion": "1.0.0",
                "agentHash": "4023248",
                "dockerVersion": "DockerVersion: 1.5.0"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 容器实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeContainerInstances。

describe-service-deployments

以下代码示例演示了如何使用 describe-service-deployments。

Amazon CLI

描述服务部署详细信息

以下 describe-service-deployments 示例返回服务部署的详细信息以及 ARN arn:aws:ecs:us-east-1:123456789012:service-deployment/example-cluster/example-service/ejGvqq2ilnbKT9qj0vLJe。

```
aws ecs describe-service-deployments \
    --service-deployment-arn arn:aws:ecs:us-east-1:123456789012:service-deployment/example-cluster/example-service/ejGvqq2ilnbKT9qj0vLJe
```

输出：

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-east-1:123456789012:service-deployment/example-cluster/example-service/ejGvqq2ilnbKT9qj0vLJe",
            "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/example-cluster/example-service",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
            "createdAt": "2024-10-31T08:03:30.917000-04:00",
            "startedAt": "2024-10-31T08:03:32.510000-04:00",
            "finishedAt": "2024-10-31T08:05:04.527000-04:00",
            "updatedAt": "2024-10-31T08:05:04.527000-04:00",
            "sourceServiceRevisions": [],
            "targetServiceRevision": {
                "arn": "arn:aws:ecs:us-east-1:123456789012:service-revision/example-cluster/example-service/1485800978477494678",
                "requestedTaskCount": 1,
                "runningTaskCount": 1,
                "pendingTaskCount": 0
            },
            "status": "SUCCESSFUL",
```

```
            "deploymentConfiguration": {
                "deploymentCircuitBreaker": {
                    "enable": true,
                    "rollback": true
                },
                "maximumPercent": 200,
                "minimumHealthyPercent": 100,
                "alarms": {
                    "alarmNames": [],
                    "rollback": false,
                    "enable": false
                }
            },
            "deploymentCircuitBreaker": {
                "status": "MONITORING_COMPLETE",
                "failureCount": 0,
                "threshold": 3
            },
            "alarms": {
                "status": "DISABLED"
            }
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用 Amazon ECS 服务部署查看服务历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeServiceDeployments。

`describe-service-revisions`

以下代码示例演示了如何使用 `describe-service-revisions`。

Amazon CLI

描述服务修订详细信息

以下 `describe-service-revisions` 示例返回服务修订的详细信息以及 ARN `arn:aws:ecs:us-east-1:123456789012:service-revision/example-cluster/example-service/1485800978477494678`。

```
aws ecs describe-service-revisions \
    --service-revision-arns arn:aws:ecs:us-east-1:123456789012:service-revision/
example-cluster/example-service/1485800978477494678
```

输出：

```
{
    "serviceRevisions": [
        {
            "serviceRevisionArn": "arn:aws:ecs:us-east-1:123456789012:service-
revision/example-cluster/example-service/1485800978477494678",
            "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/example-
cluster/example-service",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-
cluster",
            "taskDefinition": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver:5",
            "capacityProviderStrategy": [
                {
                    "capacityProvider": "FARGATE",
                    "weight": 1,
                    "base": 0
                }
            ],
            "platformVersion": "1.4.0",
            "platformFamily": "Linux",
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-0d0eab1bb38d5ca64",
                        "subnet-0db5010045995c2d5"
                    ],
                    "securityGroups": [
                        "sg-02556bf85a191f59a"
                    ],
                    "assignPublicIp": "ENABLED"
                }
            },
            "containerImages": [
                {
                    "containerName": "aws-otel-collector",
                    "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
```

```
                    "image": "public.ecr.aws/aws-observability/aws-otel-
collector:v0.32.0"
                },
                {
                    "containerName": "web",
                    "imageDigest":
 "sha256:28402db69fec7c17e179ea87882667f1e054391138f77ffaf0c3eb388efc3ffb",
                    "image": "nginx"
                }
            ],
            "guardDutyEnabled": false,
            "serviceConnectConfiguration": {
                "enabled": false
            },
            "createdAt": "2024-10-31T08:03:29.302000-04:00"
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 服务修订。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeServiceRevisions。

describe-services

以下代码示例演示了如何使用 describe-services。

Amazon CLI

描述服务

以下 describe-services 示例检索默认集群中 my-http-service 服务的详细信息。

```
aws ecs describe-services --services my-http-service
```

输出：

```
{
    "services": [
        {
```

```
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
amazon-ecs-sample:1",
            "pendingCount": 0,
            "loadBalancers": [],
            "desiredCount": 10,
            "createdAt": 1466801808.595,
            "serviceName": "my-http-service",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-http-
service",
            "deployments": [
                {
                    "status": "PRIMARY",
                    "pendingCount": 0,
                    "createdAt": 1466801808.595,
                    "desiredCount": 10,
                    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/amazon-ecs-sample:1",
                    "updatedAt": 1428326312.703,
                    "id": "ecs-svc/1234567890123456789",
                    "runningCount": 10
                }
            ],
            "events": [
                {
                    "message": "(service my-http-service) has reached a steady
 state.",
                    "id": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
                    "createdAt": 1466801812.435
                }
            ],
            "runningCount": 10
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeServices。

describe-task-definition

以下代码示例演示了如何使用 describe-task-definition。

Amazon CLI

描述作业定义

以下 describe-task-definition 示例检索作业定义的详细信息。

```
aws ecs describe-task-definition \
    --task-definition hello_world:8
```

输出：

```
{
    "taskDefinition": {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:012345678910:task-definition/
hello_world:8",
        "containerDefinitions": [
            {
                "cpu": 10,
                "environment": [],
                "essential": true,
                "image": "wordpress",
                "links": [
                    "mysql"
                ] ,
                "memory": 500,
                "mountPoints": [],
                "name": "wordpress",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80
                    }
                ],
                "volumesFrom": []
            },
            {
                "cpu": 10,
                "environment": [
                    {
                        "name": "MYSQL_ROOT_PASSWORD",
```

```
                        "value": "password"
                    }
                ],
                "essential": true,
                "image": "mysql",
                "memory": 500,
                "mountPoints": [],
                "name": "mysql",
                "portMappings": [],
                "volumesFrom": []
            }
        ],
    "family": "hello_world",
    "revision": 8,
    "volumes": [],
    "status": "ACTIVE",
    "placementConstraints": [],
    "compatibilities": [
        "EXTERNAL",
        "EC2"
    ],
    "registeredAt": "2024-06-21T11:15:12.669000-05:00",
    "registeredBy": "arn:aws:sts::012345678910:assumed-role/demo-role/jane-doe"
    },
    "tags": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTaskDefinition。

describe-task-sets

以下代码示例演示了如何使用 describe-task-sets。

Amazon CLI

描述任务集

以下 describe-task-sets 示例描述使用外部部署程序的服务中的任务集。

```
aws ecs describe-task-sets \
    --cluster MyCluster \
    --service MyService \
```

```
    --task-sets arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-
svc/1234567890123456789
```

输出：

```
{
    "taskSets": [
        {
            "id": "ecs-svc/1234567890123456789",
            "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789",
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sample-fargate:2",
            "computedDesiredCount": 0,
            "pendingCount": 0,
            "runningCount": 0,
            "createdAt": 1557207715.195,
            "updatedAt": 1557207740.014,
            "launchType": "EC2",
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-12344321"
                    ],
                    "securityGroups": [
                        "sg-1234431"
                    ],
                    "assignPublicIp": "DISABLED"
                }
            },
            "loadBalancers": [],
            "serviceRegistries": [],
            "scale": {
                "value": 0.0,
                "unit": "PERCENT"
            },
            "stabilityStatus": "STEADY_STATE",
            "stabilityStatusAt": 1557207740.014
        }
    ],
    "failures": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTaskSets。

describe-tasks

以下代码示例演示了如何使用 describe-tasks。

Amazon CLI

示例 1：描述单个任务

以下 describe-tasks 示例将检索集群中任务的详细信息。您可以使用任务的 ID 或完整 ARN 来指定任务。此示例使用任务的完整 ARN。

```
aws ecs describe-tasks \
    --cluster MyCluster \
    --tasks arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58EXAMPLE
```

输出：

```
{
    "tasks": [
        {
            "attachments": [],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
                    "value": "x86_64"
                }
            ],
            "availabilityZone": "us-east-1b",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
            "connectivity": "CONNECTED",
            "connectivityAt": "2021-08-11T12:21:26.681000-04:00",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-
instance/test/025c7e2c5e054a6790a29fc1fEXAMPLE",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
MyCluster/4d590253bb114126b7afa7b58eea9221/a992d1cc-ea46-474a-b6e8-24688EXAMPLE",
                    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58EXAMPLE",
                    "name": "simple-app",
```

```
                    "image": "httpd:2.4",
                    "runtimeId":
 "91251eed27db90006ad67b1a08187290869f216557717dd5c39b37c94EXAMPLE",
                    "lastStatus": "RUNNING",
                    "networkBindings": [
                        {
                            "bindIP": "0.0.0.0",
                            "containerPort": 80,
                            "hostPort": 80,
                            "protocol": "tcp"
                        }
                    ],
                    "networkInterfaces": [],
                    "healthStatus": "UNKNOWN",
                    "cpu": "10",
                    "memory": "300"
                }
            ],
            "cpu": "10",
            "createdAt": "2021-08-11T12:21:26.681000-04:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
            "group": "service:testupdate",
            "healthStatus": "UNKNOWN",
            "lastStatus": "RUNNING",
            "launchType": "EC2",
            "memory": "300",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "simple-app"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
            "pullStartedAt": "2021-08-11T12:21:28.234000-04:00",
            "pullStoppedAt": "2021-08-11T12:21:33.793000-04:00",
            "startedAt": "2021-08-11T12:21:34.945000-04:00",
            "startedBy": "ecs-svc/968695068243EXAMPLE",
            "tags": [],
            "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58eea9221",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/console-sample-app-static2:1",
```

```
            "version": 2
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

示例 2：描述多个任务

以下 describe-tasks 示例将检索集群中多个任务的详细信息。您可以使用任务的 ID 或完整 ARN 来指定任务。此示例使用任务的完整 ID。

```
aws ecs describe-tasks \
    --cluster MyCluster \
    --tasks "74de0355a10a4f979ac495c14EXAMPLE" "d789e94343414c25b9f6bd59eEXAMPLE"
```

输出：

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d51EXAMPLE",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnet-0d0eab1bb3EXAMPLE"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                        {
                            "name": "macAddress",
                            "value": "0e:89:76:28:07:b3"
                        },
                        {
                            "name": "privateDnsName",
                            "value": "ip-10-0-1-184.ec2.internal"
```

```
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.1.184"
                        }
                    ]
                }
            ],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
                    "value": "x86_64"
                }
            ],
            "availabilityZone": "us-east-1b",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
            "connectivity": "CONNECTED",
            "connectivityAt": "2021-12-20T12:13:37.875000-05:00",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
MyCluster/74de0355a10a4f979ac495c14EXAMPLE/aad3ba00-83b3-4dac-84d4-11f8cEXAMPLE",
                    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/74de0355a10a4f979ac495c14EXAMPLE",
                    "name": "web",
                    "image": "nginx",
                    "runtimeId": "74de0355a10a4f979ac495c14EXAMPLE-265927825",
                    "lastStatus": "RUNNING",
                    "networkBindings": [],
                    "networkInterfaces": [
                        {
                            "attachmentId": "d9e7735a-16aa-4128-bc7a-b2d51EXAMPLE",
                            "privateIpv4Address": "10.0.1.184"
                        }
                    ],
                    "healthStatus": "UNKNOWN",
                    "cpu": "99",
                    "memory": "100"
                }
            ],
            "cpu": "256",
            "createdAt": "2021-12-20T12:13:20.226000-05:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
```

```
            "group": "service:tdsevicetag",
            "healthStatus": "UNKNOWN",
            "lastStatus": "RUNNING",
            "launchType": "FARGATE",
            "memory": "512",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "web"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
            "platformVersion": "1.4.0",
            "platformFamily": "Linux",
            "pullStartedAt": "2021-12-20T12:13:42.665000-05:00",
            "pullStoppedAt": "2021-12-20T12:13:46.543000-05:00",
            "startedAt": "2021-12-20T12:13:48.086000-05:00",
            "startedBy": "ecs-svc/988401040018EXAMPLE",
            "tags": [],
            "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/74de0355a10a4f979ac495c14EXAMPLE",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/webserver:2",
            "version": 3,
            "ephemeralStorage": {
            "sizeInGiB": 20
            }
        },
        {
            "attachments": [
                {
                    "id": "214eb5a9-45cd-4bf8-87bc-57fefEXAMPLE",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnet-0d0eab1bb3EXAMPLE"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-064c7766daEXAMPLE"
                        },
```

```
                        {
                            "name": "macAddress",
                            "value": "0e:76:83:01:17:a9"
                        },
                        {
                            "name": "privateDnsName",
                            "value": "ip-10-0-1-41.ec2.internal"
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.1.41"
                        }
                    ]
                }
            ],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
                    "value": "x86_64"
                }
            ],
            "availabilityZone": "us-east-1b",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
            "connectivity": "CONNECTED",
            "connectivityAt": "2021-12-20T12:13:35.243000-05:00",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
MyCluster/d789e94343414c25b9f6bd59eEXAMPLE/9afef792-609b-43a5-bb6a-3efdbEXAMPLE",
                    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
d789e94343414c25b9f6bd59eEXAMPLE",
                    "name": "web",
                    "image": "nginx",
                    "runtimeId": "d789e94343414c25b9f6bd59eEXAMPLE-265927825",
                    "lastStatus": "RUNNING",
                    "networkBindings": [],
                    "networkInterfaces": [
                        {
                            "attachmentId": "214eb5a9-45cd-4bf8-87bc-57fefEXAMPLE",
                            "privateIpv4Address": "10.0.1.41"
                        }
                    ],
                    "healthStatus": "UNKNOWN",
                    "cpu": "99",
```

```
                    "memory": "100"
                }
            ],
            "cpu": "256",
            "createdAt": "2021-12-20T12:13:20.226000-05:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
            "group": "service:tdsevicetag",
            "healthStatus": "UNKNOWN",
            "lastStatus": "RUNNING",
            "launchType": "FARGATE",
            "memory": "512",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "web"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
            "platformVersion": "1.4.0",
            "platformFamily": "Linux",
            "pullStartedAt": "2021-12-20T12:13:44.611000-05:00",
            "pullStoppedAt": "2021-12-20T12:13:48.251000-05:00",
            "startedAt": "2021-12-20T12:13:49.326000-05:00",
            "startedBy": "ecs-svc/988401040018EXAMPLE",
            "tags": [],
            "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
d789e94343414c25b9f6bd59eEXAMPLE",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/webserver:2",
            "version": 3,
            "ephemeralStorage": {
                "sizeInGiB": 20
            }
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTasks。

execute-command

以下代码示例演示了如何使用 execute-command。

Amazon CLI

运行交互式 /bin/sh 命令

以下 execute-command 示例针对 ID 为 arn:aws:ecs:us-east-1:123456789012:task/MyCluster/d789e94343414c25b9f6bd59eEXAMPLE 的任务的名为 MyContainer 的容器运行交互式 /bin/sh 命令。

```
aws ecs execute-command \
    --cluster MyCluster \
    --task arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
d789e94343414c25b9f6bd59eEXAMPLE \
    --container MyContainer \
    --interactive \
    --command "/bin/sh"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用 Amazon ECS Exec 进行调试。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExecuteCommand。

get-task-protection

以下代码示例演示了如何使用 get-task-protection。

Amazon CLI

检索 ECS 服务中的任务的保护状态

以下 get-task-protection 提供属于 Amazon ECS 服务的 ECS 任务的保护状态。

```
aws ecs get-task-protection \
    --cluster ECS-project-update-cluster \
    --tasks c43ed3b1331041f289316f958adb6a24
```

输出：

```
{
    "protectedTasks": [
        {
            "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
c43ed3b1331041f289316f958adb6a24",
            "protectionEnabled": false
        }
    ],
    "failures": []
}
```

有关任务保护的更多信息，请参阅《Amazon ECS 开发人员指南》中的保护您的 Amazon ECS 任务不被横向缩减事件终止。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetTaskProtection。

`list-account-settings`

以下代码示例演示了如何使用 `list-account-settings`。

Amazon CLI

示例 1：查看账户的账户设置

以下 `list-account-settings` 示例显示了账户的有效账户设置。

```
aws ecs list-account-settings --effective-settings
```

输出：

```
{
    "settings": [
        {
            "name": "containerInstanceLongArnFormat",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:root"
        },
        {
            "name": "serviceLongArnFormat",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:root"
        },
```

```
        {
            "name": "taskLongArnFormat",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:root"
        }
    ]
}
```

示例 2：查看特定 IAM 用户或 IAM 角色的账户设置

以下 `list-account-settings` 示例显示指定 IAM 用户或 IAM 角色的账户设置。

```
aws ecs list-account-settings --principal-arn arn:aws:iam::123456789012:user/MyUser
```

输出：

```
{
    "settings": [
        {
            "name": "serviceLongArnFormat",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/MyUser"
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon 资源名称（ARN）和 ID。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccountSettings。

list-attributes

以下代码示例演示了如何使用 `list-attributes`。

Amazon CLI

列出包含特定属性的容器实例

以下示例列出了默认集群中具有 `stack=production` 属性的容器实例的属性。

```
aws ecs list-attributes \
```

```
    --target-type container-instance \
    --attribute-name stack \
    --attribute-value production \
    --cluster default
```

输出：

```
{
    "attributes": [
        {
            "name": "stack",
            "targetId": "arn:aws:ecs:us-west-2:130757420319:container-
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",
            "value": "production"
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 容器代理配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttributes。

list-clusters

以下代码示例演示了如何使用 list-clusters。

Amazon CLI

列出您的可用集群

以下 list-clusters 示例将列出所有可用的集群。

```
aws ecs list-clusters
```

输出：

```
{
    "clusterArns": [
        "arn:aws:ecs:us-west-2:123456789012:cluster/MyECSCluster1",
        "arn:aws:ecs:us-west-2:123456789012:cluster/AnotherECSCluster"
    ]
```

```
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListClusters。

`list-container-instances`

以下代码示例演示了如何使用 `list-container-instances`。

Amazon CLI

列出集群中的容器实例

以下 `list-container-instances` 示例列出了集群中的所有可用容器实例。

```
aws ecs list-container-instances --cluster MyCluster
```

输出：

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-west-2:123456789012:container-instance/MyCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "arn:aws:ecs:us-west-2:123456789012:container-instance/MyCluster/
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 容器实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContainerInstances。

`list-service-deployments`

以下代码示例演示了如何使用 `list-service-deployments`。

Amazon CLI

列出服务部署

以下 `list-service-deployments` 示例检索名为 `example-service` 的服务的服务部署。

```
aws ecs list-service-deployments \
    --service arn:aws:ecs:us-east-1:123456789012:service/example-cluster/example-service
```

输出：

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-east-1:123456789012:service-deployment/example-cluster/example-service/ejGvqq2ilnbKT9qj0vLJe",
            "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/example-cluster/example-service",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
            "startedAt": "2024-10-31T08:03:32.510000-04:00",
            "createdAt": "2024-10-31T08:03:30.917000-04:00",
            "finishedAt": "2024-10-31T08:05:04.527000-04:00",
            "targetServiceRevisionArn": "arn:aws:ecs:us-east-1:123456789012:service-revision/example-cluster/example-service/1485800978477494678",
            "status": "SUCCESSFUL"
        }
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用 Amazon ECS 服务部署查看服务历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceDeployments。

`list-services-by-namespace`

以下代码示例演示了如何使用 `list-services-by-namespace`。

Amazon CLI

列出命名空间中的服务

以下 `list-services-by-namespace` 示例列出了在您的默认区域中为指定命名空间配置的所有服务。

```
aws ecs list-services-by-namespace \
    --namespace service-connect
```

输出：

```
{
    "serviceArns": [
        "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService",
        "arn:aws:ecs:us-west-2:123456789012:service/tutorial/service-connect-nginx-service"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的服务连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServicesByNamespace。

`list-services`

以下代码示例演示了如何使用 `list-services`。

Amazon CLI

列出集群中的服务

以下 `list-services` 示例演示如何列出集群中运行的服务。

```
aws ecs list-services --cluster MyCluster
```

输出：

```
 {
    "serviceArns": [
        "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出了特定集群的标签。

```
aws ecs list-tags-for-resource \
    --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster
```

输出：

```
{
    "tags": [
        {
            "key": "key1",
            "value": "value1"
        },
        {
            "key": "key2",
            "value": "value2"
        },
        {
            "key": "key3",
            "value": "value3"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-task-definition-families

以下代码示例演示了如何使用 list-task-definition-families。

Amazon CLI

示例 1：列出注册作业定义系列

以下 list-task-definition-families 示例列出了所有注册作业定义系列。

```
aws ecs list-task-definition-families
```

输出：

```
{
    "families": [
        "node-js-app",
        "web-timer",
        "hpcc",
        "hpcc-c4-8xlarge"
    ]
}
```

示例 2：筛选注册作业定义系列

以下 list-task-definition-families 示例列出了以“hpcc”开头的作业定义修订。

```
aws ecs list-task-definition-families --family-prefix hpcc
```

输出：

```
{
    "families": [
        "hpcc",
        "hpcc-c4-8xlarge"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的作业定义参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTaskDefinitionFamilies。

list-task-definitions

以下代码示例演示了如何使用 list-task-definitions。

Amazon CLI

示例 1：列出注册作业定义

以下 list-task-definitions 示例列出了所有注册作业定义。

```
aws ecs list-task-definitions
```

输出：

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:us-west-2:123456789012:task-definition/sleep300:2",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/sleep360:1",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:3",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:4",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:5",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:6"
    ]
}
```

示例 2：列出系列中的注册作业定义

以下 list-task-definitions 示例列出了指定系列的作业定义修订。

```
aws ecs list-task-definitions --family-prefix wordpress
```

输出：

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:3",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:4",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:5",
        "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:6"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTaskDefinitions。

list-tasks

以下代码示例演示了如何使用 list-tasks。

Amazon CLI

示例 1：列出集群中的任务

以下 `list-tasks` 示例将列出集群中的所有任务。

```
aws ecs list-tasks --cluster default
```

输出：

```
{
    "taskArns": [
        "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-11111EXAMPLE",
        "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE"
    ]
}
```

示例 2：列出特定容器实例上的任务

以下 `list-tasks` 示例将使用容器实例 UUID 作为筛选器列出容器实例上的任务。

```
aws ecs list-tasks --cluster default --container-instance a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE
```

输出：

```
{
    "taskArns": [
        "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-44444EXAMPLE"
    ]
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon ECS 作业定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTasks。

put-account-setting-default

以下代码示例演示了如何使用 `put-account-setting-default`。

Amazon CLI

修改默认账户设置

以下 put-account-setting-default 示例修改您账户上所有 IAM 用户或角色的默认账户设置。这些更改将应用于整个 Amazon 账户，除非一个 IAM 用户或角色显式覆盖自己的这些设置。

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled
```

输出：

```
{
    "setting": {
        "name": "serviceLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的 Amazon 资源名称（ARN）和 ID。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAccountSettingDefault。

put-account-setting

以下代码示例演示了如何使用 put-account-setting。

Amazon CLI

修改 IAM 用户账户的账户设置

以下 put-account-setting 示例将您的 IAM 用户账户的 containerInsights 账户设置设为 enhanced。这将开启具有增强型可观测性的 Container Insights。

```
aws ecs put-account-setting \
    --name containerInsights \
    --value enhanced
```

输出：

```
{
```

```
    "setting": {
        "name": "containerInsights",
        "value": "enhanced",
        "principalArn": "arn:aws:iam::123456789012:user/johndoe",
        "type": "user"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的修改账户设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAccountSetting。

put-account-settings

以下代码示例演示了如何使用 put-account-settings。

Amazon CLI

修改 IAM 用户或 IAM 角色的账户设置

以下 put-account-setting 示例修改指定 IAM 用户或 IAM 角色的账户设置。

```
aws ecs put-account-setting \
    --name serviceLongArnFormat \
    --value enabled \
    --principal-arn arn:aws:iam::123456789012:user/MyUser
```

输出：

```
{
    "setting": {
        "name": "serviceLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:user/MyUser"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAccountSettings。

put-attributes

以下代码示例演示了如何使用 put-attributes。

Amazon CLI

创建属性并将其与 Amazon ECS 资源相关联

以下 put-attributes 将名称为 stack 和值为 production 的属性应用于容器实例。

```
aws ecs put-attributes \
    --attributes name=stack,value=production,targetId=arn:aws:ecs:us-
west-2:130757420319:container-instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34
```

输出：

```
{
    "attributes": [
        {
            "name": "stack",
            "targetId": "arn:aws:ecs:us-west-2:130757420319:container-
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",
            "value": "production"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAttributes。

put-cluster-capacity-providers

以下代码示例演示了如何使用 put-cluster-capacity-providers。

Amazon CLI

示例 1：将现有容量提供程序添加到集群

以下 put-cluster-capacity-providers 示例将现有容量提供程序添加到集群。使用 create-capacity-provider 命令创建容量提供程序。describe-clusters 命令用于描述与集群关联的当前容量提供程序和默认容量提供程序策略。将新的容量提供程序添加到集群时，除了要与集群相关联的新容量提供程序之外，还必须指定所有现有容量提供程序。您还必须指定要与集群相关联的默认容量提供程序策略。在此示例中，MyCluster 集群具有与之关联的 MyCapacityProvider1 容量提供程序，您希望添加 MyCapacityProvider2 容量提供程序并将其包含在默认容量提供程序策略中，以便任务在两个容量提供程序之间均匀分布。

```
aws ecs put-cluster-capacity-providers \
    --cluster MyCluster \
    --capacity-providers MyCapacityProvider1 MyCapacityProvider2 \
    --default-capacity-provider-
strategy capacityProvider=MyCapacityProvider1,weight=1 capacityProvider=MyCapacityProvider2,
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "enabled"
            }
        ],
        "capacityProviders": [
            "MyCapacityProvider1",
            "MyCapacityProvider2"
        ],
        "defaultCapacityProviderStrategy": [
            {
                "capacityProvider": "MyCapacityProvider1",
                "weight": 1,
                "base": 0
            },
            {
                "capacityProvider": "MyCapacityProvider2",
                "weight": 1,
                "base": 0
            }
        ],
        "attachments": [
           {
```

```
                "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
                "type": "as_policy",
                "status": "ACTIVE",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider1"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
                    }
                ]
            },
            {
                "id": "ae592060-2382-4663-9476-b015c685593c",
                "type": "as_policy",
                "status": "ACTIVE",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider2"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
                    }
                ]
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

示例 2：从集群中删除容量提供程序

以下 put-cluster-capacity-providers 示例从集群中删除容量提供程序。describe-clusters 命令用于描述与集群关联的当前容量提供程序。当从集群中删除容量提供程序时，必须指定要与集群保持关联的容量提供程序，以及要与集群相关联的默认容量提供程序策略。

在此示例中，集群具有与之关联的 MyCapacityProvider1 和 MyCapacityProvider2 容量提供程序，并且您希望删除 MyCapacityProvider2 容量提供程序，因此您在命令中仅指定 MyCapacityProvider1 以及更新的默认容量提供程序策略。

```
aws ecs put-cluster-capacity-providers \
    --cluster MyCluster \
    --capacity-providers MyCapacityProvider1 \
    --default-capacity-provider-
strategy capacityProvider=MyCapacityProvider1,weight=1,base=0
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "enabled"
            }
        ],
        "capacityProviders": [
            "MyCapacityProvider1"
        ],
        "defaultCapacityProviderStrategy": [
            "capacityProvider": "MyCapacityProvider1",
            "weight": 1,
            "base": 0
        ],
        "attachments": [
           {
                "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
                "type": "as_policy",
                "status": "ACTIVE",
                "details": [
```

```
                    {
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider1"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
                    }
                ]
            },
            {
                "id": "ae592060-2382-4663-9476-b015c685593c",
                "type": "as_policy",
                "status": "DELETING",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider2"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
                    }
                ]
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

示例 3：从集群中删除所有容量提供程序

以下 `put-cluster-capacity-providers` 示例将从集群中删除所有现有的容量提供程序。

```
aws ecs put-cluster-capacity-providers \
    --cluster MyCluster \
    --capacity-providers [] \
    --default-capacity-provider-strategy []
```

输出：


```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "enabled"
            }
        ],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": [],
        "attachments": [
           {
                "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
                "type": "as_policy",
                "status": "DELETING",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider1"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
                    }
                ]
            },
            {
                "id": "ae592060-2382-4663-9476-b015c685593c",
                "type": "as_policy",
                "status": "DELETING",
                "details": [
                    {
```

```
                        "name": "capacityProviderName",
                        "value": "MyCapacityProvider2"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
                    }
                ]
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的集群容量提供程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutClusterCapacityProviders。

register-task-definition

以下代码示例演示了如何使用 register-task-definition。

Amazon CLI

示例 1：使用 JSON 文件注册作业定义

以下 register-task-definition 示例将作业定义注册到指定系列。容器定义以 JSON 格式保存在指定的文件位置。

```
aws ecs register-task-definition \
    --cli-input-json file://<path_to_json_file>/sleep360.json
```

的内容sleep360.json：

```
{
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "busybox",
            "cpu": 10,
            "command": [
                "sleep",
```

```
                "360"
            ],
            "memory": 10,
            "essential": true
        }
    ],
    "family": "sleep360"
}
```

输出：

```
{
    "taskDefinition": {
        "status": "ACTIVE",
        "family": "sleep360",
        "placementConstraints": [],
        "compatibilities": [
                "EXTERNAL",
                "EC2"
        ],
        "volumes": [],
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
sleep360:1",
        "containerDefinitions": [
            {
                "environment": [],
                "name": "sleep",
                "mountPoints": [],
                "image": "busybox",
                "cpu": 10,
                "portMappings": [],
                "command": [
                    "sleep",
                    "360"
                ],
        "memory": 10,
        "essential": true,
        "volumesFrom": []
        }
    ],
        "revision": 1
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的示例作业定义。

示例 2：使用 JSON 字符串参数注册作业定义

以下 register-task-definition 示例使用作为带转义双引号的 JSON 字符串参数提供的容器定义注册作业定义。

```
aws ecs register-task-definition \
    --family sleep360 \
    --container-definitions "[{\"name\":\"sleep\",\"image\":\"busybox\",\"cpu\":10,
\"command\":[\"sleep\",\"360\"],\"memory\":10,\"essential\":true}]"
```

输出与上一个示例完全相同。

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的创建作业定义。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTaskDefinition。

run-task

以下代码示例演示了如何使用 run-task。

Amazon CLI

示例 1：在默认集群上运行任务

以下 run-task 示例在默认集群上运行任务并使用客户端令牌。

```
aws ecs run-task \
    --cluster default \
    --task-definition sleep360:1 \
    --client-token 550e8400-e29b-41d4-a716-446655440000
```

输出：

```
{
    "tasks": [
        {
            "attachments": [],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
                    "value": "x86_64"
```

```
                }
            ],
            "availabilityZone": "us-east-1b",
            "capacityProviderName": "example-capacity-provider",
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-
instance/default/bc4d2ec611d04bb7bb97e83ceEXAMPLE",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
default/d6f51cc5bbc94a47969c92035e9f66f8/75853d2d-711e-458a-8362-0f0aEXAMPLE",
                    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/
d6f51cc5bbc94a47969c9203EXAMPLE",
                    "name": "sleep",
                    "image": "busybox",
                    "lastStatus": "PENDING",
                    "networkInterfaces": [],
                    "cpu": "10",
                    "memory": "10"
                }
            ],
            "cpu": "10",
            "createdAt": "2023-11-21T16:59:34.403000-05:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
            "group": "family:sleep360",
            "lastStatus": "PENDING",
            "launchType": "EC2",
            "memory": "10",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "sleep"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
            "tags": [],
            "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/
d6f51cc5bbc94a47969c9203EXAMPLE",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/sleep360:1",
            "version": 1
        }
```

```
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将应用程序作为独立任务运行。

示例 2：为独立任务配置 Amazon EBS 卷

以下 run-task 示例为默认集群上的 Fargate 任务配置加密的 Amazon EBS 卷。您必须配置了一个附有 AmazonECSInfrastructureRolePolicyForVolumes 托管式策略的 Amazon ECS 基础设施角色。您必须使用与 run-task 请求中相同的卷名称来指定任务定义。此示例使用 --cli-input-json 选项和名为 ebs.json 的 JSON 输入文件。

```
aws ecs run-task \
    --cli-input-json file://ebs.json
```

的内容ebs.json：

```
{
   "cluster": "default",
   "taskDefinition": "mytaskdef",
   "launchType": "FARGATE",
   "networkConfiguration":{
        "awsvpcConfiguration":{
            "assignPublicIp": "ENABLED",
            "securityGroups": ["sg-12344321"],
            "subnets":["subnet-12344321"]
        }
    },
   "volumeConfigurations": [
        {
            "name": "myEBSVolume",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
    ]
```

```
}
```

输出：

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "ce868693-15ca-4083-91ac-f782f64000c9",
                    "type": "ElasticNetworkInterface",
                    "status": "PRECREATED",
                    "details": [
                        {
                        "name": "subnetId",
                        "value": "subnet-070982705451dad82"
                        }
                    ]
                },
                {
                    "id": "a17ed863-786c-4372-b5b3-b23e53f37877",
                    "type": "AmazonElasticBlockStorage",
                    "status": "CREATED",
                    "details": [
                        {
                            "name": "roleArn",
                            "value": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole"
                        },
                        {
                            "name": "volumeName",
                            "value": "myEBSVolume"
                        },
                        {
                            "name": "deleteOnTermination",
                            "value": "true"
                        }
                    ]
                }
            ],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
```

```
                    "value": "x86_64"
                }
            ],
            "availabilityZone": "us-west-2b",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-west-2:123456789012:container/
default/7f1fbd3629434cc4b82d72d2f09b67c9/e21962a2-f328-4699-98a3-5161ac2c186a",
                    "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
default/7f1fbd3629434cc4b82d72d2f09b67c9",
                    "name": "container-using-ebs",
                    "image": "amazonlinux:2",
                    "lastStatus": "PENDING",
                    "networkInterfaces": [],
                    "cpu": "0"
                }
            ],
            "cpu": "1024",
            "createdAt": "2025-01-23T10:29:46.650000-06:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
            "group": "family:mytaskdef",
            "lastStatus": "PROVISIONING",
            "launchType": "FARGATE",
            "memory": "3072",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "container-using-ebs"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
            "platformVersion": "1.4.0",
            "platformFamily": "Linux",
            "tags": [],
            "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
default/7f1fbd3629434cc4b82d72d2f09b67c9",
            "taskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:4",
            "version": 1,
            "ephemeralStorage": {
                "sizeInGiB": 20
```

```
            },
            "fargateEphemeralStorage": {
                "sizeInGiB": 20
            }
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将 Amazon EBS 卷与 Amazon ECS 结合使用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RunTask。

start-task

以下代码示例演示了如何使用 start-task。

Amazon CLI

示例 1：启动新任务

以下 start-task 示例在默认集群中的指定容器实例上使用 sleep360 任务定义的最新修订版启动任务。

```
aws ecs start-task \
    --task-definition sleep360 \
    --container-instances 765936fadbdd46b5991a4bd70c2a43d4
```

输出：

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
default/666fdccc2e2d4b6894dd422f4eeee8f8",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default",
            "taskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:task-
definition/sleep360:3",
            "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/default/765936fadbdd46b5991a4bd70c2a43d4",
            "overrides": {
                "containerOverrides": [
```

```
                    {
                        "name": "sleep"
                    }
                ]
            },
            "lastStatus": "PENDING",
            "desiredStatus": "RUNNING",
            "cpu": "128",
            "memory": "128",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-
west-2:123456789012:container/75f11ed4-8a3d-4f26-a33b-ad1db9e02d41",
                    "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
default/666fdccc2e2d4b6894dd422f4eeee8f8",
                    "name": "sleep",
                    "lastStatus": "PENDING",
                    "networkInterfaces": [],
                    "cpu": "10",
                    "memory": "10"
                }
            ],
            "version": 1,
            "createdAt": 1563421494.186,
            "group": "family:sleep360",
            "launchType": "EC2",
            "attachments": [],
            "tags": []
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的在 Amazon ECS 上计划您的容器。

示例 2：在任务启动时配置 Amazon EBS 卷

以下 `start-task` 示例为指定容器实例上的任务配置加密的 Amazon EBS 卷。您必须配置了一个附有 `AmazonECSInfrastructureRolePolicyForVolumes` 托管式策略的 Amazon ECS 基础设施角色。您必须使用与 `start-task` 请求中相同的卷名称来指定任务定义。此示例使用 `--cli-input-json` 选项和名为 `ebs.json` 的 JSON 输入文件，该文件包含以下内容。

```
aws ecs start-task \
```

```
    --cli-input-json file://ebs.json \
    --container-instances 765936fadbdd46b5991a4bd70c2a43d4
```

的内容ebs.json：

```
{
   "cluster": "default",
   "taskDefinition": "mytaskdef",
   "networkConfiguration":{
        "awsvpcConfiguration":{
            "assignPublicIp": "ENABLED",
            "securityGroups": ["sg-12344321"],
            "subnets":["subnet-12344321"]
        }
    },
   "volumeConfigurations": [
        {
            "name": "myEBSVolume",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::123456789012:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
    ]
}
```

输出：

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "aea29489-9dcd-49f1-8164-4d91566e1113",
                    "type": "ElasticNetworkInterface",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "subnetId",
```

```
                            "value": "subnet-12344321"
                        }
                    ]
                },
                {
                    "id": "f29e1222-9a1e-410f-b499-a12a7cd6d42e",
                    "type": "AmazonElasticBlockStorage",
                    "status": "CREATED",
                    "details": [
                        {
                            "name": "roleArn",
                            "value": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole"
                        },
                        {
                            "name": "volumeName",
                            "value": "myEBSVolume"
                        },
                        {
                            "name": "deleteOnTermination",
                            "value": "true"
                        }
                    ]
                }
            ],
            "attributes": [
                {
                    "name": "ecs.cpu-architecture",
                    "value": "arm64"
                }
            ],
            "availabilityZone": "us-west-2c",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default",
            "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/default/765936fadbdd46b5991a4bd70c2a43d4",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-west-2:123456789012:container/
default/bb122ace3ed84add92c00a351a03c69e/a4a9ed10-51c7-4567-9653-50e71b94f867",
                    "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/default/
bb122ace3ed84add92c00a351a03c69e",
                    "name": "container-using-ebs",
                    "image": "amazonlinux:2",
                    "lastStatus": "PENDING",
```

```
                    "networkInterfaces": [],
                    "cpu": "0"
                }
            ],
            "cpu": "1024",
            "createdAt": "2025-01-23T14:51:05.191000-06:00",
            "desiredStatus": "RUNNING",
            "enableExecuteCommand": false,
            "group": "family:mytaskdef",
            "lastStatus": "PROVISIONING",
            "launchType": "EC2",
            "memory": "3072",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "container-using-ebs"
                    }
                ],
                "inferenceAcceleratorOverrides": []
            },
             "tags": [],
            "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/default/
bb122ace3ed84add92c00a351a03c69e",
            "taskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:4",
            "version": 1
        }
    ],
    "failures": []
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将 Amazon EBS 卷与 Amazon ECS 结合使用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartTask。

stop-task

以下代码示例演示了如何使用 stop-task。

Amazon CLI

停止任务

以下 stop-task 将停止指定任务在默认集群中运行。

```
aws ecs stop-task \
    --task 666fdccc2e2d4b6894dd422f4eeee8f8
```

输出：

```
{
    "task": {
        "taskArn": "arn:aws:ecs:us-west-2:130757420319:task/
default/666fdccc2e2d4b6894dd422f4eeee8f8",
        "clusterArn": "arn:aws:ecs:us-west-2:130757420319:cluster/default",
        "taskDefinitionArn": "arn:aws:ecs:us-west-2:130757420319:task-definition/
sleep360:3",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:130757420319:container-
instance/default/765936fadbdd46b5991a4bd70c2a43d4",
        "overrides": {
            "containerOverrides": []
        },
        "lastStatus": "STOPPED",
        "desiredStatus": "STOPPED",
        "cpu": "128",
        "memory": "128",
        "containers": [],
        "version": 2,
        "stoppedReason": "Taskfailedtostart",
        "stopCode": "TaskFailedToStart",
        "connectivity": "CONNECTED",
        "connectivityAt": 1563421494.186,
        "pullStartedAt": 1563421494.252,
        "pullStoppedAt": 1563421496.252,
        "executionStoppedAt": 1563421497,
        "createdAt": 1563421494.186,
        "stoppingAt": 1563421497.252,
        "stoppedAt": 1563421497.252,
        "group": "family:sleep360",
        "launchType": "EC2",
        "attachments": [],
        "tags": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopTask。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例将单个标签添加到指定资源。

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster
    --tags key=key1,value=value1
```

此命令不生成任何输出。

将多个标签添加到资源

以下 tag-resource 示例将多个标签添加到指定资源。

```
aws ecs tag-resource \
--resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \
--tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从指定资源中删除列出的标签。

```
aws ecs untag-resource \
    --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \
    --tag-keys key1,key2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-cluster-settings

以下代码示例演示了如何使用 update-cluster-settings。

Amazon CLI

修改集群的设置

以下 update-cluster-settings 示例为 MyCluster 集群启用具有增强型可观测性的 CloudWatch Container Insights。

```
aws ecs update-cluster-settings \
    --cluster MyCluster \
    --settings name=containerInsights,value=enhanced
```

输出：

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-esat-1:123456789012:cluster/MyCluster",
        "clusterName": "default",
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "enhanced"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的修改账户设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateClusterSettings。

update-cluster

以下代码示例演示了如何使用 update-cluster。

Amazon CLI

示例 1：更新启用 containerInsights 的 ECS 集群

以下 update-cluster 将已创建的集群中的 containerInsights 值更新为 enabled。默认已禁用。

```
aws ecs update-cluster \
    --cluster ECS-project-update-cluster \
    --settings name=containerInsights,value=enabled
```

输出：

```
"cluster": {
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/ECS-project-update-
cluster",
    "clusterName": "ECS-project-update-cluster",
    "status": "ACTIVE",
    "registeredContainerInstancesCount": 0,
    "runningTasksCount": 0,
    "pendingTasksCount": 0,
    "activeServicesCount": 0,
    "statistics": [],
    "tags": [],
    "settings": [
        {
            "name": "containerInsights",
            "value": "enabled"
        }
    ],
    "capacityProviders": [
        "Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt"
    ],
    "defaultCapacityProviderStrategy": [
        {
            "capacityProvider": "Infra-ECS-Cluster-ECS-project-update-cluster-
d6bb6d5b-EC2CapacityProvider-3fIpdkLywwFt",
            "weight": 1,
```

```
            "base": 0
        }
    ],
    "attachments": [
        {
            "id": "069d002b-7634-42e4-b1d4-544f4c8f6380",
            "type": "as_policy",
            "status": "CREATED",
            "details": [
                {
                    "name": "capacityProviderName",
                    "value": "Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt"
                },
                {
                    "name": "scalingPolicyName",
                    "value": "ECSManagedAutoScalingPolicy-152363a6-8c65-484c-
b721-42c3e070ae93"
                }
            ]
        },
        {
            "id": "08b5b6ca-45e9-4209-a65d-e962a27c490a",
            "type": "managed_draining",
            "status": "CREATED",
            "details": [
                {
                    "name": "capacityProviderName",
                    "value": "Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt"
                },
                {
                    "name": "autoScalingLifecycleHookName",
                    "value": "ecs-managed-draining-termination-hook"
                }
            ]
        },
        {
            "id": "45d0b36f-8cff-46b6-9380-1288744802ab",
            "type": "sc",
            "status": "ATTACHED",
            "details": []
        }
    ],
```

```
    "attachmentsStatus": "UPDATE_COMPLETE",
    "serviceConnectDefaults": {
        "namespace": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
igwrsylmy3kwvcdx"
    }
}
```

示例 2：更新 ECS 集群以设置默认 Service Connect 命名空间

以下 `update-cluster` 通过设置默认 Service Connect 命名空间来更新 ECS 集群。

```
aws ecs update-cluster \
    --cluster ECS-project-update-cluster \
    --service-connect-defaults namespace=test
```

输出：

```
 {
     "cluster": {
         "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/ECS-project-
update-cluster",
         "clusterName": "ECS-project-update-cluster",
         "status": "ACTIVE",
         "registeredContainerInstancesCount": 0,
         "runningTasksCount": 0,
         "pendingTasksCount": 0,
         "activeServicesCount": 0,
         "statistics": [],
         "tags": [],
         "settings": [
             {
                 "name": "containerInsights",
                 "value": "enabled"
             }
         ],
         "capacityProviders": [
             "Infra-ECS-Cluster-ECS-project-update-cluster-d6bb6d5b-
EC2CapacityProvider-3fIpdkLywwFt"
         ],
         "defaultCapacityProviderStrategy": [
             {
                 "capacityProvider": "Infra-ECS-Cluster-ECS-project-update-cluster-
d6bb6d5b-EC2CapacityProvider-3fIpdkLywwFt",
```

```
                "weight": 1,
                "base": 0
            }
        ],
        "attachments": [
            {
                "id": "069d002b-7634-42e4-b1d4-544f4c8f6380",
                "type": "as_policy",
                "status": "CREATED",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "Infra-ECS-Cluster-ECS-project-update-cluster-
d6bb6d5b-EC2CapacityProvider-3fIpdkLywwFt"
                    },
                    {
                        "name": "scalingPolicyName",
                        "value": "ECSManagedAutoScalingPolicy-152363a6-8c65-484c-
b721-42c3e070ae93"
                    }
                ]
            },
            {
                "id": "08b5b6ca-45e9-4209-a65d-e962a27c490a",
                "type": "managed_draining",
                "status": "CREATED",
                "details": [
                    {
                        "name": "capacityProviderName",
                        "value": "Infra-ECS-Cluster-ECS-project-update-cluster-
d6bb6d5b-EC2CapacityProvider-3fIpdkLywwFt"
                    },
                    {
                        "name": "autoScalingLifecycleHookName",
                        "value": "ecs-managed-draining-termination-hook"
                    }
                ]
            },
            {
                "id": "45d0b36f-8cff-46b6-9380-1288744802ab",
                "type": "sc",
                "status": "DELETED",
                "details": []
            },
```

```
            {
                "id": "3e6890c3-609c-4832-91de-d6ca891b3ef1",
                "type": "sc",
                "status": "ATTACHED",
                "details": []
            },
            {
                "id": "961b8ec1-c2f1-4070-8495-e669b7668e90",
                "type": "sc",
                "status": "DELETED",
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_COMPLETE",
        "serviceConnectDefaults": {
            "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-dtjmxqpfi46ht7dr"
        }
    }
}
```

有关 Service Connect 的更多信息，请参阅《Amazon ECS 开发人员指南》中的使用 Service Connect 连接具有短名称的 Amazon ECS 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCluster。

update-container-agent

以下代码示例演示了如何使用 update-container-agent。

Amazon CLI

更新 Amazon ECS 容器实例上的容器代理

以下 update-container-agent 示例更新默认集群中指定容器实例上的容器代理。

```
aws ecs update-container-agent --cluster default --container-
instance a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "containerInstance": {
```

```
        "status": "ACTIVE",
...
        "agentUpdateStatus": "PENDING",
        "versionInfo": {
            "agentVersion": "1.0.0",
            "agentHash": "4023248",
            "dockerVersion": "DockerVersion: 1.5.0"
        }
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的更新 Amazon ECS 容器代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContainerAgent。

update-container-instances-state

以下代码示例演示了如何使用 update-container-instances-state。

Amazon CLI

更新容器实例的状态

以下 update-container-instances-state 将指定容器实例的状态更新为 DRAINING，这样将把它从其注册到的集群中删除。

```
aws ecs update-container-instances-state \
    --container-instances 765936fadbdd46b5991a4bd70c2a43d4 \
    --status DRAINING
```

输出：

```
{
    "containerInstances": [
        {
            "containerInstanceArn": "arn:aws:ecs:us-west-2:130757420319:container-
instance/default/765936fadbdd46b5991a4bd70c2a43d4",
            "ec2InstanceId": "i-013d87ffbb4d513bf",
            "version": 4390,
            "versionInfo": {
                "agentVersion": "1.29.0",
                "agentHash": "a190a73f",
                "dockerVersion": "DockerVersion:18.06.1-ce"
```

```
            },
            "remainingResources": [
                {
                    "name": "CPU",
                    "type": "INTEGER",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 1536
                },
                {
                    "name": "MEMORY",
                    "type": "INTEGER",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 2681
                },
                {
                    "name": "PORTS",
                    "type": "STRINGSET",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678",
                        "51679"
                    ]
                },
                {
                    "name": "PORTS_UDP",
                    "type": "STRINGSET",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 0,
                    "stringSetValue": []
                }
            ],
            "registeredResources": [
                {
                    "name": "CPU",
                    "type": "INTEGER",
                    "doubleValue": 0,
```

```
                    "longValue": 0,
                    "integerValue": 2048
                },
                {
                    "name": "MEMORY",
                    "type": "INTEGER",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 3705
                },
                {
                    "name": "PORTS",
                    "type": "STRINGSET",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678",
                        "51679"
                    ]
                },
                {
                    "name": "PORTS_UDP",
                    "type": "STRINGSET",
                    "doubleValue": 0,
                    "longValue": 0,
                    "integerValue": 0,
                    "stringSetValue": []
                }
            ],
            "status": "DRAINING",
            "agentConnected": true,
            "runningTasksCount": 2,
            "pendingTasksCount": 0,
            "attributes": [
                {
                    "name": "ecs.capability.secrets.asm.environment-variables"
                },
                {
                    "name": "ecs.capability.branch-cni-plugin-version",
                    "value": "e0703516-"
```

```
        },
        {
            "name": "ecs.ami-id",
            "value": "ami-00e0090ac21971297"
        },
        {
            "name": "ecs.capability.secrets.asm.bootstrap.log-driver"
        },
        {
            "name": "com.amazonaws.ecs.capability.logging-driver.none"
        },
        {
            "name": "ecs.capability.ecr-endpoint"
        },
        {
            "name": "ecs.capability.docker-plugin.local"
        },
        {
            "name": "ecs.capability.task-cpu-mem-limit"
        },
        {
            "name": "ecs.capability.secrets.ssm.bootstrap.log-driver"
        },
        {
            "name": "com.amazonaws.ecs.capability.docker-remote-api.1.30"
        },
        {
            "name": "com.amazonaws.ecs.capability.docker-remote-api.1.31"
        },
        {
            "name": "com.amazonaws.ecs.capability.docker-remote-api.1.32"
        },
        {
            "name": "ecs.availability-zone",
            "value": "us-west-2c"
        },
        {
            "name": "ecs.capability.aws-appmesh"
        },
        {
            "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
        },
        {
            "name": "com.amazonaws.ecs.capability.docker-remote-api.1.24"
```

```
                },
                {
                    "name": "ecs.capability.task-eni-trunking"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.25"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.26"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.27"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.28"
                },
                {
                    "name": "com.amazonaws.ecs.capability.privileged-container"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.29"
                },
                {
                    "name": "ecs.cpu-architecture",
                    "value": "x86_64"
                },
                {
                    "name": "com.amazonaws.ecs.capability.ecr-auth"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
                },
                {
                    "name": "ecs.os-type",
                    "value": "linux"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
                },
                {
                    "name": "ecs.capability.task-eia"
```

```
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
                },
                {
                    "name": "ecs.capability.private-registry-
authentication.secretsmanager"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
                },
                {
                    "name": "ecs.capability.execution-role-awslogs"
                },
                {
                    "name": "ecs.vpc-id",
                    "value": "vpc-1234"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
                },
                {
                    "name": "ecs.capability.task-eni"
                },
                {
                    "name": "ecs.capability.execution-role-ecr-pull"
                },
                {
                    "name": "ecs.capability.container-health-check"
                },
                {
                    "name": "ecs.subnet-id",
                    "value": "subnet-1234"
                },
                {
```

```
                    "name": "ecs.instance-type",
                    "value": "c5.large"
                },
                {
                    "name": "com.amazonaws.ecs.capability.task-iam-role-network-
host"
                },
                {
                    "name": "ecs.capability.container-ordering"
                },
                {
                    "name": "ecs.capability.cni-plugin-version",
                    "value": "91ccefc8-2019.06.0"
                },
                {
                    "name": "ecs.capability.pid-ipc-namespace-sharing"
                },
                {
                    "name": "ecs.capability.secrets.ssm.environment-variables"
                },
                {
                    "name": "com.amazonaws.ecs.capability.task-iam-role"
                }
            ],
            "registeredAt": 1560788724.507,
            "attachments": [],
            "tags": []
        }
    ],
    "failures": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContainerInstancesState。

update-service-primary-task-set

以下代码示例演示了如何使用 update-service-primary-task-set。

Amazon CLI

更新服务的主任务集

以下 update-service-primary-task-set 示例更新指定服务的主任务集。

```
aws ecs update-service-primary-task-set \
    --cluster MyCluster \
    --service MyService \
    --primary-task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789
```

输出：

```
{
    "taskSet": {
        "id": "ecs-svc/1234567890123456789",
        "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789",
        "status": "PRIMARY",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sample-fargate:2",
        "computedDesiredCount": 1,
        "pendingCount": 0,
        "runningCount": 0,
        "createdAt": 1557128360.711,
        "updatedAt": 1557129412.653,
        "launchType": "EC2",
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344312"
                ],
                "assignPublicIp": "DISABLED"
            }
        },
        "loadBalancers": [],
        "serviceRegistries": [],
        "scale": {
            "value": 50.0,
            "unit": "PERCENT"
        },
        "stabilityStatus": "STABILIZING",
        "stabilityStatusAt": 1557129279.914
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServicePrimaryTaskSet。

update-service

以下代码示例演示了如何使用 update-service。

Amazon CLI

示例 1：更改服务中使用的作业定义

以下 update-service 示例将 my-http-service 服务更新为使用 amazon-ecs-sample 作业定义。

```
aws ecs update-service \
    --cluster test \
    --service my-http-service \
    --task-definition amazon-ecs-sample
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/test/my-http-
service",
        "serviceName": "my-http-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/test",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 2,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "1.4.0",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
amazon-ecs-sample:2",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": true,
```

```
                "rollback": true
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 100,
            "alarms": {
                "alarmNames": [],
                "rollback": false,
                "enable": false
            }
        },
        "deployments": [
            {
                "id": "ecs-svc/7419115625193919142",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/amazon-ecs-sample:2",
                "desiredCount": 0,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": "2025-02-21T13:26:02.734000-06:00",
                "updatedAt": "2025-02-21T13:26:02.734000-06:00",
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-svc/7419115625193919142 in
 progress."
            },
            {
                "id": "ecs-svc/1709597507655421668",
                "status": "ACTIVE",
```

```
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/old-amazon-ecs-sample:4",
                "desiredCount": 2,
                "pendingCount": 0,
                "runningCount": 2,
                "failedTasks": 0,
                "createdAt": "2025-01-24T11:13:07.621000-06:00",
                "updatedAt": "2025-02-02T16:11:30.838000-06:00",
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                             "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "COMPLETED",
                "rolloutStateReason": "ECS deployment ecs-svc/1709597507655421668
 completed."
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [
            {
                "id": "e40b4d1c-80d9-4834-aaf3-6a268e530e17",
                "createdAt": "2025-02-21T10:31:26.037000-06:00",
                "message": "(my-http-service) has reached a steady state."
            },
            {
                "id": "6ac069ad-fc8b-4e49-a35d-b5574a964c8e",
                "createdAt": "2025-02-21T04:31:22.703000-06:00",
                "message": "(my-http-service) has reached a steady state."
            },
            {
                "id": "265f7d37-dfd1-4880-a846-ec486f341919",
                "createdAt": "2025-02-20T22:31:22.514000-06:00",
                "message": "(my-http-service) has reached a steady state."
```

```
            }
        ],
        "createdAt": "2024-10-30T17:12:43.218000-05:00",
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321",
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:role/AIDACKCEVSQ6C2EXAMPLE",
        "enableECSManagedTags": true,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
        "availabilityZoneRebalancing": "DISABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用控制台更新 Amazon ECS 服务。

示例 2：更改服务中的任务数

以下 `update-service` 示例将服务 `my-http-service` 所需的任务计数更新为 2。

```
aws ecs update-service \
    --cluster MyCluster \
    --service my-http-service \
    --desired-count 2
```

输出：

```
{
```

```
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/my-http-
service",
        "serviceName": "my-http-service",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 1,
        "pendingCount": 0,
        "capacityProviderStrategy": [
            {
                "capacityProvider": "FARGATE",
                "weight": 1,
                "base": 0
            }
        ],
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-east-1:123456789012:task-definition/
MyTaskDefinition",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": true,
                "rollback": true
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 100,
            "alarms": {
                "alarmNames": [],
                "rollback": false,
                "enable": false
            }
        },
        "deployments": [
            {
                "id": "ecs-svc/1976744184940610707",
                "status": "PRIMARY",
                "taskkDefinition": "arn:aws:ecs:us-east-1:123456789012:task-
definition/MyTaskDefinition",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 1,
```

```
                "failedTasks": 0,
                "createdAt": "2024-12-03T16:24:25.225000-05:00",
                "updatedAt": "2024-12-03T16:25:15.837000-05:00",
                "capacityProviderStrategy": [
                    {
                        "capacityProvider": "FARGATE",
                        "weight": 1,
                        "base": 0
                    }
                ],
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-0d0eab1bb38d5ca64",
                            "subnet-0db5010045995c2d5"
                        ],
                        "securityGroups": [
                            "sg-02556bf85a191f59a"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "COMPLETED",
                "rolloutStateReason": "ECS deployment ecs-svc/1976744184940610707
 completed."
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [
            {
                "id": "f27350b9-4b2a-4e2e-b72e-a4b68380de45",
                "createdAt": "2024-12-30T13:24:07.345000-05:00",
                "message": "(service my-http-service) has reached a steady state."
            },
            {
                "id": "e764ec63-f53f-45e3-9af2-d99f922d2957",
                "createdAt": "2024-12-30T12:32:21.600000-05:00",
                "message": "(service my-http-service) has reached a steady state."
            },
            {
                "id": "28444756-c2fa-47f8-bd60-93a8e05f3991",
```

```
                "createdAt": "2024-12-08T19:26:10.367000-05:00",
                "message": "(service my-http-service) has reached a steady state."
            }
        ],
        "createdAt": "2024-12-03T16:24:25.225000-05:00",
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-0d0eab1bb38d5ca64",
                    "subnet-0db5010045995c2d5"
                ],
                "securityGroups": [
                    "sg-02556bf85a191f59a"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:role/Admin",
        "enableECSManagedTags": true,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
        "availabilityZoneRebalancing": "ENABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用控制台更新 Amazon ECS 服务。

示例 3：配置 Amazon EBS 卷以便在服务更新时附加

以下 `update-service` 示例将服务 `my-http-service` 更新为使用 Amazon EBS 卷。您必须配置了一个附有 `AmazonECSInfrastructureRolePolicyForVolumes` 托管式策略的 Amazon ECS 基础设施角色。您还必须指定一个任务定义，其卷名称与 `update-service` 请求中的卷名称相同，并且 `configuredAtLaunch` 设置为 `true`。此示例使用 `--cli-input-json` 选项和名为 `ebs.json` 的 JSON 输入文件。

```
aws ecs update-service \
```

```
    --cli-input-json file://ebs.json
```

的内容ebs.json：

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "my-http-service",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
                "roleArn":"arn:aws:iam::123456789012:role/ecsInfrastructureRole",
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/mycluster/my-http-
service",
        "serviceName": "my-http-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/mycluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 2,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
mytaskdef:1",
```

```
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": true,
                "rollback": true
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 100,
            "alarms": {
                "alarmNames": [],
                "rollback": false,
                "enable": false
            }
        },
        "deployments": [
            {
                "id": "ecs-svc/2420458347226626275",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:1",
                "desiredCount": 0,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": "2025-02-21T15:07:20.519000-06:00",
                "updatedAt": "2025-02-21T15:07:20.519000-06:00",
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321",
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-svc/2420458347226626275 in
 progress.",
                "volumeConfigurations": [
                    {
```

```
                        "name": "ebs-volume",
                        "managedEBSVolume": {
                            "volumeType": "gp3",
                            "sizeInGiB": 100,
                            "iops": 3000,
                            "throughput": 125,
                            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
                            "filesystemType": "ext4"
                        }
                    }
                ]
            },
            {
                "id": "ecs-svc/5191625155316533644",
                "status": "ACTIVE",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:2",
                "desiredCount": 2,
                "pendingCount": 0,
                "runningCount": 2,
                "failedTasks": 0,
                "createdAt": "2025-02-21T14:54:48.862000-06:00",
                "updatedAt": "2025-02-21T14:57:22.502000-06:00",
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "COMPLETED",
                "rolloutStateReason": "ECS deployment ecs-svc/5191625155316533644
 completed."
            }
        ],
```

```
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [
            {
                "id": "b5823113-c2c5-458e-9649-8c2ed38f23a5",
                "createdAt": "2025-02-21T14:57:22.508000-06:00",
                "message": "(service my-http-service) has reached a steady state."
            },
            {
                "id": "b05a48e8-da35-4074-80aa-37ceb3167357",
                "createdAt": "2025-02-21T14:57:22.507000-06:00",
                "message": "(service my-http-service) (deployment ecs-
svc/5191625155316533644) deployment completed."
            },
            {
                "id": "a10cd55d-4ba6-4cea-a655-5a5d32ada8a0",
                "createdAt": "2025-02-21T14:55:32.833000-06:00",
                "message": "(service my-http-service) has started 1 tasks: (task
 fb9c8df512684aec92f3c57dc3f22361)."
            },
        ],
        "createdAt": "2025-02-21T14:54:48.862000-06:00",
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:role/AIDACKCEVSQ6C2EXAMPLE",
        "enableECSManagedTags": true,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
```

```
        "availabilityZoneRebalancing": "ENABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将 Amazon EBS 卷与 Amazon ECS 结合使用。

示例 4：更新服务以不再使用 Amazon EBS 卷

以下 `update-service` 示例将服务 `my-http-service` 更新为不再使用 Amazon EBS 卷。您必须指定 `configuredAtLaunch` 设置为 `false` 的任务定义修订版。

```
aws ecs update-service \
    --cluster mycluster \
    --task-definition mytaskdef \
    --service my-http-service \
    --desired-count 2 \
    --volume-configurations "[]"
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/mycluster/my-http-
service",
        "serviceName": "my-http-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/mycluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 2,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
mytaskdef:3",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": true,
                "rollback": true
            },
```

```
            "maximumPercent": 200,
            "minimumHealthyPercent": 100,
            "alarms": {
                "alarmNames": [],
                "rollback": false,
                "enable": false
            }
        },
        "deployments": [
            {
                "id": "ecs-svc/7522791612543716777",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/mytaskdef:3",
                "desiredCount": 0,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": "2025-02-21T15:25:38.598000-06:00",
                "updatedAt": "2025-02-21T15:25:38.598000-06:00",
                    "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-svc/7522791612543716777 in
 progress."
            },
            {
                "id": "ecs-svc/2420458347226626275",
                "status": "ACTIVE",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/myoldtaskdef:1",
                "desiredCount": 2,
```

```
                "pendingCount": 0,
                "runningCount": 2,
                "failedTasks": 0,
                "createdAt": "2025-02-21T15:07:20.519000-06:00",
                "updatedAt": "2025-02-21T15:10:59.955000-06:00",
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-12344321"
                        ],
                        "securityGroups": [
                            "sg-12344321"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "COMPLETED",
                "rolloutStateReason": "ECS deployment ecs-svc/2420458347226626275
 completed.",
                "volumeConfigurations": [
                    {
                        "name": "ebs-volume",
                        "managedEBSVolume": {
                            "volumeType": "gp3",
                            "sizeInGiB": 100,
                            "iops": 3000,
                            "throughput": 125,
                            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
                            "filesystemType": "ext4"
                        }
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [
            {
                "id": "4f2c3ca1-7800-4048-ba57-bba210ada2ad",
                "createdAt": "2025-02-21T15:10:59.959000-06:00",
```

```
                "message": "(service my-http-service) has reached a steady state."
            },
            {
                "id": "4b36a593-2d40-4ed6-8be8-b9b699eb6198",
                "createdAt": "2025-02-21T15:10:59.958000-06:00",
                "message": "(service my-http-service) (deployment ecs-
svc/2420458347226626275) deployment completed."
            },
            {
                "id": "88380089-14e2-4ef0-8dbb-a33991683371",
                "createdAt": "2025-02-21T15:09:39.055000-06:00",
                "message": "(service my-http-service) has stopped 1 running tasks:
 (task fb9c8df512684aec92f3c57dc3f22361)."
            },
            {
                "id": "97d84243-d52f-4255-89bb-9311391c61f6",
                "createdAt": "2025-02-21T15:08:57.653000-06:00",
                "message": "(service my-http-service) has stopped 1 running tasks:
 (task 33eff090ad2c40539daa837e6503a9bc)."
            },
            {
                "id": "672ece6c-e2d0-4021-b5da-eefb14001687",
                "createdAt": "2025-02-21T15:08:15.631000-06:00",
                "message": "(service my-http-service) has started 1 tasks: (task
 996c02a66ff24f3190a4a8e0c841740f)."
            },
            {
                "id": "a3cf9bea-9be6-4175-ac28-4c68360986eb",
                "createdAt": "2025-02-21T15:07:36.931000-06:00",
                "message": "(service my-http-service) has started 1 tasks: (task
 d5d23c39f89e46cf9a647b9cc6572feb)."
            },
            {
                "id": "b5823113-c2c5-458e-9649-8c2ed38f23a5",
                "createdAt": "2025-02-21T14:57:22.508000-06:00",
                "message": "(service my-http-service) has reached a steady state."
            },
            {
                "id": "b05a48e8-da35-4074-80aa-37ceb3167357",
                "createdAt": "2025-02-21T14:57:22.507000-06:00",
                "message": "(service my-http-service) (deployment ecs-
svc/5191625155316533644) deployment completed."
            },
            {
```

```
                "id": "a10cd55d-4ba6-4cea-a655-5a5d32ada8a0",
                "createdAt": "2025-02-21T14:55:32.833000-06:00",
                "message": "(service my-http-service) has started 1 tasks: (task
 fb9c8df512684aec92f3c57dc3f22361)."
            },
            {
                "id": "42da91fa-e26d-42ef-88c3-bb5965c56b2f",
                "createdAt": "2025-02-21T14:55:02.703000-06:00",
                "message": "(service my-http-service) has started 1 tasks: (task
 33eff090ad2c40539daa837e6503a9bc)."
            }
        ],
        "createdAt": "2025-02-21T14:54:48.862000-06:00",
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:role/AIDACKCEVSQ6C2EXAMPLE",
        "enableECSManagedTags": true,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
        "availabilityZoneRebalancing": "ENABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的将 Amazon EBS 卷与 Amazon ECS 结合使用。

示例 5：为服务开启可用区重新平衡

以下 update-service 示例为服务 my-http-service 开启可用区重新平衡。

```
aws ecs update-service \
    --cluster MyCluster \
    --service my-http-service \
    --availability-zone-rebalancing ENABLED
```

输出：

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/my-http-
service",
        "serviceName": "my-http-service",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 2,
        "runningCount": 1,
        "pendingCount": 0,
        "capacityProviderStrategy": [
            {
                "capacityProvider": "FARGATE",
                "weight": 1,
                "base": 0
            }
        ],
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-east-1:123456789012:task-definition/
MyTaskDefinition",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": true,
                "rollback": true
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 100,
            "alarms": {
                "alarmNames": [],
                "rollback": false,
                "enable": false
```

```
            }
        },
        "deployments": [
            {
                "id": "ecs-svc/1976744184940610707",
                "status": "PRIMARY",
                "taskkDefinition": "arn:aws:ecs:us-east-1:123456789012:task-
definition/MyTaskDefinition",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 1,
                "failedTasks": 0,
                "createdAt": "2024-12-03T16:24:25.225000-05:00",
                "updatedAt": "2024-12-03T16:25:15.837000-05:00",
                "capacityProviderStrategy": [
                    {
                        "capacityProvider": "FARGATE",
                        "weight": 1,
                        "base": 0
                    }
                ],
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-0d0eab1bb38d5ca64",
                            "subnet-0db5010045995c2d5"
                        ],
                        "securityGroups": [
                            "sg-02556bf85a191f59a"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "rolloutState": "COMPLETED",
                "rolloutStateReason": "ECS deployment ecs-svc/1976744184940610707
 completed."
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "events": [],
        "createdAt": "2024-12-03T16:24:25.225000-05:00",
```

```
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
                    "subnet-0d0eab1bb38d5ca64",
                    "subnet-0db5010045995c2d5"
                ],
                "securityGroups": [
                    "sg-02556bf85a191f59a"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "healthCheckGracePeriodSeconds": 0,
        "schedulingStrategy": "REPLICA",
        "deploymentController": {
            "type": "ECS"
        },
        "createdBy": "arn:aws:iam::123456789012:role/Admin",
        "enableECSManagedTags": true,
        "propagateTags": "NONE",
        "enableExecuteCommand": false,
        "availabilityZoneRebalancing": "ENABLED"
    }
}
```

有关更多信息，请参阅《Amazon ECS 开发人员指南》中的使用控制台更新 Amazon ECS 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateService。

update-task-protection

以下代码示例演示了如何使用 update-task-protection。

Amazon CLI

示例 1：为 ECS 任务启用任务保护

以下 update-task-protection 保护您的 ECS 任务在横向缩减期间不被部署或服务自动扩缩终止。您可以将任务保护的自定义有效期指定为 1 到 2880 分钟（48 小时）。如果您未指定有效期，则启用任务保护默认时间为 2 小时。

```
aws ecs update-task-protection \
    --cluster ECS-project-update-cluster \
    --tasks c43ed3b1331041f289316f958adb6a24 \
    --protection-enabled \
    --expires-in-minutes 300
```

输出：

```
{
"protectedTasks": [
    {
        "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
c43ed3b1331041f289316f958adb6a24",
        "protectionEnabled": true,
        "expirationDate": "2024-09-14T19:53:36.687000-05:00"
    }
],
"failures": []
}
```

示例 2：为 ECS 任务禁用任务保护

以下 update-task-protection 取消保护任务在横向缩减期间不被部署或服务自动扩缩终止。

```
aws ecs update-task-protection \
    --cluster ECS-project-update-cluster \
    --tasks c43ed3b1331041f289316f958adb6a24 \
    --no-protection-enabled
```

输出：

```
{
    "protectedTasks": [
        {
            "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/
c43ed3b1331041f289316f958adb6a24",
            "protectionEnabled": false
        }
    ],
    "failures": []
}
```

有关任务保护的更多信息，请参阅《Amazon ECS 开发人员指南》中的保护您的 Amazon ECS 任务不被横向缩减事件终止。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateTaskProtection。

update-task-set

以下代码示例演示了如何使用 update-task-set。

Amazon CLI

更新任务集

以下 update-task-set 示例更新任务集以调整规模。

```
aws ecs update-task-set \
    --cluster MyCluster \
    --service MyService \
    --task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-svc/1234567890123456789 \
    --scale value=50,unit=PERCENT
```

输出：

```
{
    "taskSet": {
        "id": "ecs-svc/1234567890123456789",
        "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-svc/1234567890123456789",
        "status": "ACTIVE",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/sample-fargate:2",
        "computedDesiredCount": 0,
        "pendingCount": 0,
        "runningCount": 0,
        "createdAt": 1557128360.711,
        "updatedAt": 1557129279.914,
        "launchType": "EC2",
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "subnets": [
```

```
                    "subnet-12344321"
                ],
                "securityGroups": [
                    "sg-12344321"
                ],
                "assignPublicIp": "DISABLED"
            }
        },
        "loadBalancers": [],
        "serviceRegistries": [],
        "scale": {
            "value": 50.0,
            "unit": "PERCENT"
        },
        "stabilityStatus": "STABILIZING",
        "stabilityStatusAt": 1557129279.914
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTaskSet。

使用 Amazon CLI 的 Amazon EFS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon EFS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-file-system`

以下代码示例演示了如何使用 `create-file-system`。

Amazon CLI

创建加密文件系统

以下 create-file-system 示例使用默认 CMK 创建加密文件系统。它还添加了标签 Name=my-file-system。

```
aws efs create-file-system \
    --performance-mode generalPurpose \
    --throughput-mode bursting \
    --encrypted \
    --tags Key=Name,Value=my-file-system
```

输出：

```
{
    "OwnerId": "123456789012",
    "CreationToken": "console-d7f56c5f-e433-41ca-8307-9d9c0example",
    "FileSystemId": "fs-c7a0456e",
    "FileSystemArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-system/
fs-48499b4d",
    "CreationTime": 1595286880.0,
    "LifeCycleState": "creating",
    "Name": "my-file-system",
    "NumberOfMountTargets": 0,
    "SizeInBytes": {
        "Value": 0,
        "ValueInIA": 0,
        "ValueInStandard": 0
    },
    "PerformanceMode": "generalPurpose",
    "Encrypted": true,
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/a59b3472-e62c-42e4-
adcf-30d92example",
    "ThroughputMode": "bursting",
    "Tags": [
        {
            "Key": "Name",
            "Value": "my-file-system"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建 Amazon EFS 文件系统。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFileSystem。

`create-mount-target`

以下代码示例演示了如何使用 `create-mount-target`。

Amazon CLI

创建挂载目标

以下 `create-mount-target` 示例为指定文件系统创建挂载目标。

```
aws efs create-mount-target \
    --file-system-id fs-c7a0456e \
    --subnet-id subnet-02bf4c428bexample \
    --security-groups sg-068f739363example
```

输出：

```
{
    "OwnerId": "123456789012",
    "MountTargetId": "fsmt-f9a14450",
    "FileSystemId": "fs-c7a0456e",
    "SubnetId": "subnet-02bf4c428bexample",
    "LifeCycleState": "creating",
    "IpAddress": "10.0.1.24",
    "NetworkInterfaceId": "eni-02d542216aexample",
    "AvailabilityZoneId": "use2-az2",
    "AvailabilityZoneName": "us-east-2b",
    "VpcId": "vpc-0123456789abcdef0"
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建挂载目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMountTarget。

`delete-file-system`

以下代码示例演示了如何使用 `delete-file-system`。

Amazon CLI

删除文件系统

以下 delete-file-system 示例删除指定的文件系统。

```
aws efs delete-file-system \
    --file-system-id fs-c7a0456e
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的删除 Amazon EFS 文件系统。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFileSystem。

delete-mount-target

以下代码示例演示了如何使用 delete-mount-target。

Amazon CLI

删除挂载目标

以下 delete-mount-target 示例删除指定的挂载目标。

```
aws efs delete-mount-target \
    --mount-target-id fsmt-f9a14450
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建挂载目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMountTarget。

describe-file-systems

以下代码示例演示了如何使用 describe-file-systems。

Amazon CLI

描述文件系统

以下 describe-file-systems 示例描述指定的文件系统。

```
aws efs describe-file-systems \
    --file-system-id fs-c7a0456e
```

输出：

```
{
    "FileSystems": [
        {
            "OwnerId": "123456789012",
            "CreationToken": "console-d7f56c5f-e433-41ca-8307-9d9c0example",
            "FileSystemId": "fs-c7a0456e",
            "FileSystemArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-
system/fs-48499b4d",
            "CreationTime": 1595286880.0,
            "LifeCycleState": "available",
            "Name": "my-file-system",
            "NumberOfMountTargets": 3,
            "SizeInBytes": {
                "Value": 6144,
                "Timestamp": 1600991437.0,
                "ValueInIA": 0,
                "ValueInStandard": 6144
            },
            "PerformanceMode": "generalPurpose",
            "Encrypted": true,
            "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/a59b3472-e62c-42e4-
adcf-30d92example",
            "ThroughputMode": "bursting",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-file-system"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的管理 Amazon EFS 文件系统。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFileSystems。

`describe-mount-targets`

以下代码示例演示了如何使用 `describe-mount-targets`。

Amazon CLI

描述挂载目标

以下 `describe-mount-targets` 示例描述指定的挂载目标。

```
aws efs describe-mount-targets \
    --mount-target-id fsmt-f9a14450
```

输出：

```
{
    "MountTargets": [
        {
            "OwnerId": "123456789012",
            "MountTargetId": "fsmt-f9a14450",
            "FileSystemId": "fs-c7a0456e",
            "SubnetId": "subnet-02bf4c428bexample",
            "LifeCycleState": "creating",
            "IpAddress": "10.0.1.24",
            "NetworkInterfaceId": "eni-02d542216aexample",
            "AvailabilityZoneId": "use2-az2",
            "AvailabilityZoneName": "us-east-2b",
            "VpcId": "vpc-0123456789abcdef0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建挂载目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMountTargets。

`describe-tags`

以下代码示例演示了如何使用 `describe-tags`。

Amazon CLI

描述文件系统的标签

以下 describe-tags 示例描述指定文件系统的标签。

```
aws efs describe-tags \
    --file-system-id fs-c7a0456e
```

输出：

```
{
    "Tags": [
        {
            "Key": "Name",
            "Value": "my-file-system"
        },
        {
            "Key": "Department",
            "Value": "Business Intelligence"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的管理文件系统标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

检索资源的标签

以下 list-tags-for-resource 示例检索与指定文件系统关联的标签。

```
aws efs list-tags-for-resource \
    --resource-id fs-c7a0456e
```

输出：

```
{
    "Tags": [
        {
            "Key": "Name",
            "Value": "my-file-system"
        },
        {
            "Key": "Department",
            "Value": "Business Intelligence"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的管理文件系统标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

标记资源

以下 `tag-resource` 示例将标签 `Department=Business Intelligence` 添加到指定的文件系统中。

```
aws efs tag-resource \
    --resource-id fs-c7a0456e \
    --tags Key=Department,Value="Business Intelligence"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的管理文件系统标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源中删除标签

以下 `untag-resource` 示例从指定的文件系统中删除带有 `Department` 标签键的标签。

```
aws efs untag-resource \
    --resource-id fs-c7a0456e \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的管理文件系统标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Amazon EKS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon EKS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-access-policy`

以下代码示例演示了如何使用 `associate-access-policy`。

Amazon CLI

将访问策略及其范围与集群的访问条目相关联

以下 `associate-access-policy` 将访问策略及其范围与指定集群的访问条目相关联。

```
aws eks associate-access-policy \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/Admin \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy \
    --access-scope type=namespace,namespaces=default
```

输出：

```
{
    "clusterName": "eks-customer",
    "principalArn": "arn:aws:iam::111122223333:role/Admin",
    "associatedAccessPolicy": {
        "policyArn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy",
        "accessScope": {
            "type": "namespace",
            "namespaces": [
                "default"
            ]
        },
        "associatedAt": "2025-05-24T15:59:51.981000-05:00",
        "modifiedAt": "2025-05-24T15:59:51.981000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的将访问策略与访问条目关联起来。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateAccessPolicy。

associate-encryption-config

以下代码示例演示了如何使用 associate-encryption-config。

Amazon CLI

将加密配置关联到现有集群

以下 associate-encryption-config 示例在尚未启用加密的现有 EKS 集群上启用加密。

```
aws eks associate-encryption-config \
    --cluster-name my-eks-cluster \
```

```
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

输出：

```
{
    "update": {
        "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
        "status": "InProgress",
        "type": "AssociateEncryptionConfig",
        "params": [
            {
                "type": "EncryptionConfig",
                "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
            }
        ],
        "createdAt": "2024-03-14T11:01:26.297000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的在现有集群中启用密钥加密。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateEncryptionConfig。

associate-identity-provider-config

以下代码示例演示了如何使用 associate-identity-provider-config。

Amazon CLI

将身份提供商关联到您的 Amazon EKS 集群

以下 associate-identity-provider-config 示例将身份提供商关联到您的 Amazon EKS 集群。

```
aws eks associate-identity-provider-config \
    --cluster-name my-eks-cluster \
    --oidc 'identityProviderConfigName=my-identity-provider,issuerUrl=https://
oidc.eks.us-east-2.amazonaws.com/
id/38D6A4619A0A69E342B113ED7F1A7652,clientId=kubernetes,usernameClaim=email,usernamePrefix=m
```

```
username-prefix,groupsClaim=my-claim,groupsPrefix=my-groups-
prefix,requiredClaims={Claim1=value1,Claim2=value2}' \
    --tags env=dev
```

输出：

```
{
    "update": {
        "id": "8c6c1bef-61fe-42ac-a242-89412387b8e7",
        "status": "InProgress",
        "type": "AssociateIdentityProviderConfig",
        "params": [
            {
                "type": "IdentityProviderConfig",
                "value": "[{\"type\":\"oidc\",\"name\":\"my-identity-provider\"}]"
            }
        ],
        "createdAt": "2024-04-11T13:46:49.648000-04:00",
        "errors": []
    },
    "tags": {
        "env": "dev"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的通过 OpenID Connect 身份提供商对集群的用户进行身份验证 – 关联 OIDC 身份提供商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateIdentityProviderConfig。

create-access-entry

以下代码示例演示了如何使用 create-access-entry。

Amazon CLI

示例 1：为 EKS 集群创建访问条目

以下 create-access-entry 示例创建一个访问条目，该条目支持 IAM 主体访问 EKS 集群。

```
aws eks create-access-entry \
    --cluster-name eks-customer \
```

```
    --principal-arn arn:aws:iam::111122223333:user/eks-user
```

输出：

```
{
    "accessEntry": {
        "clusterName": "eks-customer",
        "principalArn": "arn:aws:iam::111122223333:user/eks-user",
        "kubernetesGroups": [],
        "accessEntryArn": "arn:aws:eks:us-west-2:111122223333:access-entry/eks-
customer/user/111122223333/eks-user/a1b2c3d4-5678-90ab-cdef-a6506e3d36p0",
        "createdAt": "2025-04-14T22:45:48.097000-05:00",
        "modifiedAt": "2025-04-14T22:45:48.097000-05:00",
        "tags": {},
        "username": "arn:aws:iam::111122223333:user/eks-user",
        "type": "STANDARD"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的创建访问条目。

示例 2：通过指定访问条目的类型为 EKS 集群创建访问条目

以下 create-access-entry 示例在 EKS 集群中创建类型为 EC2_LINUX 的访问条目。默认情况下，会创建类型为 STANDARD 的访问条目。除了默认类型之外，如果我们指定任何其它访问条目类型，则需要在 CLI 中传递 IAM 角色 ARN。

```
aws eks create-access-entry \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/admin-test-ip \
    --type EC2_LINUX
```

输出：

```
{
    "accessEntry": {
        "clusterName": "eks-customer",
        "principalArn": "arn:aws:iam::111122223333:role/admin-test-ip",
        "kubernetesGroups": [
            "system:nodes"
        ],
```

```
        "accessEntryArn": "arn:aws:eks:us-west-2:111122223333:access-entry/eks-
customer/role/111122223333/admin-test-ip/accb5418-f493-f390-3e6e-c3f19f725fcp",
        "createdAt": "2025-05-06T19:42:45.453000-05:00",
        "modifiedAt": "2025-05-06T19:42:45.453000-05:00",
        "tags": {},
        "username": "system:node:{{EC2PrivateDNSName}}",
        "type": "EC2_LINUX"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的创建访问条目。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateAccessEntry。

`create-addon`

以下代码示例演示了如何使用 `create-addon`。

Amazon CLI

示例 1：为相应的 EKS 集群版本创建具有默认兼容版本的 Amazon EKS 附加组件

以下 `create-addon` 示例命令为相应的 EKS 集群版本创建具有默认兼容版本的 Amazon EKS 附加组件。

```
aws eks create-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "CREATING",
        "addonVersion": "v1.15.1-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/1ec71ee1-b9c2-8915-4e17-e8be0a55a149",
```

```
        "createdAt": "2024-03-14T12:20:03.264000-04:00",
        "modifiedAt": "2024-03-14T12:20:03.283000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 2：使用特定附加组件版本创建 Amazon EKS 附加组件

以下 `create-addon` 示例命令使用特定附加组件版本创建 Amazon EKS 附加组件。

```
aws eks create-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
    --addon-version v1.16.4-eksbuild.2
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "CREATING",
        "addonVersion": "v1.16.4-eksbuild.2",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/34c71ee6-7738-6c8b-c6bd-3921a176b5ff",
        "createdAt": "2024-03-14T12:30:24.507000-04:00",
        "modifiedAt": "2024-03-14T12:30:24.521000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 3：创建具有自定义配置值的 Amazon EKS 附加组件并解决冲突详细信息

以下 create-addon 示例命令创建具有自定义配置值的 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks create-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
    --addon-version v1.16.4-eksbuild.2 \
    --configuration-values '{"resources":{"limits":{"cpu":"100m"}}}' \
    --resolve-conflicts OVERWRITE
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "CREATING",
        "addonVersion": "v1.16.4-eksbuild.2",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/a6c71ee9-0304-9237-1be8-25af1b0f1ffb",
        "createdAt": "2024-03-14T12:35:58.313000-04:00",
        "modifiedAt": "2024-03-14T12:35:58.327000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {},
        "configurationValues": "{\"resources\":{\"limits\":{\"cpu\":\"100m\"}}}"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 4：使用自定义 JSON 配置值文件创建 Amazon EKS 附加组件

以下 create-addon 示例命令创建具有自定义配置值的 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks create-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
    --addon-version v1.16.4-eksbuild.2 \
    --configuration-values 'file://configuration-values.json' \
    --resolve-conflicts OVERWRITE \
    --tags '{"eks-addon-key-1": "value-1" , "eks-addon-key-2": "value-2"}'
```

的内容configuration-values.json：

```
{
    "resources": {
        "limits": {
            "cpu": "150m"
        }
    },
    "env": {
        "AWS_VPC_K8S_CNI_LOGLEVEL": "ERROR"
    }
}
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "CREATING",
        "addonVersion": "v1.16.4-eksbuild.2",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/d8c71ef8-fbd8-07d0-fb32-6a7be19ececd",
        "createdAt": "2024-03-14T13:10:51.763000-04:00",
        "modifiedAt": "2024-03-14T13:10:51.777000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {
            "eks-addon-key-1": "value-1",
            "eks-addon-key-2": "value-2"
        },
```

```
        "configurationValues": "{\n    \"resources\": {\n        \"limits\":
 {\n            \"cpu\": \"150m\"\n        }\n    },\n    \"env\": {\n
 \"AWS_VPC_K8S_CNI_LOGLEVEL\": \"ERROR\"\n    }\n}"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 5：使用自定义 YAML 配置值文件创建 Amazon EKS 附加组件

以下 `create-addon` 示例命令创建具有自定义配置值的 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks create-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
    --addon-version v1.16.4-eksbuild.2 \
    --configuration-values 'file://configuration-values.yaml' \
    --resolve-conflicts OVERWRITE \
    --tags '{"eks-addon-key-1": "value-1" , "eks-addon-key-2": "value-2"}'
```

的内容`configuration-values.yaml`：

```
resources:
    limits:
        cpu: '100m'
env:
    AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "CREATING",
        "addonVersion": "v1.16.4-eksbuild.2",
        "health": {
            "issues": []
        },
```

```
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/d4c71efb-3909-6f36-a548-402cd4b5d59e",
        "createdAt": "2024-03-14T13:15:45.220000-04:00",
        "modifiedAt": "2024-03-14T13:15:45.237000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {
            "eks-addon-key-3": "value-3",
            "eks-addon-key-4": "value-4"
        },
        "configurationValues": "resources:\n    limits:\n        cpu: '100m'\nenv:\n
    AWS_VPC_K8S_CNI_LOGLEVEL: 'INFO'"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAddon。

create-cluster

以下代码示例演示了如何使用 create-cluster。

Amazon CLI

创建新集群

此示例命令将在您的默认区域中创建一个名为 prod 的集群。

命令:

```
aws eks create-cluster --name prod \
--role-arn arn:aws:iam::012345678910:role/eks-service-role-
AWSServiceRoleForAmazonEKS-J7ONKE3BQ4PI \
--resources-vpc-config subnetIds=subnet-6782e71e,subnet-
e7e761ac,securityGroupIds=sg-6979fe18
```

输出：

```
{
    "cluster": {
        "name": "prod",
        "arn": "arn:aws:eks:us-west-2:012345678910:cluster/prod",
```

```
        "createdAt": 1527808069.147,
        "version": "1.10",
        "roleArn": "arn:aws:iam::012345678910:role/eks-service-role-
AWSServiceRoleForAmazonEKS-J7ONKE3BQ4PI",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-6782e71e",
                "subnet-e7e761ac"
            ],
            "securityGroupIds": [
                "sg-6979fe18"
            ],
            "vpcId": "vpc-950809ec"
        },
        "status": "CREATING",
        "certificateAuthority": {}
    }
}
```

创建启用了私有端点访问和日志记录的新集群

此示例命令在您的默认区域中创建一个名为 example 的集群，该集群禁用了公共端点访问，启用了私有端点访问和所有日志记录类型。

命令:

```
aws eks create-cluster --name example --kubernetes-version 1.12 \
--role-arn arn:aws:iam::012345678910:role/example-cluster-ServiceRole-1XWBQWYSFRE2Q
 \
--resources-vpc-
config subnetIds=subnet-0a188dccd2f9a632f,subnet-09290d93da4278664,subnet-0f21dd86e0e91134a,
 \
--logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

输出：

```
{
    "cluster": {
        "name": "example",
        "arn": "arn:aws:eks:us-west-2:012345678910:cluster/example",
        "createdAt": 1565804921.901,
        "version": "1.12",
```

```
        "roleArn": "arn:aws:iam::012345678910:role/example-cluster-
ServiceRole-1XWBQWYSFRE2Q",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-0a188dccd2f9a632f",
                "subnet-09290d93da4278664",
                "subnet-0f21dd86e0e91134a",
                "subnet-0173dead68481a583",
                "subnet-051f70a57ed6fcab6",
                "subnet-01322339c5c7de9b4"
            ],
            "securityGroupIds": [
                "sg-0c5b580845a031c10"
            ],
            "vpcId": "vpc-0f622c01f68d4afec",
            "endpointPublicAccess": false,
            "endpointPrivateAccess": true
        },
        "logging": {
            "clusterLogging": [
                {
                    "types": [
                        "api",
                        "audit",
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ],
                    "enabled": true
                }
            ]
        },
        "status": "CREATING",
        "certificateAuthority": {},
        "platformVersion": "eks.3"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-fargate-profile

以下代码示例演示了如何使用 create-fargate-profile。

Amazon CLI

示例 1：为具有命名空间的选择器创建 EKS Fargate 配置文件

以下 `create-fargate-profile` 示例为具有命名空间的选择器创建 EKS Fargate 配置文件。

```
aws eks create-fargate-profile \
    --cluster-name my-eks-cluster \
    --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
    --fargate-profile-name my-fargate-profile \
    --selectors '[{"namespace": "default"}]'
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/a2c72bca-318e-abe8-8ed1-27c6d4892e9e",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T12:38:47.368000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "default"
            }
        ],
        "status": "CREATING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——创建 Fargate 配置文件。

示例 2：为具有命名空间和标签的选择器创建 EKS Fargate 配置文件

以下 create-fargate-profile 示例为具有命名空间和标签的选择器创建 EKS Fargate 配置文件。

```
aws eks create-fargate-profile \
    --cluster-name my-eks-cluster \
    --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
    --fargate-profile-name my-fargate-profile \
    --selectors '[{"namespace": "default", "labels": {"labelname1":
 "labelvalue1"}}]'
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/88c72bc7-e8a4-fa34-44e4-2f1397224bb3",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T12:33:48.125000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "default",
                "labels": {
                    "labelname1": "labelvalue1"
                }
            }
        ],
        "status": "CREATING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——创建 Fargate 配置文件。

示例 3：为具有单个命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 的选择器创建 EKS Fargate 配置文件

以下 `create-fargate-profile` 示例为具有单个命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 的选择器创建 EKS Fargate 配置文件。

```
aws eks create-fargate-profile \
    --cluster-name my-eks-cluster \
    --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
    --fargate-profile-name my-fargate-profile \
    --selectors '[{"namespace": "default", "labels": {"labelname1":
 "labelvalue1"}}]' \
    --subnets '["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d",
 "subnet-0e2907431c9988b72"]'
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/e8c72bc8-e87b-5eb6-57cb-ed4fe57577e3",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T12:35:58.640000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "default",
                "labels": {
                    "labelname1": "labelvalue1"
                }
            }
        ],
        "status": "CREATING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——创建 Fargate 配置文件。

示例 4：为具有多个命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 的选择器创建 EKS Fargate 配置文件

以下 `create-fargate-profile` 示例为具有多个命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 的选择器创建 EKS Fargate 配置文件。

```
aws eks create-fargate-profile \
    --cluster-name my-eks-cluster \
    --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
    --fargate-profile-name my-fargate-profile \
    --selectors '[{"namespace": "default1", "labels": {"labelname1": "labelvalue1",
 "labelname2": "labelvalue2"}}, {"namespace": "default2", "labels": {"labelname1":
 "labelvalue1", "labelname2": "labelvalue2"}}]' \
    --subnets '["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d",
 "subnet-0e2907431c9988b72"]' \
    --tags '{"eks-fargate-profile-key-1": "value-1" , "eks-fargate-profile-key-2":
 "value-2"}'
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/4cc72bbf-b766-8ee6-8d29-e62748feb3cd",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T12:15:55.271000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "default1",
                "labels": {
                    "labelname2": "labelvalue2",
                    "labelname1": "labelvalue1"
                }
```

```
            },
            {
                "namespace": "default2",
                "labels": {
                    "labelname2": "labelvalue2",
                    "labelname1": "labelvalue1"
                }
            }
        ],
        "status": "CREATING",
        "tags": {
            "eks-fargate-profile-key-2": "value-2",
            "eks-fargate-profile-key-1": "value-1"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——创建 Fargate 配置文件。

示例 5：为命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 创建带有通配符选择器的 EKS Fargate 配置文件

以下 `create-fargate-profile` 示例为具有多个命名空间和标签以及要将容器组（Pod）启动到的子网的 ID 的选择器创建 EKS Fargate 配置文件。

```
aws eks create-fargate-profile \
    --cluster-name my-eks-cluster \
    --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
    --fargate-profile-name my-fargate-profile \
    --selectors '[{"namespace": "prod*", "labels": {"labelname*?": "*value1"}},
 {"namespace": "*dev*", "labels": {"labelname*?": "*value*"}}]' \
    --subnets '["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d",
 "subnet-0e2907431c9988b72"]' \
    --tags '{"eks-fargate-profile-key-1": "value-1" , "eks-fargate-profile-key-2":
 "value-2"}'
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
```

```
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/e8c72bd6-5966-0bfe-b77b-1802893e5a6f",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T13:05:20.550000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "prod*",
                "labels": {
                    "labelname*?": "*value1"
                }
            },
            {
                "namespace": "*dev*",
                "labels": {
                    "labelname*?": "*value*"
                }
            }
        ],
        "status": "CREATING",
        "tags": {
            "eks-fargate-profile-key-2": "value-2",
            "eks-fargate-profile-key-1": "value-1"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——创建 Fargate 配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFargateProfile。

`create-nodegroup`

以下代码示例演示了如何使用 `create-nodegroup`。

Amazon CLI

示例 1：为 Amazon EKS 集群创建托管节点组

以下 `create-nodegroup` 示例为 Amazon EKS 集群创建托管节点组。

```
aws eks create-nodegroup \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --node-role arn:aws:iam::111122223333:role/role-name \
    --
subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d" "subnet-09d912bb63ef21b9a"
 \
    --scaling-config minSize=1,maxSize=3,desiredSize=1 \
    --region us-east-2
```

输出：

```
{
    "nodegroup": {
        "nodegroupName": "my-eks-nodegroup",
        "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/bac7550f-b8b8-5fbb-4f3e-7502a931119e",
        "clusterName": "my-eks-cluster",
        "version": "1.26",
        "releaseVersion": "1.26.12-20240329",
        "createdAt": "2024-04-04T13:19:32.260000-04:00",
        "modifiedAt": "2024-04-04T13:19:32.260000-04:00",
        "status": "CREATING",
        "capacityType": "ON_DEMAND",
        "scalingConfig": {
            "minSize": 1,
            "maxSize": 3,
            "desiredSize": 1
        },
        "instanceTypes": [
            "t3.medium"
        ],
        "subnets": [
            "subnet-0e2907431c9988b72, subnet-04ad87f71c6e5ab4d,
 subnet-09d912bb63ef21b9a"
        ],
        "amiType": "AL2_x86_64",
```

```
        "nodeRole": "arn:aws:iam::111122223333:role/role-name",
        "diskSize": 20,
        "health": {
            "issues": []
        },
        "updateConfig": {
            "maxUnavailable": 1
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的创建托管节点组。

示例 2：使用自定义实例类型和磁盘大小为 Amazon EKS 集群创建托管节点组

以下 `create-nodegroup` 示例使用自定义实例类型和磁盘大小为 Amazon EKS 集群创建托管节点组。

```
aws eks create-nodegroup \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --node-role arn:aws:iam::111122223333:role/role-name \
    --
subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d" "subnet-09d912bb63ef21b9a"
 \
    --scaling-config minSize=1,maxSize=3,desiredSize=1 \
    --capacity-type ON_DEMAND \
    --instance-types 'm5.large' \
    --disk-size 50 \
    --region us-east-2
```

输出：

```
{
    "nodegroup": {
        "nodegroupName": "my-eks-nodegroup",
        "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/c0c7551b-e4f9-73d9-992c-a450fdb82322",
        "clusterName": "my-eks-cluster",
        "version": "1.26",
        "releaseVersion": "1.26.12-20240329",
```

```
        "createdAt": "2024-04-04T13:46:07.595000-04:00",
        "modifiedAt": "2024-04-04T13:46:07.595000-04:00",
        "status": "CREATING",
        "capacityType": "ON_DEMAND",
        "scalingConfig": {
            "minSize": 1,
            "maxSize": 3,
            "desiredSize": 1
        },
        "instanceTypes": [
            "m5.large"
        ],
        "subnets": [
            "subnet-0e2907431c9988b72",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-09d912bb63ef21b9a"
        ],
        "amiType": "AL2_x86_64",
        "nodeRole": "arn:aws:iam::111122223333:role/role-name",
        "diskSize": 50,
        "health": {
            "issues": []
        },
        "updateConfig": {
            "maxUnavailable": 1
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的创建托管节点组。

示例 3：使用自定义实例类型、磁盘大小、AMI 类型、容量类型、更新配置、标签、污点和标记为 Amazon EKS 集群创建托管节点组

以下 `create-nodegroup` 示例使用自定义实例类型、磁盘大小、AMI 类型、容量类型、更新配置、标签、污点和标记为 Amazon EKS 集群创建托管节点组。

```
aws eks create-nodegroup  \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --node-role arn:aws:iam::111122223333:role/role-name \
```

```
    --
subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d" "subnet-09d912bb63ef21b9a"
 \
    --scaling-config minSize=1,maxSize=5,desiredSize=4 \
    --instance-types 't3.large' \
    --disk-size 50 \
    --ami-type AL2_x86_64 \
    --capacity-type SPOT \
    --update-config maxUnavailable=2 \
    --labels '{"my-eks-nodegroup-label-1": "value-1" , "my-eks-nodegroup-label-2":
 "value-2"}' \
    --taints '{"key": "taint-key-1" , "value": "taint-value-1", "effect":
 "NO_EXECUTE"}' \
    --tags '{"my-eks-nodegroup-key-1": "value-1" , "my-eks-nodegroup-key-2":
 "value-2"}'
```

输出：

```
{
    "nodegroup": {
        "nodegroupName": "my-eks-nodegroup",
        "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/88c75524-97af-0cb9-a9c5-7c0423ab5314",
        "clusterName": "my-eks-cluster",
        "version": "1.26",
        "releaseVersion": "1.26.12-20240329",
        "createdAt": "2024-04-04T14:05:07.940000-04:00",
        "modifiedAt": "2024-04-04T14:05:07.940000-04:00",
        "status": "CREATING",
        "capacityType": "SPOT",
        "scalingConfig": {
            "minSize": 1,
            "maxSize": 5,
            "desiredSize": 4
        },
        "instanceTypes": [
            "t3.large"
        ],
        "subnets": [
            "subnet-0e2907431c9988b72",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-09d912bb63ef21b9a"
        ],
```

```
        "amiType": "AL2_x86_64",
        "nodeRole": "arn:aws:iam::111122223333:role/role-name",
        "labels": {
            "my-eks-nodegroup-label-2": "value-2",
            "my-eks-nodegroup-label-1": "value-1"
        },
        "taints": [
            {
                "key": "taint-key-1",
                "value": "taint-value-1",
                "effect": "NO_EXECUTE"
            }
        ],
        "diskSize": 50,
        "health": {
            "issues": []
        },
        "updateConfig": {
            "maxUnavailable": 2
        },
        "tags": {
            "my-eks-nodegroup-key-1": "value-1",
            "my-eks-nodegroup-key-2": "value-2"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的创建托管节点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNodegroup。

create-pod-identity-association

以下代码示例演示了如何使用 create-pod-identity-association。

Amazon CLI

示例 1：在 EKS 集群中创建 EKS 容器组身份关联

以下 create-pod-identity-association 示例在 EKS 集群中的服务账户与 IAM 角色之间创建 EKS 容器组身份关联。

```
aws eks create-pod-identity-association \
```

```
    --cluster-name eks-customer \
    --namespace default \
    --service-account default \
    --role-arn arn:aws:iam::111122223333:role/my-role
```

输出：

```
{
    "association": {
        "clusterName": "eks-customer",
        "namespace": "default",
        "serviceAccount": "default",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-8mvwvh57cu74mgcst",
        "associationId": "a-8mvwvh57cu74mgcst",
        "tags": {},
        "createdAt": "2025-05-24T19:40:13.961000-05:00",
        "modifiedAt": "2025-05-24T19:40:13.961000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

示例 2：使用标签在 EKS 集群中创建 EKS 容器组身份关联

以下 `create-pod-identity-association` 使用标签在 EKS 集群中的服务账户与 IAM 角色之间创建 EKS 容器组身份关联。

```
aws eks create-pod-identity-association \
    --cluster-name eks-customer \
    --namespace default \
    --service-account default \
    --role-arn arn:aws:iam::111122223333:role/my-role \
    --tags Key1=value1,Key2=value2
```

输出：

```
{
    "association": {
```

```
        "clusterName": "eks-customer",
        "namespace": "default",
        "serviceAccount": "default",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-9njjin9gfghecgoda",
        "associationId": "a-9njjin9gfghecgoda",
        "tags": {
            "Key2": "value2",
            "Key1": "value1"
        },
        "createdAt": "2025-05-24T19:52:14.135000-05:00",
        "modifiedAt": "2025-05-24T19:52:14.135000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreatePodIdentityAssociation。

delete-access-entry

以下代码示例演示了如何使用 delete-access-entry。

Amazon CLI

删除与集群关联的访问条目

以下 delete-access-entry 删除与名为 eks-customer 的 EKS 集群关联的访问条目。

```
aws eks delete-access-entry \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/Admin
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EKS 用户指南》中的删除访问条目。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteAccessEntry。

delete-addon

以下代码示例演示了如何使用 delete-addon。

Amazon CLI

示例 1：删除 Amazon EKS 附加组件，但在 EKS 集群上保留附加组件软件

以下 delete-addon 示例命令删除 Amazon EKS 附加组件，但在 EKS 集群上保留附加组件软件。

```
aws eks delete-addon \
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon \
    --preserve
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "DELETING",
        "addonVersion": "v1.9.3-eksbuild.7",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/a8c71ed3-944e-898b-9167-c763856af4b8",
        "createdAt": "2024-03-14T11:49:09.009000-04:00",
        "modifiedAt": "2024-03-14T12:03:49.776000-04:00",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS》中的管理 Amazon EKS 附加组件 – 删除附加组件。

示例 2：删除 Amazon EKS 附加组件，并从 EKS 集群中删除该附加组件软件

以下 delete-addon 示例命令删除 Amazon EKS 附加组件，并从 EKS 集群中删除该附加组件软件。

```
aws eks delete-addon \
```

```
    --cluster-name my-eks-cluster \
    --addon-name my-eks-addon
```

输出：

```
{
    "addon": {
        "addonName": "my-eks-addon",
        "clusterName": "my-eks-cluster",
        "status": "DELETING",
        "addonVersion": "v1.15.1-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/bac71ed1-ec43-3bb6-88ea-f243cdb58954",
        "createdAt": "2024-03-14T11:45:31.983000-04:00",
        "modifiedAt": "2024-03-14T11:58:40.136000-04:00",
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS》中的管理 Amazon EKS 附加组件 – 删除附加组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAddon。

delete-cluster

以下代码示例演示了如何使用 delete-cluster。

Amazon CLI

删除 Amazon EKS 集群控制面板

以下 delete-cluster 示例删除 Amazon EKS 集群控制面板。

```
aws eks delete-cluster \
    --name my-eks-cluster
```

输出：

```
{
```

```
    "cluster": {
        "name": "my-eks-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",
        "createdAt": "2024-03-14T11:31:44.348000-04:00",
        "version": "1.27",
        "endpoint": "https://DALSJ343KE23J3RN45653DSKJTT647TYD.yl4.us-
east-2.eks.amazonaws.com",
        "roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-
ServiceRole-zMF6CBakwwbW",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-0fb75d2d8401716e7",
                "subnet-02184492f67a3d0f9",
                "subnet-04098063527aab776",
                "subnet-0e2907431c9988b72",
                "subnet-04ad87f71c6e5ab4d",
                "subnet-09d912bb63ef21b9a"
            ],
            "securityGroupIds": [
                "sg-0c1327f6270afbb36"
            ],
            "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
            "vpcId": "vpc-0012b8e1cc0abb17d",
            "endpointPublicAccess": true,
            "endpointPrivateAccess": true,
            "publicAccessCidrs": [
                "0.0.0.0/0"
            ]
        },
        "kubernetesNetworkConfig": {
            "serviceIpv4Cidr": "10.100.0.0/16",
            "ipFamily": "ipv4"
        },
        "logging": {
            "clusterLogging": [
                {
                    "types": [
                        "api",
                        "audit",
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ],
                    "enabled": true
```

```
                }
            ]
        },
        "identity": {
            "oidc": {
                "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
DALSJ343KE23J3RN45653DSKJTT647TYD"
            }
        },
        "status": "DELETING",
        "certificateAuthority": {
            "data": "XXX_CA_DATA_XXX"
        },
        "platformVersion": "eks.16",
        "tags": {
            "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
            "alpha.eksctl.io/cluster-name": "my-eks-cluster",
            "karpenter.sh/discovery": "my-eks-cluster",
            "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
            "auto-delete": "no",
            "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
            "EKS-Cluster-Name": "my-eks-cluster",
            "alpha.eksctl.io/cluster-oidc-enabled": "true",
            "aws:cloudformation:logical-id": "ControlPlane",
            "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
            "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
        },
        "accessConfig": {
            "authenticationMode": "API_AND_CONFIG_MAP"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的删除 Amazon EKS 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCluster。

delete-fargate-profile

以下代码示例演示了如何使用 delete-fargate-profile。

Amazon CLI

示例 1：为具有命名空间的选择器创建 EKS Fargate 配置文件

以下 delete-fargate-profile 示例为具有命名空间的选择器创建 EKS Fargate 配置文件。

```
aws eks delete-fargate-profile \
    --cluster-name my-eks-cluster \
    --fargate-profile-name my-fargate-profile
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/1ac72bb3-3fc6-2631-f1e1-98bff53bed62",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-03-19T11:48:39.975000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "default",
                "labels": {
                    "foo": "bar"
                }
            }
        ],
        "status": "DELETING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon Fargate 配置文件——删除 Fargate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFargateProfile。

delete-nodegroup

以下代码示例演示了如何使用 delete-nodegroup。

Amazon CLI

示例 1：删除 Amazon EKS 集群的托管节点组

以下 delete-nodegroup 示例删除 Amazon EKS 集群的托管节点组。

```
aws eks delete-nodegroup \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup
```

输出：

```
{
    "nodegroup": {
        "nodegroupName": "my-eks-nodegroup",
        "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/1ec75f5f-0e21-dcc0-b46e-f9c442685cd8",
        "clusterName": "my-eks-cluster",
        "version": "1.26",
        "releaseVersion": "1.26.12-20240329",
        "createdAt": "2024-04-08T13:25:15.033000-04:00",
        "modifiedAt": "2024-04-08T13:25:31.252000-04:00",
        "status": "DELETING",
        "capacityType": "SPOT",
        "scalingConfig": {
            "minSize": 1,
            "maxSize": 5,
            "desiredSize": 4
        },
        "instanceTypes": [
            "t3.large"
        ],
        "subnets": [
            "subnet-0e2907431c9988b72",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-09d912bb63ef21b9a"
        ],
        "amiType": "AL2_x86_64",
        "nodeRole": "arn:aws:iam::111122223333:role/role-name",
        "labels": {
```

```
            "my-eks-nodegroup-label-2": "value-2",
            "my-eks-nodegroup-label-1": "value-1"
        },
        "taints": [
            {
                "key": "taint-key-1",
                "value": "taint-value-1",
                "effect": "NO_EXECUTE"
            }
        ],
        "diskSize": 50,
        "health": {
            "issues": []
        },
        "updateConfig": {
            "maxUnavailable": 2
        },
        "tags": {
            "my-eks-nodegroup-key-1": "value-1",
            "my-eks-nodegroup-key-2": "value-2"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNodegroup。

delete-pod-identity-association

以下代码示例演示了如何使用 delete-pod-identity-association。

Amazon CLI

删除 EKS 容器组身份关联

以下 delete-pod-identity-association 示例从名为 eks-customer 的 EKS 集群中删除关联 ID 为 a-9njjin9gfghecgocd 的 EKS 容器组身份关联。

```
aws eks delete-pod-identity-association \
    --cluster-name eks-customer \
    --association-id a-9njjin9gfghecgocd
```

输出：

```
{
    "association": {
        "clusterName": "eks-customer",
        "namespace": "default",
        "serviceAccount": "default",
        "roleArn": "arn:aws:iam::111122223333:role/s3-role",
        "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-9njjin9gfghecgocd",
        "associationId": "a-9njjin9gfghecgocd",
        "tags": {
            "Key2": "value2",
            "Key1": "value1"
        },
        "createdAt": "2025-05-24T19:52:14.135000-05:00",
        "modifiedAt": "2025-05-25T21:10:56.923000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeletePodIdentityAssociation。

deregister-cluster

以下代码示例演示了如何使用 deregister-cluster。

Amazon CLI

取消注册已连接的集群以将其从 Amazon EKS 控制面板中删除

以下 deregister-cluster 示例取消注册已连接的集群以将其从 Amazon EKS 控制面板中删除。

```
aws eks deregister-cluster \
    --name my-eks-anywhere-cluster
```

输出：

```
{
    "cluster": {
```

```
        "name": "my-eks-anywhere-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-anywhere-cluster",
        "createdAt": "2024-04-12T12:38:37.561000-04:00",
        "status": "DELETING",
        "tags": {},
        "connectorConfig": {
            "activationId": "dfb5ad28-13c3-4e26-8a19-5b2457638c74",
            "activationExpiry": "2024-04-15T12:38:37.082000-04:00",
            "provider": "EKS_ANYWHERE",
            "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSConnectorAgentRole"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的取消注册集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterCluster。

describe-access-entry

以下代码示例演示了如何使用 describe-access-entry。

Amazon CLI

描述 EKS 集群的访问条目

以下 describe-access-entry 示例描述 EKS 集群的访问条目。

```
aws eks describe-access-entry \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:user/eks-admin-user
```

输出：

```
{
    "accessEntry": {
        "clusterName": "eks-customer",
        "principalArn": "arn:aws:iam::111122223333:user/eks-admin-user",
        "kubernetesGroups": [],
        "accessEntryArn": "arn:aws:eks:us-west-2:111122223333:access-entry/eks-
customer/user/111122223333/eks-admin-user/0acb1bc6-cb0a-ede6-11ae-a6506e3d36p0",
        "createdAt": "2025-04-14T22:45:48.097000-05:00",
        "modifiedAt": "2025-04-14T22:45:48.097000-05:00",
```

```
        "tags": {},
        "username": "arn:aws:iam::111122223333:user/eks-admin-user",
        "type": "STANDARD"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的使用 EKS 访问条目向 IAM 用户授予 Kubernetes 访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAccessEntry。

`describe-addon-configuration`

以下代码示例演示了如何使用 `describe-addon-configuration`。

Amazon CLI

示例 1：创建或更新 Amazon vpc-cni 附加组件时可用的配置选项

以下 `describe-addon-configuration` 示例返回您在为相应版本的 vpc-cni 附加组件创建或更新附加组件时使用的所有可用配置架构。

```
aws eks describe-addon-configuration \
    --addon-name vpc-cni \
    --addon-version v1.15.1-eksbuild.1
```

输出：

```
{
    "addonName": "vpc-cni",
    "addonVersion": "v1.15.1-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema\":\"http://
json-schema.org/draft-06/schema#\",\"definitions\":{\"Affinity\":{\"type\":
[\"object\",\"null\"]},\"EniConfig\":{\"additionalProperties\":false,\"properties
\":{\"create\":{\"type\":\"boolean\"},\"region\":{\"type\":\"string\"},\"subnets\":
{\"additionalProperties\":{\"additionalProperties\":false,\"properties\":{\"id\":
{\"type\":\"string\"},\"securityGroups\":{\"items\":{\"type\":\"string\"},\"type\":
\"array\"}},\"required\":[\"id\"],\"type\":\"object\"},\"minProperties\":1,\"type
\":\"object\"}},\"required\":[\"create\",\"region\",\"subnets\"],\"type\":\"object
\"},\"Env\":{\"additionalProperties\":false,\"properties\":{\"ADDITIONAL_ENI_TAGS
\":{\"type\":\"string\"},\"ANNOTATE_POD_IP\":{\"format\":\"boolean\",\"type\":
\"string\"},\"AWS_EC2_ENDPOINT\":{\"type\":\"string\"},\"AWS_EXTERNAL_SERVICE_CIDRS
\":{\"type\":\"string\"},\"AWS_MANAGE_ENIS_NON_SCHEDULABLE\":{\"format\":\"boolean
```

\",\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT\":{\"format\":\"boolean
\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":\"integer\",\"type\":
\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":{\"format\":\"boolean\",\"type
\":\"string\"},\"AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS\":{\"type\":\"string\"},
\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":{\"format\":\"boolean\",\"type\":\"string\"},
\"AWS_VPC_K8S_CNI_LOGLEVEL\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOG_FILE\":
{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT\":{\"type\":\"string\"},
\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string\"},\"AWS_VPC_K8S_PLUGIN_LOG_FILE
\":{\"type\":\"string\"},\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL\":{\"type\":\"string
\"},\"CLUSTER_ENDPOINT\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION\":
{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_LEAKED_ENI_CLEANUP\":
{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_METRICS\":{\"format
\":\"boolean\",\"type\":\"string\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING
\":{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_POD_V6\":{\"format
\":\"boolean\",\"type\":\"string\"},\"ENABLE_BANDWIDTH_PLUGIN\":{\"format\":
\"boolean\",\"type\":\"string\"},\"ENABLE_POD_ENI\":{\"format\":\"boolean\",
\"type\":\"string\"},\"ENABLE_PREFIX_DELEGATION\":{\"format\":\"boolean\",
\"type\":\"string\"},\"ENABLE_V4_EGRESS\":{\"format\":\"boolean\",\"type\":
\"string\"},\"ENABLE_V6_EGRESS\":{\"format\":\"boolean\",\"type\":\"string\"},
\"ENI_CONFIG_ANNOTATION_DEF\":{\"type\":\"string\"},\"ENI_CONFIG_LABEL_DEF\":
{\"type\":\"string\"},\"INTROSPECTION_BIND_ADDRESS\":{\"type\":\"string\"},
\"IP_COOLDOWN_PERIOD\":{\"format\":\"integer\",\"type\":\"string\"},\"MAX_ENI
\":{\"format\":\"integer\",\"type\":\"string\"},\"MINIMUM_IP_TARGET\":{\"format
\":\"integer\",\"type\":\"string\"},\"POD_SECURITY_GROUP_ENFORCING_MODE\":
{\"type\":\"string\"},\"WARM_ENI_TARGET\":{\"format\":\"integer\",\"type\":
\"string\"},\"WARM_IP_TARGET\":{\"format\":\"integer\",\"type\":\"string\"},
\"WARM_PREFIX_TARGET\":{\"format\":\"integer\",\"type\":\"string\"}},\"title
\":\"Env\",\"type\":\"object\"},\"Init\":{\"additionalProperties\":false,
\"properties\":{\"env\":{\"$ref\":\"#/definitions/InitEnv\"}},\"title\":\"Init
\",\"type\":\"object\"},\"InitEnv\":{\"additionalProperties\":false,\"properties
\":{\"DISABLE_TCP_EARLY_DEMUX\":{\"format\":\"boolean\",\"type\":\"string\"},
\"ENABLE_V6_EGRESS\":{\"format\":\"boolean\",\"type\":\"string\"}},\"title\":
\"InitEnv\",\"type\":\"object\"},\"Limits\":{\"additionalProperties\":false,
\"properties\":{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type\":\"string\"}},
\"title\":\"Limits\",\"type\":\"object\"},\"NodeAgent\":{\"additionalProperties
\":false,\"properties\":{\"enableCloudWatchLogs\":{\"format\":\"boolean\",
\"type\":\"string\"},\"enablePolicyEventLogs\":{\"format\":\"boolean\",\"type\":
\"string\"},\"healthProbeBindAddr\":{\"format\":\"integer\",\"type\":\"string
\"},\"metricsBindAddr\":{\"format\":\"integer\",\"type\":\"string\"}},\"title\":
\"NodeAgent\",\"type\":\"object\"},\"Resources\":{\"additionalProperties\":false,
\"properties\":{\"limits\":{\"$ref\":\"#/definitions/Limits\"},\"requests\":
{\"$ref\":\"#/definitions/Limits\"}},\"title\":\"Resources\",\"type\":\"object
\"},\"Tolerations\":{\"additionalProperties\":false,\"items\":{\"type\":\"object
\"},\"type\":\"array\"},\"VpcCni\":{\"additionalProperties\":false,\"properties

```
\":{\"affinity\":{\"$ref\":\"#/definitions/Affinity\"},\"enableNetworkPolicy\":
{\"format\":\"boolean\",\"type\":\"string\"},\"enableWindowsIpam\":{\"format\":
\"boolean\",\"type\":\"string\"},\"eniConfig\":{\"$ref\":\"#/definitions/EniConfig
\"},\"env\":{\"$ref\":\"#/definitions/Env\"},\"init\":{\"$ref\":\"#/definitions/Init
\"},\"livenessProbeTimeoutSeconds\":{\"type\":\"integer\"},\"nodeAgent\":{\"$ref\":
\"#/definitions/NodeAgent\"},\"readinessProbeTimeoutSeconds\":{\"type\":\"integer
\"},\"resources\":{\"$ref\":\"#/definitions/Resources\"},\"tolerations\":{\"$ref
\":\"#/definitions/Tolerations\"}},\"title\":\"VpcCni\",\"type\":\"object\"}},
\"description\":\"vpc-cni\"}"
}
```

示例 2：创建或更新 Amazon coredns 附加组件时可用的配置选项

以下 `describe-addon-configuration` 示例返回您在为相应版本的 coredns 附加组件创建或更新附加组件时使用的所有可用配置架构。

```
aws eks describe-addon-configuration \
    --addon-name coredns \
    --addon-version v1.8.7-eksbuild.4
```

输出：

```
{
    "addonName": "coredns",
    "addonVersion": "v1.8.7-eksbuild.4",
    "configurationSchema": "{\"$ref\":\"#/definitions/Coredns\",\"$schema
\":\"http://json-schema.org/draft-06/schema#\",\"definitions\":{\"Coredns\":
{\"additionalProperties\":false,\"properties\":{\"computeType\":{\"type\":
\"string\"},\"corefile\":{\"description\":\"Entire corefile contents to use with
 installation\",\"type\":\"string\"},\"nodeSelector\":{\"additionalProperties\":
{\"type\":\"string\"},\"type\":\"object\"},\"replicaCount\":{\"type\":\"integer
\"},\"resources\":{\"$ref\":\"#/definitions/Resources\"}},\"title\":\"Coredns\",
\"type\":\"object\"},\"Limits\":{\"additionalProperties\":false,\"properties\":
{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type\":\"string\"}},\"title\":\"Limits
\",\"type\":\"object\"},\"Resources\":{\"additionalProperties\":false,\"properties
\":{\"limits\":{\"$ref\":\"#/definitions/Limits\"},\"requests\":{\"$ref\":\"#/
definitions/Limits\"}},\"title\":\"Resources\",\"type\":\"object\"}}}"
}
```

有关更多信息，请参阅《Amazon EKS》中的为 Amazon EKS 集群创建或更新 kubeconfig 文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddonConfiguration。

describe-addon-versions

以下代码示例演示了如何使用 describe-addon-versions。

Amazon CLI

示例 1：列出 EKS 集群的所有可用附加组件

以下 describe-addon-versions 示例列出了所有可用的 Amazon 附加组件。

```
aws eks describe-addon-versions  \
    --query 'sort_by(addons  &owner)[].{publisher: publisher, owner: owner,
 addonName: addonName, type: type}' \
    --output table
```

输出：

```
-----------------------------------------------------------------------------------------------
|                                                DescribeAddonVersions
                                |
+---------------------------------------------+------------------
+-----------------------+------------------------+
|                  addonName                  |      owner       |       publisher
     |          type           |
+---------------------------------------------+------------------
+-----------------------+------------------------+
|  vpc-cni                                    |  aws             |  eks
     |  networking             |
|  snapshot-controller                        |  aws             |  eks
     |  storage                |
|  kube-proxy                                 |  aws             |  eks
     |  networking             |
|  eks-pod-identity-agent                     |  aws             |  eks
     |  security               |
|  coredns                                    |  aws             |  eks
     |  networking             |
|  aws-mountpoint-s3-csi-driver               |  aws             |  s3
     |  storage                |
|  aws-guardduty-agent                        |  aws             |  eks
     |  security               |
|  aws-efs-csi-driver                         |  aws             |  eks
     |  storage                |
|  aws-ebs-csi-driver                         |  aws             |  eks
     |  storage                |
```

```
|  amazon-cloudwatch-observability           |  aws             |  eks
     |  observability          |
|  adot                                      |  aws             |  eks
     |  observability          |
|  upwind-security_upwind-operator           |  aws-marketplace |  Upwind Security
     |  security               |
|  upbound_universal-crossplane              |  aws-marketplace |  upbound
     |  infra-management       |
|  tetrate-io_istio-distro                   |  aws-marketplace |  tetrate-io
     |  policy-management      |
|  teleport_teleport                         |  aws-marketplace |  teleport
     |  policy-management      |
|  stormforge_optimize-live                  |  aws-marketplace |  StormForge
     |  cost-management        |
|  splunk_splunk-otel-collector-chart        |  aws-marketplace |  Splunk
     |  monitoring             |
|  solo-io_istio-distro                      |  aws-marketplace |  Solo.io
     |  service-mesh           |
|  rafay-systems_rafay-operator              |  aws-marketplace |  rafay-systems
     |  kubernetes-management  |
|  new-relic_kubernetes-operator             |  aws-marketplace |  New Relic
     |  observability          |
|  netapp_trident-operator                   |  aws-marketplace |  NetApp Inc.
     |  storage                |
|  leaksignal_leakagent                      |  aws-marketplace |  leaksignal
     |  monitoring             |
|  kubecost_kubecost                         |  aws-marketplace |  kubecost
     |  cost-management        |
|  kong_konnect-ri                           |  aws-marketplace |  kong
     |  ingress-service-type   |
|  kasten_k10                                |  aws-marketplace |  Kasten by Veeam
     |  data-protection        |
|  haproxy-technologies_kubernetes-ingress-ee |  aws-marketplace |  HAProxy
 Technologies |  ingress-controller     |
|  groundcover_agent                         |  aws-marketplace |  groundcover
     |  monitoring             |
|  grafana-labs_kubernetes-monitoring        |  aws-marketplace |  Grafana Labs
     |  monitoring             |
|  factorhouse_kpow                          |  aws-marketplace |  factorhouse
     |  monitoring             |
|  dynatrace_dynatrace-operator              |  aws-marketplace |  dynatrace
     |  monitoring             |
|  datree_engine-pro                         |  aws-marketplace |  datree
     |  policy-management      |
```

```
|  datadog_operator                              |  aws-marketplace |  Datadog
     |  monitoring             |
|  cribl_cribledge                               |  aws-marketplace |  Cribl
     |  observability          |
|  calyptia_fluent-bit                           |  aws-marketplace |  Calyptia Inc
     |  observability          |
|  accuknox_kubearmor                            |  aws-marketplace |  AccuKnox
     |  security               |
+------------------------------------------------+------------------
+----------------------+-------------------------+
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 2：列出 EKS 支持的指定 Kubernetes 版本的所有可用附加组件

以下 `describe-addon-versions` 示例列出了 EKS 支持的指定 Kubernetes 版本的所有可用附加组件。

```
aws eks describe-addon-versions  \
    --kubernetes-version=1.26 \
    --query 'sort_by(addons  &owner)[].{publisher: publisher, owner: owner,
 addonName: addonName, type: type}' \
    --output table
```

输出：

```
-------------------------------------------------------------------------------------------
|                                          DescribeAddonVersions
                              |
+------------------------------------------------+------------------
+----------------------+-------------------------+
|                  addonName                     |      owner       |       publisher
     |          type           |
+------------------------------------------------+------------------
+----------------------+-------------------------+
|  vpc-cni                                       |  aws             |  eks
     |  networking             |
|  snapshot-controller                           |  aws             |  eks
     |  storage                |
|  kube-proxy                                    |  aws             |  eks
     |  networking             |
```

```
|  eks-pod-identity-agent                      |  aws             |  eks
     |  security               |
|  coredns                                     |  aws             |  eks
     |  networking             |
|  aws-mountpoint-s3-csi-driver                |  aws             |  s3
     |  storage                |
|  aws-guardduty-agent                         |  aws             |  eks
     |  security               |
|  aws-efs-csi-driver                          |  aws             |  eks
     |  storage                |
|  aws-ebs-csi-driver                          |  aws             |  eks
     |  storage                |
|  amazon-cloudwatch-observability             |  aws             |  eks
     |  observability          |
|  adot                                        |  aws             |  eks
     |  observability          |
|  upwind-security_upwind-operator             |  aws-marketplace |  Upwind Security
     |  security               |
|  tetrate-io_istio-distro                     |  aws-marketplace |  tetrate-io
     |  policy-management      |
|  stormforge_optimize-live                    |  aws-marketplace |  StormForge
     |  cost-management        |
|  splunk_splunk-otel-collector-chart          |  aws-marketplace |  Splunk
     |  monitoring             |
|  solo-io_istio-distro                        |  aws-marketplace |  Solo.io
     |  service-mesh           |
|  rafay-systems_rafay-operator                |  aws-marketplace |  rafay-systems
     |  kubernetes-management  |
|  new-relic_kubernetes-operator               |  aws-marketplace |  New Relic
     |  observability          |
|  netapp_trident-operator                     |  aws-marketplace |  NetApp Inc.
     |  storage                |
|  leaksignal_leakagent                        |  aws-marketplace |  leaksignal
     |  monitoring             |
|  kubecost_kubecost                           |  aws-marketplace |  kubecost
     |  cost-management        |
|  kong_konnect-ri                             |  aws-marketplace |  kong
     |  ingress-service-type   |
|  haproxy-technologies_kubernetes-ingress-ee |  aws-marketplace |  HAProxy
 Technologies |  ingress-controller     |
|  groundcover_agent                           |  aws-marketplace |  groundcover
     |  monitoring             |
|  grafana-labs_kubernetes-monitoring          |  aws-marketplace |  Grafana Labs
     |  monitoring             |
```

```
|  dynatrace_dynatrace-operator                |  aws-marketplace |  dynatrace
     |  monitoring             |
|  datadog_operator                            |  aws-marketplace |  Datadog
     |  monitoring             |
|  cribl_cribledge                             |  aws-marketplace |  Cribl
     |  observability          |
|  calyptia_fluent-bit                         |  aws-marketplace |  Calyptia Inc
     |  observability          |
|  accuknox_kubearmor                          |  aws-marketplace |  AccuKnox
     |  security               |
+----------------------------------------------+------------------
+----------------------+-------------------------+
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

示例 3：列出 EKS 支持的指定 Kubernetes 版本的所有可用 vpc-cni 附加组件版本

以下 `describe-addon-versions` 示例列出了 EKS 支持的指定 Kubernetes 版本的所有可用 vpc-cni 附加组件版本。

```
aws eks describe-addon-versions \
    --kubernetes-version=1.26 \
    --addon-name=vpc-cni \
    --query='addons[].addonVersions[].addonVersion'
```

输出：

```
[
    "v1.18.0-eksbuild.1",
    "v1.17.1-eksbuild.1",
    "v1.16.4-eksbuild.2",
    "v1.16.3-eksbuild.2",
    "v1.16.2-eksbuild.1",
    "v1.16.0-eksbuild.1",
    "v1.15.5-eksbuild.1",
    "v1.15.4-eksbuild.1",
    "v1.15.3-eksbuild.1",
    "v1.15.1-eksbuild.1",
    "v1.15.0-eksbuild.2",
    "v1.14.1-eksbuild.1",
    "v1.14.0-eksbuild.3",
    "v1.13.4-eksbuild.1",
```

```
    "v1.13.3-eksbuild.1",
    "v1.13.2-eksbuild.1",
    "v1.13.0-eksbuild.1",
    "v1.12.6-eksbuild.2",
    "v1.12.6-eksbuild.1",
    "v1.12.5-eksbuild.2",
    "v1.12.0-eksbuild.2"
]
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 创建附加组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddonVersions。

describe-addon

以下代码示例演示了如何使用 describe-addon。

Amazon CLI

描述在 Amazon EKS 集群中主动运行的 EKS 附加组件

以下 describe-addon 示例描述在您的 Amazon EKS 集群中主动运行的 EKS 附加组件。

```
aws eks describe-addon \
    --cluster-name my-eks-cluster \
    --addon-name vpc-cni
```

输出：

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-eks-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.16.4-eksbuild.2",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/vpc-
cni/0ec71efc-98dd-3203-60b0-4b939b2a5e5f",
        "createdAt": "2024-03-14T13:18:45.417000-04:00",
        "modifiedAt": "2024-03-14T13:18:49.557000-04:00",
```

```
        "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-
cluster-addon-vpc-cni-Role1-YfakrqOC1UTm",
        "tags": {
            "eks-addon-key-3": "value-3",
            "eks-addon-key-4": "value-4"
        },
        "configurationValues": "resources:\n    limits:\n        cpu: '100m'\nenv:\n
    AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAddon。

describe-cluster

以下代码示例演示了如何使用 describe-cluster。

Amazon CLI

描述在 Amazon EKS 集群中主动运行的 EKS 附加组件

以下 describe-cluster 示例描述在您的 Amazon EKS 集群中主动运行的 EKS 附加组件。

```
aws eks describe-cluster \
    --name my-eks-cluster
```

输出：

```
{
    "cluster": {
        "name": "my-eks-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",
        "createdAt": "2024-03-14T11:31:44.348000-04:00",
        "version": "1.26",
        "endpoint": "https://JSA79429HJDASKJDJ8223829MNDNASW.yl4.us-
east-2.eks.amazonaws.com",
        "roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-
ServiceRole-zMF6CBakwwbW",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-0fb75d2d8401716e7",
                "subnet-02184492f67a3d0f9",
                "subnet-04098063527aab776",
```

```
                "subnet-0e2907431c9988b72",
                "subnet-04ad87f71c6e5ab4d",
                "subnet-09d912bb63ef21b9a"
            ],
            "securityGroupIds": [
                "sg-0c1327f6270afbb36"
            ],
            "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
            "vpcId": "vpc-0012b8e1cc0abb17d",
            "endpointPublicAccess": true,
            "endpointPrivateAccess": true,
            "publicAccessCidrs": [
                "22.19.18.2/32"
            ]
        },
        "kubernetesNetworkConfig": {
            "serviceIpv4Cidr": "10.100.0.0/16",
            "ipFamily": "ipv4"
        },
        "logging": {
            "clusterLogging": [
                {
                    "types": [
                        "api",
                        "audit",
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ],
                    "enabled": true
                }
            ]
        },
        "identity": {
            "oidc": {
                "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
JSA79429HJDASKJDJ8223829MNDNASW"
            }
        },
        "status": "ACTIVE",
        "certificateAuthority": {
            "data": "CA_DATA_STRING..."
        },
        "platformVersion": "eks.14",
```

```
        "tags": {
            "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
            "alpha.eksctl.io/cluster-name": "my-eks-cluster",
            "karpenter.sh/discovery": "my-eks-cluster",
            "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
            "auto-delete": "no",
            "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
            "EKS-Cluster-Name": "my-eks-cluster",
            "alpha.eksctl.io/cluster-oidc-enabled": "true",
            "aws:cloudformation:logical-id": "ControlPlane",
            "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
            "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
        },
        "health": {
            "issues": []
        },
        "accessConfig": {
            "authenticationMode": "API_AND_CONFIG_MAP"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCluster。

describe-fargate-profile

以下代码示例演示了如何使用 describe-fargate-profile。

Amazon CLI

描述 Fargate 配置文件

以下 describe-fargate-profile 示例描述了 Fargate 配置文件。

```
aws eks describe-fargate-profile \
    --cluster-name my-eks-cluster \
    --fargate-profile-name my-fargate-profile
```

输出：

```
{
    "fargateProfile": {
        "fargateProfileName": "my-fargate-profile",
        "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/96c766ce-43d2-f9c9-954c-647334391198",
        "clusterName": "my-eks-cluster",
        "createdAt": "2024-04-11T10:42:52.486000-04:00",
        "podExecutionRoleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-
cluster-farga-FargatePodExecutionRole-1htfAaJdJUEO",
        "subnets": [
            "subnet-09d912bb63ef21b9a",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-0e2907431c9988b72"
        ],
        "selectors": [
            {
                "namespace": "prod*",
                "labels": {
                    "labelname*?": "*value1"
                }
            },
            {
                "namespace": "*dev*",
                "labels": {
                    "labelname*?": "*value*"
                }
            }
        ],
        "status": "ACTIVE",
        "tags": {
            "eks-fargate-profile-key-2": "value-2",
            "eks-fargate-profile-key-1": "value-1"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFargateProfile。

describe-identity-provider-config

以下代码示例演示了如何使用 describe-identity-provider-config。

Amazon CLI

描述与 Amazon EKS 集群关联的身份提供商配置

以下 `describe-identity-provider-config` 示例描述了与您的 Amazon EKS 集群关联的身份提供商配置。

```
aws eks describe-identity-provider-config \
    --cluster-name my-eks-cluster \
    --identity-provider-config type=oidc,name=my-identity-provider
```

输出：

```
{
    "identityProviderConfig": {
        "oidc": {
            "identityProviderConfigName": "my-identity-provider",
            "identityProviderConfigArn": "arn:aws:eks:us-
east-2:111122223333:identityproviderconfig/my-eks-cluster/oidc/my-identity-
provider/8ac76722-78e4-cec1-ed76-d49eea058622",
            "clusterName": "my-eks-cluster",
            "issuerUrl": "https://oidc.eks.us-east-2.amazonaws.com/
id/38D6A4619A0A69E342B113ED7F1A7652",
            "clientId": "kubernetes",
            "usernameClaim": "email",
            "usernamePrefix": "my-username-prefix",
            "groupsClaim": "my-claim",
            "groupsPrefix": "my-groups-prefix",
            "requiredClaims": {
                "Claim1": "value1",
                "Claim2": "value2"
            },
            "tags": {
                "env": "dev"
            },
            "status": "ACTIVE"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的通过 OpenID Connect 身份提供商对集群的用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIdentityProviderConfig。

`describe-insight`

以下代码示例演示了如何使用 `describe-insight`。

Amazon CLI

使用 EKS 集群的 ID 获取该集群的见解的详细信息

以下 `describe-insight` 示例返回有关使用集群名称和见解 ID 指定的见解的详细信息。

```
aws eks describe-insight \
    --cluster-name eks-customer \
    --id 38ea7a64-a14f-4e0e-95c7-8dbcab3c3623
```

输出：

```
{
    "insight": {
        "id": "38ea7a64-a14f-4e0e-95c7-8dbcab3c3623",
        "name": "Kubelet version skew",
        "category": "UPGRADE_READINESS",
        "kubernetesVersion": "1.33",
        "lastRefreshTime": "2025-05-24T11:22:50-05:00",
        "lastTransitionTime": "2025-05-24T11:22:50-05:00",
        "description": "Checks for kubelet versions of worker nodes in the cluster
 to see if upgrade would cause noncompliance with supported Kubernetes kubelet
 version skew policy.",
        "insightStatus": {
            "status": "PASSING",
            "reason": "Node kubelet versions match the cluster control plane
 version."
        },
        "recommendation": "Upgrade your worker nodes to match the Kubernetes version
 of your cluster control plane.",
        "additionalInfo": {
            "Kubelet version skew policy": "https://kubernetes.io/releases/version-
skew-policy/#kubelet",
            "Updating a managed node group": "https://docs.aws.amazon.com/eks/
latest/userguide/update-managed-node-group.html"
        },
        "resources": [],
```

```
        "categorySpecificSummary": {
            "deprecationDetails": []
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的查看集群见解。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInsight。

describe-nodegroup

以下代码示例演示了如何使用 describe-nodegroup。

Amazon CLI

描述 Amazon EKS 集群的托管节点组

以下 describe-nodegroup 示例描述了 Amazon EKS 集群的托管节点组。

```
aws eks describe-nodegroup \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup
```

输出：

```
{
    "nodegroup": {
        "nodegroupName": "my-eks-nodegroup",
        "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/a8c75f2f-df78-a72f-4063-4b69af3de5b1",
        "clusterName": "my-eks-cluster",
        "version": "1.26",
        "releaseVersion": "1.26.12-20240329",
        "createdAt": "2024-04-08T11:42:10.555000-04:00",
        "modifiedAt": "2024-04-08T11:44:12.402000-04:00",
        "status": "ACTIVE",
        "capacityType": "ON_DEMAND",
        "scalingConfig": {
            "minSize": 1,
            "maxSize": 3,
            "desiredSize": 1
        },
```

```
        "instanceTypes": [
            "t3.medium"
        ],
        "subnets": [
            "subnet-0e2907431c9988b72",
            "subnet-04ad87f71c6e5ab4d",
            "subnet-09d912bb63ef21b9a"
        ],
        "amiType": "AL2_x86_64",
        "nodeRole": "arn:aws:iam::111122223333:role/role-name",
        "labels": {},
        "resources": {
            "autoScalingGroups": [
                {
                    "name": "eks-my-eks-nodegroup-a8c75f2f-df78-
a72f-4063-4b69af3de5b1"
                }
            ]
        },
        "diskSize": 20,
        "health": {
            "issues": []
        },
        "updateConfig": {
            "maxUnavailable": 1
        },
        "tags": {}
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNodegroup。

describe-pod-identity-association

以下代码示例演示了如何使用 describe-pod-identity-association。

Amazon CLI

提供有关容器组身份关联的详细信息

以下 describe-pod-identity-association 示例描述 EKS 集群中的容器组身份关联。

```
aws eks describe-pod-identity-association \
```

```
    --cluster-name eks-customer \
    --association-id a-9njjin9gfghecgocd
```

输出：

```
{
    "association": {
        "clusterName": "eks-customer",
        "namespace": "default",
        "serviceAccount": "default",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-9njjin9gfghecgocd",
        "associationId": "a-9njjin9gfghecgocd",
        "tags": {
            "Key2": "value2",
            "Key1": "value1"
        },
        "createdAt": "2025-05-24T19:52:14.135000-05:00",
        "modifiedAt": "2025-05-24T19:52:14.135000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribePodIdentityAssociation。

describe-update

以下代码示例演示了如何使用 `describe-update`。

Amazon CLI

示例 1：描述集群的更新

以下 `describe-update` 示例描述了指定集群的更新。

```
aws eks describe-update \
    --name my-eks-cluster \
    --update-id 10bddb13-a71b-425a-b0a6-71cd03e59161
```

输出：

```
{
    "update": {
        "id": "10bddb13-a71b-425a-b0a6-71cd03e59161",
        "status": "Successful",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "false"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            }
        ],
        "createdAt": "2024-03-14T10:01:26.297000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新 Amazon EKS 集群 Kubernetes 版本。

示例 2：描述集群的更新

以下 describe-update 示例描述了指定集群的更新。

```
aws eks describe-update \
    --name my-eks-cluster \
    --update-id e4994991-4c0f-475a-a040-427e6da52966
```

输出：

```
{
    "update": {
        "id": "e4994991-4c0f-475a-a040-427e6da52966",
        "status": "Successful",
        "type": "AssociateEncryptionConfig",
        "params": [
            {
                "type": "EncryptionConfig",
```

```
                "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
            }
        ],
        "createdAt": "2024-03-14T11:01:26.297000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新 Amazon EKS 集群 Kubernetes 版本。

示例 3：描述集群的更新

以下 `describe-update` 示例描述了指定集群的更新。

```
aws eks describe-update \
    --name my-eks-cluster \
    --update-id b5f0ba18-9a87-4450-b5a0-825e6e84496f
```

输出：

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "Successful",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.29"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
        "createdAt": "2024-03-14T12:05:26.297000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新 Amazon EKS 集群 Kubernetes 版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUpdate。

disassociate-access-policy

以下代码示例演示了如何使用 disassociate-access-policy。

Amazon CLI

取消访问策略与访问条目的关联

以下 disassociate-access-policy 移除与访问条目关联的访问策略。

```
aws eks disassociate-access-policy \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/Admin \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon EKS 用户指南》中的将访问策略与访问条目关联起来。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateAccessPolicy。

disassociate-identity-provider-config

以下代码示例演示了如何使用 disassociate-identity-provider-config。

Amazon CLI

取消身份提供商与 Amazon EKS 集群的关联

以下 disassociate-identity-provider-config 示例取消身份提供商与您的 Amazon EKS 集群的关联。

```
aws eks disassociate-identity-provider-config \
    --cluster-name my-eks-cluster \
    --identity-provider-config 'type=oidc,name=my-identity-provider'
```

输出：

```
{
```

```
    "update": {
        "id": "5f78d14e-c57b-4857-a3e4-cf664ae20949",
        "status": "InProgress",
        "type": "DisassociateIdentityProviderConfig",
        "params": [
            {
                "type": "IdentityProviderConfig",
                "value": "[]"
            }
        ],
        "createdAt": "2024-04-11T13:53:43.314000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的通过 OpenID Connect 身份提供商对集群的用户进行身份验证 – 取消 OIDC 身份提供商与集群的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateIdentityProviderConfig。

`get-token`

以下代码示例演示了如何使用 `get-token`。

Amazon CLI

示例 1：获取名为“my-eks-cluster”的 Amazon EKS 集群的身份验证令牌

以下 `get-token` 示例获取名为 my-eks-cluster 的 Amazon EKS 集群的身份验证令牌。

```
aws eks get-token \
    --cluster-name my-eks-cluster
```

输出：

```
{
    "kind": "ExecCredential",
    "apiVersion": "client.authentication.k8s.io/v1beta1",
    "spec": {},
    "status": {
        "expirationTimestamp": "2024-04-11T20:59:56Z",
        "token": "k8s-aws-v1.EXAMPLE_TOKEN_DATA_STRING..."
```

```
    }
}
```

示例 2：通过在签名令牌时为凭证代入此 roleARN，从而获取名为“my-eks-cluster”的 Amazon EKS 集群的身份验证令牌

以下 `get-token` 示例通过在签名令牌时为凭证代入此 roleARN，从而获取名为 my-eks-cluster 的 Amazon EKS 集群的身份验证令牌。

```
aws eks get-token \
    --cluster-name my-eks-cluster \
    --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-cluster-ServiceRole-j1k7AfTIQtnM
```

输出：

```
{
    "kind": "ExecCredential",
    "apiVersion": "client.authentication.k8s.io/v1beta1",
    "spec": {},
    "status": {
        "expirationTimestamp": "2024-04-11T21:05:26Z",
        "token": "k8s-aws-v1.EXAMPLE_TOKEN_DATA_STRING..."
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetToken。

list-access-entries

以下代码示例演示了如何使用 `list-access-entries`。

Amazon CLI

列出 EKS 集群的访问条目

以下 `list-access-entries` 返回与 EKS 集群 `eks-customer` 关联的访问条目的列表。

```
aws eks list-access-entries \
    --cluster-name eks-customer
```

输出：

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/Admin",
        "arn:aws:iam::111122223333:role/admin-test-ip",
        "arn:aws:iam::111122223333:role/assume-worker-node-role",
        "arn:aws:iam::111122223333:user/eks-admin-user"
    ]
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的使用 EKS 访问条目向 IAM 用户授予 Kubernetes 访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAccessEntries。

list-access-policies

以下代码示例演示了如何使用 list-access-policies。

Amazon CLI

列出所有可用的访问策略

此 list-access-policies 示例返回所有可用访问策略的列表。

```
aws eks list-access-policies
```

输出：

```
{
    "accessPolicies": [
        {
            "name": "AmazonEKSAdminPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy"
        },
        {
            "name": "AmazonEKSAdminViewPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy"
        },
        {
            "name": "AmazonEKSAutoNodePolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy"
        },
```

```
        {
            "name": "AmazonEKSBlockStorageClusterPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSBlockStorageClusterPolicy"
        },
        {
            "name": "AmazonEKSBlockStoragePolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSBlockStoragePolicy"
        },
        {
            "name": "AmazonEKSClusterAdminPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSClusterAdminPolicy"
        },
        {
            "name": "AmazonEKSComputeClusterPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSComputeClusterPolicy"
        },
        {
            "name": "AmazonEKSComputePolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSComputePolicy"
        },
        {
            "name": "AmazonEKSEditPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy"
        },
        {
            "name": "AmazonEKSHybridPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSHybridPolicy"
        },
        {
            "name": "AmazonEKSLoadBalancingClusterPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSLoadBalancingClusterPolicy"
        },
        {
            "name": "AmazonEKSLoadBalancingPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSLoadBalancingPolicy"
        },
        {
            "name": "AmazonEKSNetworkingClusterPolicy",
```

```
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSNetworkingClusterPolicy"
        },
        {
            "name": "AmazonEKSNetworkingPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSNetworkingPolicy"
        },
        {
            "name": "AmazonEKSViewPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy"
        },
        {
            "name": "AmazonEMRJobPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEMRJobPolicy"
        },
        {
            "name": "AmazonSagemakerHyperpodClusterPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonSagemakerHyperpodClusterPolicy"
        },
        {
            "name": "AmazonSagemakerHyperpodControllerPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonSagemakerHyperpodControllerPolicy"
        },
        {
            "name": "AmazonSagemakerHyperpodSystemNamespacePolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonSagemakerHyperpodSystemNamespacePolicy"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的将访问策略与访问条目关联起来。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessPolicies。

list-addons

以下代码示例演示了如何使用 list-addons。

Amazon CLI

列出名为“my-eks-cluster”的 Amazon EKS 集群中所有已安装的附加组件

以下 `list-addons` 示例列出了名为 my-eks-cluster 的 Amazon EKS 集群中所有已安装的附加组件。

```
aws eks list-addons \
    --cluster-name my-eks-cluster
```

输出：

```
{
    "addons": [
        "kube-proxy",
        "vpc-cni"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAddons。

list-associated-access-policies

以下代码示例演示了如何使用 `list-associated-access-policies`。

Amazon CLI

列出与访问条目关联的访问策略

以下 `list-associated-access-policies` 示例返回与 EKS 集群中的访问条目关联的访问策略列表。

```
aws eks list-associated-access-policies \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/Admin
```

输出：

```
{
    "associatedAccessPolicies": [
        {
```

```
            "policyArn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2025-05-24T17:26:22.935000-05:00",
            "modifiedAt": "2025-05-24T17:26:22.935000-05:00"
        }
    ],
    "clusterName": "eks-customer",
    "principalArn": "arn:aws:iam::111122223333:role/Admin"
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的使用 EKS 访问条目向 IAM 用户授予 Kubernetes 访问权限。

- 有关 API 详细信息，请参阅 Amazon CLI Command Reference 中的 ListAssociatedAccessPolicies。

`list-clusters`

以下代码示例演示了如何使用 `list-clusters`。

Amazon CLI

列出名为“my-eks-cluster”的 Amazon EKS 集群中所有已安装的附加组件

以下 `list-clusters` 示例列出了名为 my-eks-cluster 的 Amazon EKS 集群中所有已安装的附加组件。

```
aws eks list-clusters
```

输出：

```
{
    "clusters": [
        "prod",
        "qa",
        "stage",
        "my-eks-cluster"
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListClusters。

`list-fargate-profiles`

以下代码示例演示了如何使用 `list-fargate-profiles`。

Amazon CLI

列出名为“my-eks-cluster”的 Amazon EKS 集群中的所有 Fargate 配置文件

以下 `list-fargate-profiles` 示例列出了名为 my-eks-cluster 的 Amazon EKS 集群中的所有 Fargate 配置文件。

```
aws eks list-fargate-profiles \
    --cluster-name my-eks-cluster
```

输出：

```
{
    "fargateProfileNames": [
        "my-fargate-profile"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFargateProfiles。

`list-identity-provider-configs`

以下代码示例演示了如何使用 `list-identity-provider-configs`。

Amazon CLI

列出与 Amazon EKS 集群关联的身份提供商

以下 `list-identity-provider-configs` 示例列出了与 Amazon EKS 集群关联的身份提供商。

```
aws eks list-identity-provider-configs \
    --cluster-name my-eks-cluster
```

输出：

```
{
    "identityProviderConfigs": [
        {
            "type": "oidc",
            "name": "my-identity-provider"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的通过 OpenID Connect 身份提供商对集群的用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIdentityProviderConfigs。

`list-insights`

以下代码示例演示了如何使用 `list-insights`。

Amazon CLI

列出指定集群的所有见解

以下 `list-insights` 示例返回针对指定的集群检查的所有见解的列表。

```
aws eks list-insights \
--cluster-name eks-customer
```

输出：

```
{
    "insights": [
        {
            "id": "38ea7a64-a14f-4e0e-95c7-8dbcab3c3616",
            "name": "Kubelet version skew",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "1.33",
            "lastRefreshTime": "2025-05-24T11:22:50-05:00",
            "lastTransitionTime": "2025-05-24T11:22:50-05:00",
            "description": "Checks for kubelet versions of worker nodes in the
 cluster to see if upgrade would cause noncompliance with supported Kubernetes
 kubelet version skew policy.",
```

```
            "insightStatus": {
                "status": "PASSING",
                "reason": "Node kubelet versions match the cluster control plane
version."
            }
        },
        {
            "id": "9cd91472-f99c-45a9-b7d7-54d4900dee23",
            "name": "EKS add-on version compatibility",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "1.33",
            "lastRefreshTime": "2025-05-24T11:22:59-05:00",
            "lastTransitionTime": "2025-05-24T11:22:50-05:00",
            "description": "Checks version of installed EKS add-ons to ensure they
are compatible with the next version of Kubernetes. ",
            "insightStatus": {
                "status": "PASSING",
                "reason": "All installed EKS add-on versions are compatible with
next Kubernetes version."
            }
        },
        {
            "id": "0deb269d-b1e1-458c-a2b4-7a57f940c875",
            "name": "Cluster health issues",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "1.33",
            "lastRefreshTime": "2025-05-24T11:22:59-05:00",
            "lastTransitionTime": "2025-05-24T11:22:50-05:00",
            "description": "Checks for any cluster health issues that prevent
successful upgrade to the next Kubernetes version on EKS.",
            "insightStatus": {
                "status": "PASSING",
                "reason": "No cluster health issues detected."
            }
        },
        {
            "id": "214fa274-344f-420b-812a-5049ce72c9ww",
            "name": "kube-proxy version skew",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "1.33",
            "lastRefreshTime": "2025-05-24T11:22:50-05:00",
            "lastTransitionTime": "2025-05-24T11:22:50-05:00",
```

```
            "description": "Checks version of kube-proxy in cluster to see if
 upgrade would cause noncompliance with supported Kubernetes kube-proxy version skew
 policy.",
            "insightStatus": {
                "status": "PASSING",
                "reason": "kube-proxy versions match the cluster control plane
 version."
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的查看集群见解。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListInsights。

`list-nodegroups`

以下代码示例演示了如何使用 `list-nodegroups`。

Amazon CLI

列出 Amazon EKS 集群中的所有节点组

以下 `list-nodegroups` 示例列出了 Amazon EKS 集群中的所有节点组。

```
aws eks list-nodegroups \
    --cluster-name my-eks-cluster
```

输出：

```
{
    "nodegroups": [
        "my-eks-managed-node-group",
        "my-eks-nodegroup"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListNodegroups。

list-pod-identity-associations

以下代码示例演示了如何使用 list-pod-identity-associations。

Amazon CLI

示例 1：列出 EKS 集群中的容器组身份关联

以下 list-pod-identity-associations 返回与所有命名空间和服务账户中名为 eks-customer 的 EKS 集群关联的容器组身份关联的列表。

```
aws eks list-pod-identity-associations \
    --cluster-name eks-customer
```

输出：

```
{
    "associations": [
        {
            "clusterName": "eks-customer",
            "namespace": "default",
            "serviceAccount": "default",
            "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-9njjin9gfghecgocd",
            "associationId": "a-9njjin9gfghecgocd"
        },
        {
            "clusterName": "eks-customer",
            "namespace": "kube-system",
            "serviceAccount": "eks-customer",
            "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-dvtacahdvjn01ffbc",
            "associationId": "a-dvtacahdvjn01ffbc"
        },
        {
            "clusterName": "eks-customer",
            "namespace": "kube-system",
            "serviceAccount": "coredns",
            "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-yrpsdroc4ei7k6xps",
            "associationId": "a-yrpsdroc4ei7k6xps"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

示例 2：根据命名空间和服务账户列出 EKS 集群中的容器组身份关联

以下 `list-pod-identity-associations` 根据命名空间和服务账户返回 EKS 集群中的容器组身份关联的列表。

```
aws eks list-pod-identity-associations \
    --cluster-name eks-customer \
    --namespace kube-system \
    --service-account eks-customer
```

输出：

```
{
    "associations": [
        {
            "clusterName": "eks-customer",
            "namespace": "kube-system",
            "serviceAccount": "eks-customer",
            "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-dvtacahdvjn01ffbc",
            "associationId": "a-dvtacahdvjn01ffbc"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListPodIdentityAssociations。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

示例 1：列出 Amazon EKS 集群 ARN 的所有标签

以下 `list-tags-for-resource` 示例列出了 Amazon EKS 集群 ARN 的所有标签。

```
aws eks list-tags-for-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster
```

输出：

```
{
    "tags": {
        "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
        "alpha.eksctl.io/cluster-name": "my-eks-cluster",
        "karpenter.sh/discovery": "my-eks-cluster",
        "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
        "auto-delete": "no",
        "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
        "EKS-Cluster-Name": "my-eks-cluster",
        "alpha.eksctl.io/cluster-oidc-enabled": "true",
        "aws:cloudformation:logical-id": "ControlPlane",
        "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
        "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
    }
}
```

示例 2：列出 Amazon EKS 节点组 ARN 的所有标签

以下 `list-tags-for-resource` 示例列出了 Amazon EKS 节点组 ARN 的所有标签。

```
aws eks list-tags-for-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-
eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c
```

输出：

```
{
    "tags": {
```

```
        "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-nodegroup-my-eks-
managed-node-group",
        "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-nodegroup-my-eks-managed-node-group/
eaa20310-e219-11ee-b851-0ab9ad8228ff",
        "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
        "EKS-Cluster-Name": "my-eks-cluster",
        "alpha.eksctl.io/nodegroup-type": "managed",
        "NodeGroup Name 1": "my-eks-managed-node-group",
        "k8s.io/cluster-autoscaler/enabled": "true",
        "nodegroup-role": "worker",
        "alpha.eksctl.io/cluster-name": "my-eks-cluster",
        "alpha.eksctl.io/nodegroup-name": "my-eks-managed-node-group",
        "karpenter.sh/discovery": "my-eks-cluster",
        "NodeGroup Name 2": "AmazonLinux-Linux-Managed-NG-v1-26-v1",
        "auto-delete": "no",
        "k8s.io/cluster-autoscaler/my-eks-cluster": "owned",
        "aws:cloudformation:logical-id": "ManagedNodeGroup",
        "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z"
    }
}
```

示例 3：列出 Amazon EKS Fargate 配置文件 ARN 上的所有标签

以下 `list-tags-for-resource` 示例列出了 Amazon EKS Fargate 配置文件 ARN 上的所有标签。

```
aws eks list-tags-for-resource  \
    --resource-arn arn:aws:eks:us-east-2:111122223333:fargateprofile/my-eks-cluster/
my-fargate-profile/d6c76780-e541-0725-c816-36754cab734b
```

输出：

```
{
    "tags": {
        "eks-fargate-profile-key-2": "value-2",
        "eks-fargate-profile-key-1": "value-1"
    }
}
```

示例 4：列出 Amazon EKS 附加组件 ARN 的所有标签

以下 `list-tags-for-resource` 示例列出了 Amazon EKS 附加组件 ARN 的所有标签。

```
aws eks list-tags-for-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/vpc-cni/0ec71efc-98dd-3203-60b0-4b939b2a5e5f
```

输出：

```
{
    "tags": {
        "eks-addon-key-2": "value-2",
        "eks-addon-key-1": "value-1"
    }
}
```

示例 5：列出 Amazon EKS OIDC 身份提供商 ARN 的所有标签

以下 `list-tags-for-resource` 示例列出了 Amazon EKS OIDC 身份提供商 ARN 的所有标签。

```
aws eks list-tags-for-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:identityproviderconfig/my-eks-cluster/oidc/my-identity-provider/8ac76722-78e4-cec1-ed76-d49eea058622
```

输出：

```
{
    "tags": {
        "my-identity-provider": "test"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-update

以下代码示例演示了如何使用 `list-update`。

Amazon CLI

示例 1：列出与 Amazon EKS 集群名称相关的更新

以下 `list-updates` 示例列出了 Amazon EKS 集群名称的所有更新 ID。

```
aws eks list-updates \
    --name my-eks-cluster
```

输出：

```
{
    "updateIds": [
        "5f78d14e-c57b-4857-a3e4-cf664ae20949",
        "760e5a3f-adad-48c7-88d3-7ac283c09c26",
        "cd4ec863-bc55-47d5-a377-3971502f529b",
        "f12657ce-e869-4f17-b158-a82ab8b7d937"
    ]
}
```

示例 2：列出 Amazon EKS 节点组的所有更新 ID

以下 `list-updates` 示例列出了 Amazon EKS 节点组的所有更新 ID。

```
aws eks list-updates \
    --name my-eks-cluster \
    --nodegroup-name my-eks-managed-node-group
```

输出：

```
{
    "updateIds": [
        "8c6c1bef-61fe-42ac-a242-89412387b8e7"
    ]
}
```

示例 3：列出 Amazon EKS 附加组件上的所有更新 ID

以下 `list-updates` 示例列出了 Amazon EKS 附加组件的所有更新 ID。

```
aws eks list-updates \
    --name my-eks-cluster \
    --addon-name vpc-cni
```

输出：

```
{
    "updateIds": [
        "9cdba8d4-79fb-3c83-afe8-00b508d33268"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUpdate。

`list-updates`

以下代码示例演示了如何使用 `list-updates`。

Amazon CLI

列出集群的更新

此示例命令列出了默认区域中名为 `example` 的集群的当前更新。

命令:

```
aws eks list-updates --name example
```

输出：

```
{
    "updateIds": [
        "10bddb13-a71b-425a-b0a6-71cd03e59161"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUpdates。

`register-cluster`

以下代码示例演示了如何使用 `register-cluster`。

Amazon CLI

示例 1：将外部 EKS_ANYWHERE Kubernetes 集群注册到 Amazon EKS

以下 register-cluster 示例将外部 EKS_ANYWHERE Kubernetes 集群注册到 Amazon EKS。

```
aws eks register-cluster \
    --name my-eks-anywhere-cluster \
    --connector-config 'roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider=EKS_ANYWHERE'
```

输出：

```
{
    "cluster": {
        "name": "my-eks-anywhere-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-anywhere-cluster",
        "createdAt": "2024-04-12T12:38:37.561000-04:00",
        "status": "PENDING",
        "tags": {},
        "connectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": "2024-04-15T12:38:37.082000-04:00",
            "provider": "EKS_ANYWHERE",
            "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSConnectorAgentRole"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的连接外部集群。

示例 2：将任何外部 Kubernetes 集群注册到 Amazon EKS

以下 register-cluster 示例将外部 EKS_ANYWHERE Kubernetes 集群注册到 Amazon EKS。

```
aws eks register-cluster \
    --name my-eks-anywhere-cluster \
    --connector-config 'roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider=OTHER'
```

输出：

```
{
```

```
    "cluster": {
        "name": "my-onprem-k8s-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-onprem-k8s-cluster",
        "createdAt": "2024-04-12T12:42:10.861000-04:00",
        "status": "PENDING",
        "tags": {},
        "connectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": "2024-04-15T12:42:10.339000-04:00",
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSConnectorAgentRole"
        }
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的连接外部集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterCluster。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

示例 1：将指定的标签添加到 Amazon EKS 集群

以下 `tag-resource` 示例将指定的标签添加到 Amazon EKS 集群。

```
aws eks tag-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster \
    --tag 'my-eks-cluster-test-1=test-value-1,my-eks-cluster-dev-1=dev-value-2'
```

此命令不生成任何输出。

示例 2：将指定的标签添加到 Amazon EKS 节点组

以下 `tag-resource` 示例将指定的标签添加到 Amazon EKS 节点组。

```
aws eks tag-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-
eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c \
```

```
    --tag 'my-eks-nodegroup-test-1=test-value-1,my-eks-nodegroup-dev-1=dev-value-2'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：从 Amazon EKS 集群中删除指定的标签

以下 untag-resource 示例从 Amazon EKS 集群中删除指定的标签。

```
aws eks untag-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster \
    --tag-keys "my-eks-cluster-test-1" "my-eks-cluster-dev-1"
```

此命令不生成任何输出。

示例 2：从 Amazon EKS 节点组中删除指定的标签

以下 untag-resource 示例从 Amazon EKS 节点组中删除指定的标签。

```
aws eks untag-resource \
    --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c \
    --tag-keys "my-eks-nodegroup-test-1" "my-eks-nodegroup-dev-1"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-access-entry

以下代码示例演示了如何使用 update-access-entry。

Amazon CLI

更新 EKS 集群的访问条目

以下 update-access-entry 通过添加 Kubernetes 组 tester 来更新 EKS 集群的访问条目。

```
aws eks update-access-entry \
    --cluster-name eks-customer \
    --principal-arn arn:aws:iam::111122223333:role/Admin \
    --kubernetes-groups tester
```

输出：

```
{
    "accessEntry": {
        "clusterName": "eks-customer",
        "principalArn": "arn:aws:iam::111122223333:role/Admin",
        "kubernetesGroups": [
            "tester"
        ],
        "accessEntryArn": "arn:aws:eks:us-west-2:111122223333:access-entry/eks-
customer/role/111122223333/Admin/d2cb8183-d6ec-b82a-d967-eca21902a4b4",
        "createdAt": "2025-05-24T11:02:04.432000-05:00",
        "modifiedAt": "2025-05-24T17:08:01.608000-05:00",
        "tags": {},
        "username": "arn:aws:sts::111122223333:assumed-role/Admin/{{SessionName}}",
        "type": "STANDARD"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新访问条目。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateAccessEntry。

update-addon

以下代码示例演示了如何使用 update-addon。

Amazon CLI

示例 1：使用服务账户角色 ARN 更新 Amazon EKS 附加组件

以下 update-addon 示例命令使用服务账户角色 ARN 更新 Amazon EKS 附加组件。

```
aws eks update-addon \
    --cluster-name my-eks-cluster \
```

```
    --addon-name vpc-cni \
    --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-addon-vpc-cni-Role1-YfakrqOC1UTm
```

输出：

```
{
    "update": {
        "id": "c00d2de2-c2e4-3d30-929e-46b8edec2ce4",
        "status": "InProgress",
        "type": "AddonUpdate",
        "params": [
            {
                "type": "ServiceAccountRoleArn",
                "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-addon-vpc-cni-Role1-YfakrqOC1UTm"
            }
        ],
        "updatedAt": "2024-04-12T16:04:55.614000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 更新附加组件。

示例 2：使用特定的附加组件版本更新 Amazon EKS 附加组件

以下 update-addon 示例命令使用特定的附加组件版本更新 Amazon EKS 附加组件。

```
aws eks update-addon \
    --cluster-name my-eks-cluster \
    --addon-name vpc-cni \
    --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-addon-vpc-cni-Role1-YfakrqOC1UTm \
    --addon-version v1.16.4-eksbuild.2
```

输出：

```
{
    "update": {
        "id": "f58dc0b0-2b18-34bd-bc6a-e4abc0011f36",
```

```
        "status": "InProgress",
        "type": "AddonUpdate",
        "params": [
            {
                "type": "AddonVersion",
                "value": "v1.16.4-eksbuild.2"
            },
            {
                "type": "ServiceAccountRoleArn",
                "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm"
            }
        ],
        "createdAt": "2024-04-12T16:07:16.550000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 更新附加组件。

示例 3：使用自定义配置值更新 Amazon EKS 附加组件并解决冲突详细信息

以下 update-addon 示例命令使用自定义配置值更新 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks update-addon \
    --cluster-name my-eks-cluster \
    --addon-name vpc-cni \
    --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm \
    --addon-version v1.16.4-eksbuild.2 \
    --configuration-values '{"resources": {"limits":{"cpu":"100m"}, "requests":
{"cpu":"50m"}}}' \
    --resolve-conflicts PRESERVE
```

输出：

```
{
    "update": {
        "id": "cd9f2173-a8d8-3004-a90f-032f14326520",
        "status": "InProgress",
        "type": "AddonUpdate",
```

```
        "params": [
            {
                "type": "AddonVersion",
                "value": "v1.16.4-eksbuild.2"
            },
            {
                "type": "ServiceAccountRoleArn",
                "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm"
            },
            {
                "type": "ResolveConflicts",
                "value": "PRESERVE"
            },
            {
                "type": "ConfigurationValues",
                "value": "{\"resources\": {\"limits\":{\"cpu\":\"100m\"}, \"requests
\":{\"cpu\":\"50m\"}}}"
            }
        ],
        "createdAt": "2024-04-12T16:16:27.363000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 更新附加组件。

示例 4：使用自定义 JSON 配置值文件更新 Amazon EKS 附加组件

以下 update-addon 示例命令使用自定义 JSON 配置值更新 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks update-addon \
    --cluster-name my-eks-cluster \
    --addon-name vpc-cni \
    --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm \
    --addon-version v1.17.1-eksbuild.1 \
    --configuration-values 'file://configuration-values.json' \
    --resolve-conflicts PRESERVE
```

的内容configuration-values.json：

```
{
    "resources": {
        "limits": {
            "cpu": "100m"
        },
        "requests": {
            "cpu": "50m"
        }
    },
    "env": {
        "AWS_VPC_K8S_CNI_LOGLEVEL": "ERROR"
    }
}
```

输出：

```
{
    "update": {
        "id": "6881a437-174f-346b-9a63-6e91763507cc",
        "status": "InProgress",
        "type": "AddonUpdate",
        "params": [
            {
                "type": "AddonVersion",
                "value": "v1.17.1-eksbuild.1"
            },
            {
                "type": "ServiceAccountRoleArn",
                "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm"
            },
            {
                "type": "ResolveConflicts",
                "value": "PRESERVE"
            },
            {
                "type": "ConfigurationValues",
                "value": "{\n    \"resources\": {\n        \"limits\": {\n
  \"cpu\": \"100m\"\n        },\n        \"requests\": {\n            \"cpu\": \"50m
\"\n        }\n    },\n    \"env\": {\n        \"AWS_VPC_K8S_CNI_LOGLEVEL\": \"ERROR
\"\n    }\n}"
            }
        ],
```

```
        "createdAt": "2024-04-12T16:22:55.519000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 更新附加组件。

示例 5：使用自定义 YAML 配置值文件更新 Amazon EKS 附加组件

以下 update-addon 示例命令使用自定义 YAML 配置值更新 Amazon EKS 附加组件并解决冲突详细信息。

```
aws eks update-addon \
    --cluster-name my-eks-cluster \
    --addon-name vpc-cni \
    --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-addon-vpc-cni-Role1-YfakrqOC1UTm \
    --addon-version v1.18.0-eksbuild.1 \
    --configuration-values 'file://configuration-values.yaml' \
    --resolve-conflicts PRESERVE
```

的内容configuration-values.yaml：

```
resources:
    limits:
        cpu: '100m'
    requests:
        cpu: '50m'
env:
    AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'
```

输出：

```
{
    "update": {
        "id": "a067a4c9-69d0-3769-ace9-d235c5b16701",
        "status": "InProgress",
        "type": "AddonUpdate",
        "params": [
            {
                "type": "AddonVersion",
```

```
                "value": "v1.18.0-eksbuild.1"
            },
            {
                "type": "ServiceAccountRoleArn",
                "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-YfakrqOC1UTm"
            },
            {
                "type": "ResolveConflicts",
                "value": "PRESERVE"
            },
            {
                "type": "ConfigurationValues",
                "value": "resources:\n    limits:\n        cpu: '100m'\n
 requests:\n        cpu: '50m'\nenv:\n    AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'"
            }
        ],
        "createdAt": "2024-04-12T16:25:07.212000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的管理 Amazon EKS 附加组件 – 更新附加组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAddon。

update-cluster-config

以下代码示例演示了如何使用 update-cluster-config。

Amazon CLI

更新集群端点访问权限

此示例命令更新集群以禁用端点公有访问权限并启用私有端点访问权限。

命令:

```
aws eks update-cluster-config --name example \
--resources-vpc-config endpointPublicAccess=false,endpointPrivateAccess=true
```

输出：

```
{
    "update": {
        "id": "ec883c93-2e9e-407c-a22f-8f6fa6e67d4f",
        "status": "InProgress",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "false"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            }
        ],
        "createdAt": 1565806986.506,
        "errors": []
    }
}
```

为集群启用日志记录

此示例命令为名为 example 的集群启用所有集群控制面板日志记录类型。

命令:

```
aws eks update-cluster-config --name example \
--logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

输出：

```
{
    "update": {
        "id": "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd",
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
```

```
        ],
        "createdAt": 1565807210.37,
        "errors": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateClusterConfig。

update-cluster-version

以下代码示例演示了如何使用 update-cluster-version。

Amazon CLI

将名为“my-eks-cluster”的 Amazon EKS 集群更新为指定的 Kubernetes 版本

以下 update-cluster-version 示例将 Amazon EKS 集群更新为指定的 Kubernetes 版本。

```
aws eks update-cluster-version \
    --name my-eks-cluster \
    --kubernetes-version 1.27
```

输出：

```
{
    "update": {
        "id": "e4091a28-ea14-48fd-a8c7-975aeb469e8a",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.27"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.16"
            }
        ],
        "createdAt": "2024-04-12T16:56:01.082000-04:00",
        "errors": []
    }
```

```
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新 Amazon EKS 集群 Kubernetes 版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateClusterVersion。

update-kubeconfig

以下代码示例演示了如何使用 update-kubeconfig。

Amazon CLI

示例 1：通过创建或更新 kubeconfig 来配置 kubectl，以便您可以连接到名为“my-eks-cluster”的 Amazon EKS 集群

以下 update-kubeconfig 示例通过创建或更新 kubeconfig 来配置 kubectl，以便您可以连接到名为 my-eks-cluster 的 Amazon EKS 集群。

```
aws eks update-kubeconfig \
    --name my-eks-cluster
```

输出：

```
Updated context arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster in /Users/
xxx/.kube/config
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的为 Amazon EKS 集群创建或更新 kubeconfig 文件。

示例 2：通过创建或更新 kubeconfig（使用可代入集群身份验证角色的 role-arn 选项）来配置 kubectl，以便您可以连接到名为“my-eks-cluster”的 Amazon EKS 集群

以下 update-kubeconfig 示例通过创建或更新 kubeconfig（使用可代入集群身份验证角色的 role-arn 选项）来配置 kubectl，以便您可以连接到名为 my-eks-cluster 的 Amazon EKS 集群。

```
aws eks update-kubeconfig \
    --name my-eks-cluster \
    --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-
cluster-ServiceRole-j1k7AfTIQtnM
```

输出：

```
Updated context arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster in /Users/
xxx/.kube/config
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的为 Amazon EKS 集群创建或更新 kubeconfig 文件。

示例 3：通过创建或更新 kubeconfig（使用可代入集群身份验证角色的 role-arn 选项以及自定义集群别名和用户别名）来配置 kubectl，以便您可以连接到名为“my-eks-cluster”的 Amazon EKS 集群

以下 `update-kubeconfig` 示例通过创建或更新 kubeconfig（使用可代入集群身份验证角色的 role-arn 选项以及自定义集群别名和用户别名）来配置 kubectl，以便您可以连接到名为 my-eks-cluster 的 Amazon EKS 集群。

```
aws eks update-kubeconfig \
    --name my-eks-cluster \
    --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-
cluster-ServiceRole-j1k7AfTIQtnM \
    --alias stage-eks-cluster \
    --user-alias john
```

输出：

```
Updated context stage-eks-cluster in /Users/dubaria/.kube/config
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的为 Amazon EKS 集群创建或更新 kubeconfig 文件。

示例 4：打印 kubeconfig 文件条目以供查看并配置您的 kubectl，以便您可以连接到名为“my-eks-cluster”的 Amazon EKS 集群

以下 `update-kubeconfig` 示例通过创建或更新 kubeconfig（使用可代入集群身份验证角色的 role-arn 选项以及自定义集群别名和用户别名）来配置 kubectl，以便您可以连接到名为 my-eks-cluster 的 Amazon EKS 集群。

```
aws eks update-kubeconfig \
    --name my-eks-cluster \
    --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-
cluster-ServiceRole-j1k7AfTIQtnM \
    --alias stage-eks-cluster \
    --user-alias john \
```

```
    --verbose
```

输出：

```
Updated context stage-eks-cluster in /Users/dubaria/.kube/config
Entries:

context:
cluster: arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster
user: john
name: stage-eks-cluster

name: john
user:
exec:
    apiVersion: client.authentication.k8s.io/v1beta1
    args:
    - --region
    - us-east-2
    - eks
    - get-token
    - --cluster-name
    - my-eks-cluster
    - --output
    - json
    - --role
    - arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-cluster-
ServiceRole-j1k7AfTIQtnM
    command: aws

cluster:
certificate-authority-data: xxx_CA_DATA_xxx
server: https://DALSJ343KE23J3RN45653DSKJTT647TYD.yl4.us-east-2.eks.amazonaws.com
name: arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的为 Amazon EKS 集群创建或更新 kubeconfig 文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateKubeconfig。

update-nodegroup-config

以下代码示例演示了如何使用 update-nodegroup-config。

Amazon CLI

示例 1：更新托管节点组以向 Amazon EKS 集群的 EKS Worker 节点添加新标签和污点

以下 update-nodegroup-config 示例更新托管节点组以向 Amazon EKS 集群的 EKS Worker 节点添加新标签和污点。

```
aws eks update-nodegroup-config \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --labels 'addOrUpdateLabels={my-eks-nodegroup-label-1=value-1,my-eks-nodegroup-label-2=value-2}' \
    --taints 'addOrUpdateTaints=[{key=taint-key-1,value=taint-value-1,effect=NO_EXECUTE}]'
```

输出：

```
{
    "update": {
        "id": "e66d21d3-bd8b-3ad1-a5aa-b196dc08c7c1",
        "status": "InProgress",
        "type": "ConfigUpdate",
        "params": [
            {
                "type": "LabelsToAdd",
                "value": "{\"my-eks-nodegroup-label-2\":\"value-2\",\"my-eks-nodegroup-label-1\":\"value-1\"}"
            },
            {
                "type": "TaintsToAdd",
                "value": "[{\"effect\":\"NO_EXECUTE\",\"value\":\"taint-value-1\",\"key\":\"taint-key-1\"}]"
            }
        ],
        "createdAt": "2024-04-08T12:05:19.161000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新托管节点组。

示例 2：更新托管节点组以删除 Amazon EKS 集群的 EKS Worker 节点的标签和污点

以下 update-nodegroup-config 示例更新托管节点组以删除 Amazon EKS 集群的 EKS Worker 节点的标签和污点。

```
aws eks update-nodegroup-config \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --labels 'removeLabels=my-eks-nodegroup-label-1, my-eks-nodegroup-label-2' \
    --taints 'removeTaints=[{key=taint-key-1,value=taint-
value-1,effect=NO_EXECUTE}]'
```

输出：

```
{
    "update": {
        "id": "67a08692-9e59-3ace-a916-13929f44cec3",
        "status": "InProgress",
        "type": "ConfigUpdate",
        "params": [
            {
                "type": "LabelsToRemove",
                "value": "[\"my-eks-nodegroup-label-1\",\"my-eks-nodegroup-
label-2\"]"
            },
            {
                "type": "TaintsToRemove",
                "value": "[{\"effect\":\"NO_EXECUTE\",\"value\":\"taint-value-1\",
\"key\":\"taint-key-1\"}]"
            }
        ],
        "createdAt": "2024-04-08T12:17:31.817000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新托管节点组。

示例 3：更新托管节点组以删除和添加 Amazon EKS 集群的 EKS Worker 节点的标签和污点

以下 update-nodegroup-config 示例更新托管节点组以删除和添加 Amazon EKS 集群的 EKS Worker 节点的标签和污点。

```
aws eks update-nodegroup-config \
    --cluster-name my-eks-cluster \
```

```
    --nodegroup-name my-eks-nodegroup \
    --labels 'addOrUpdateLabels={my-eks-nodegroup-new-label-1=new-value-1,my-eks-nodegroup-new-label-2=new-value-2},removeLabels=my-eks-nodegroup-label-1, my-eks-nodegroup-label-2' \
    --taints 'addOrUpdateTaints=[{key=taint-new-key-1,value=taint-new-value-1,effect=PREFER_NO_SCHEDULE}],removeTaints=[{key=taint-key-1,value=taint-value-1,effect=NO_EXECUTE}]'
```

输出：

```
{
    "update": {
        "id": "4a9c8c45-6ac7-3115-be71-d6412a2339b7",
        "status": "InProgress",
        "type": "ConfigUpdate",
        "params": [
            {
                "type": "LabelsToAdd",
                "value": "{\"my-eks-nodegroup-new-label-1\":\"new-value-1\",\"my-eks-nodegroup-new-label-2\":\"new-value-2\"}"
            },
            {
                "type": "LabelsToRemove",
                "value": "[\"my-eks-nodegroup-label-1\",\"my-eks-nodegroup-label-2\"]"
            },
            {
                "type": "TaintsToAdd",
                "value": "[{\"effect\":\"PREFER_NO_SCHEDULE\",\"value\":\"taint-new-value-1\",\"key\":\"taint-new-key-1\"}]"
            },
            {
                "type": "TaintsToRemove",
                "value": "[{\"effect\":\"NO_EXECUTE\",\"value\":\"taint-value-1\",\"key\":\"taint-key-1\"}]"
            }
        ],
        "createdAt": "2024-04-08T12:30:55.486000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新托管节点组。

示例 4：更新托管节点组以更新 Amazon EKS 集群的 EKS Worker 节点的扩展配置和更新配置

以下 update-nodegroup-config 示例更新托管节点组以更新 Amazon EKS 集群的 EKS Worker 节点的扩展配置和更新配置。

```
aws eks update-nodegroup-config \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --scaling-config minSize=1,maxSize=5,desiredSize=2 \
    --update-config maxUnavailable=2
```

输出：

```
{
    "update": {
        "id": "a977160f-59bf-3023-805d-c9826e460aea",
        "status": "InProgress",
        "type": "ConfigUpdate",
        "params": [
            {
                "type": "MinSize",
                "value": "1"
            },
            {
                "type": "MaxSize",
                "value": "5"
            },
            {
                "type": "DesiredSize",
                "value": "2"
            },
            {
                "type": "MaxUnavailable",
                "value": "2"
            }
        ],
        "createdAt": "2024-04-08T12:35:17.036000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新托管节点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateNodegroupConfig。

update-nodegroup-version

以下代码示例演示了如何使用 update-nodegroup-version。

Amazon CLI

示例 1：更新 Amazon EKS 托管节点组的 Kubernetes 版本或 AMI 版本

以下 update-nodegroup-version 示例将 Amazon EKS 托管节点组的 Kubernetes 版本或 AMI 版本更新为 Kubernetes 集群的最新可用版本。

```
aws eks update-nodegroup-version \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --no-force
```

输出：

```
{
    "update": {
        "id": "a94ebfc3-6bf8-307a-89e6-7dbaa36421f7",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.26"
            },
            {
                "type": "ReleaseVersion",
                "value": "1.26.12-20240329"
            }
        ],
        "createdAt": "2024-04-08T13:16:00.724000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的更新托管节点组。

示例 2：更新 Amazon EKS 托管节点组的 Kubernetes 版本或 AMI 版本

以下 update-nodegroup-version 示例将 Amazon EKS 托管节点组的 Kubernetes 版本或 AMI 版本更新为指定的 AMI 发行版本。

```
aws eks update-nodegroup-version \
    --cluster-name my-eks-cluster \
    --nodegroup-name my-eks-nodegroup \
    --kubernetes-version '1.26' \
    --release-version '1.26.12-20240307' \
    --no-force
```

输出：

```
{
    "update": {
        "id": "4db06fe1-088d-336b-bdcd-3fdb94995fb7",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.26"
            },
            {
                "type": "ReleaseVersion",
                "value": "1.26.12-20240307"
            }
        ],
        "createdAt": "2024-04-08T13:13:58.595000-04:00",
        "errors": []
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的“更新托管节点组”– <https://docs.aws.amazon.com/eks/latest/userguide/update-managed-node-group.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateNodegroupVersion。

update-pod-identity-association

以下代码示例演示了如何使用 update-pod-identity-association。

Amazon CLI

更新 EKS 容器组身份关联

以下 update-pod-identity-association 示例通过针对关联 ID a-9njjin9gfghecgocd 将关联的 IAM 角色从 arn:aws:iam::111122223333:role/my-role 更改为 arn:aws:iam::111122223333:role/s3-role，来更新 EKS 容器组身份关联。此 API 仅支持更新关联的 IAM 角色。

```
aws eks update-pod-identity-association \
    --cluster-name eks-customer \
    --association-id a-9njjin9gfghecgocd \
    --role-arn arn:aws:iam::111122223333:role/s3-role
```

输出：

```
{
    "association": {
        "clusterName": "eks-customer",
        "namespace": "default",
        "serviceAccount": "default",
        "roleArn": "arn:aws:iam::111122223333:role/s3-role",
        "associationArn": "arn:aws:eks:us-
west-2:111122223333:podidentityassociation/eks-customer/a-9njjin9gfghecgocd",
        "associationId": "a-9njjin9gfghecgocd",
        "tags": {
            "Key2": "value2",
            "Key1": "value1"
        },
        "createdAt": "2025-05-24T19:52:14.135000-05:00",
        "modifiedAt": "2025-05-25T21:01:53.120000-05:00"
    }
}
```

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 EKS 容器组身份如何向容器组（pod）授予对 Amazon 服务的访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdatePodIdentityAssociation。

使用 Amazon CLI 的 Elastic Beanstalk 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Elastic Beanstalk 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`abort-environment-update`

以下代码示例演示了如何使用 `abort-environment-update`。

Amazon CLI

中止部署

以下命令将中止名为 `my-env` 的环境的正在运行的应用程序版本部署：

```
aws elasticbeanstalk abort-environment-update --environment-name my-env
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortEnvironmentUpdate。

`check-dns-availability`

以下代码示例演示了如何使用 `check-dns-availability`。

Amazon CLI

检查 CNAME 的可用性

以下命令检查子域 `my-cname.elasticbeanstalk.com` 的可用性：

```
aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
```

输出：

```
{
    "Available": true,
    "FullyQualifiedCNAME": "my-cname.elasticbeanstalk.com"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckDnsAvailability。

create-application-version

以下代码示例演示了如何使用 create-application-version。

Amazon CLI

创建新应用程序版本

以下命令创建名为“MyApp”的应用程序的新版本“v1”：

```
aws elasticbeanstalk create-application-version --application-name MyApp --
version-label v1 --description MyAppv1 --source-bundle S3Bucket="amzn-s3-demo-
bucket",S3Key="sample.war" --auto-create-application
```

由于 auto-create-application 选项，如果该应用程序尚不存在，则将自动创建该应用程序。源包是一个 .war 文件，存储在名为“amzn-s3-demo-bucket”的 s3 存储桶中，其中包含 Apache Tomcat 示例应用程序。

输出：

```
{
  "ApplicationVersion": {
      "ApplicationName": "MyApp",
      "VersionLabel": "v1",
      "Description": "MyAppv1",
      "DateCreated": "2015-02-03T23:01:25.412Z",
      "DateUpdated": "2015-02-03T23:01:25.412Z",
      "SourceBundle": {
          "S3Bucket": "amzn-s3-demo-bucket",
          "S3Key": "sample.war"
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApplicationVersion。

create-application

以下代码示例演示了如何使用 create-application。

Amazon CLI

创建新应用程序

以下命令创建名为“MyApp”的新应用程序：

```
aws elasticbeanstalk create-application --application-name MyApp --description "my
 application"
```

create-application 命令仅配置该应用程序的名称和描述。要上传应用程序的源代码，请使用 create-application-version 创建应用程序的初始版本。create-application-version 还有一个 auto-create-application 选项，可让您通过一个步骤即可创建应用程序和应用程序版本。

输出：

```
{
    "Application": {
        "ApplicationName": "MyApp",
        "ConfigurationTemplates": [],
        "DateUpdated": "2015-02-12T18:32:21.181Z",
        "Description": "my application",
        "DateCreated": "2015-02-12T18:32:21.181Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApplication。

create-configuration-template

以下代码示例演示了如何使用 create-configuration-template。

Amazon CLI

创建配置模板

以下命令根据应用于 ID 为 e-rpqsewtp2j 的环境的设置创建名为 my-app-v1 的配置模板：

```
aws elasticbeanstalk create-configuration-template --application-name my-app --
template-name my-app-v1 --environment-id e-rpqsewtp2j
```

输出：

```
{
    "ApplicationName": "my-app",
    "TemplateName": "my-app-v1",
    "DateCreated": "2015-08-12T18:40:39Z",
    "DateUpdated": "2015-08-12T18:40:39Z",
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConfigurationTemplate。

create-environment

以下代码示例演示了如何使用 create-environment。

Amazon CLI

为应用程序创建新环境

以下命令为名为“my-app”的 Java 应用程序的版本“v1”创建一个新环境：

```
aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --cname-prefix my-app --version-label v1 --solution-stack-name "64bit
 Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
```

输出：

```
{
  "ApplicationName": "my-app",
  "EnvironmentName": "my-env",
```

```
  "VersionLabel": "v1",
  "Status": "Launching",
  "EnvironmentId": "e-izqpassy4h",
  "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8",
  "CNAME": "my-app.elasticbeanstalk.com",
  "Health": "Grey",
  "Tier": {
      "Type": "Standard",
      "Name": "WebServer",
      "Version": " "
  },
  "DateUpdated": "2015-02-03T23:04:54.479Z",
  "DateCreated": "2015-02-03T23:04:54.479Z"
}
```

v1 是之前使用 create-application-version 上传的应用程序版本的标签。

指定 JSON 文件以定义环境配置选项

以下 `create-environment` 命令指定应使用名为 `myoptions.json` 的 JSON 文件来覆盖从解决方案堆栈或配置模板获取的值：

```
aws elasticbeanstalk create-environment --environment-name sample-env --application-name sampleapp --option-settings file://myoptions.json
```

`myoptions.json` 是一个 JSON 对象，可定义多项设置：

```
[
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "Interval",
    "Value": "15"
  },
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "Timeout",
    "Value": "8"
  },
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "HealthyThreshold",
    "Value": "2"
  },
```

```
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "UnhealthyThreshold",
    "Value": "3"
  }
]
```

有关更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEnvironment。

`create-storage-location`

以下代码示例演示了如何使用 `create-storage-location`。

Amazon CLI

创建存储位置

以下命令在 Amazon S3 中创建一个存储位置：

```
aws elasticbeanstalk create-storage-location
```

输出：

```
{
    "S3Bucket": "elasticbeanstalk-us-west-2-0123456789012"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStorageLocation。

`delete-application-version`

以下代码示例演示了如何使用 `delete-application-version`。

Amazon CLI

删除应用程序版本

以下命令将删除名为 `my-app` 的应用程序的名为 `22a0-stage-150819_182129` 的应用程序版本：

```
aws elasticbeanstalk delete-application-version --version-label 22a0-
stage-150819_182129 --application-name my-app
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApplicationVersion。

delete-application

以下代码示例演示了如何使用 delete-application。

Amazon CLI

删除 应用程序

以下命令将删除名为 my-app 的应用程序：

```
aws elasticbeanstalk delete-application --application-name my-app
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApplication。

delete-configuration-template

以下代码示例演示了如何使用 delete-configuration-template。

Amazon CLI

删除配置模板

以下命令将删除名为 my-app 的应用程序的名为 my-template 的配置模板：

```
aws elasticbeanstalk delete-configuration-template --template-name my-template --
application-name my-app
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConfigurationTemplate。

delete-environment-configuration

以下代码示例演示了如何使用 delete-environment-configuration。

Amazon CLI

删除草稿配置

以下命令将删除名为 my-env 的环境的草稿配置：

```
aws elasticbeanstalk delete-environment-configuration --environment-name my-env --
application-name my-app
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEnvironmentConfiguration。

describe-application-versions

以下代码示例演示了如何使用 describe-application-versions。

Amazon CLI

查看有关应用程序版本的信息

以下命令检索有关标记为 v2 的应用程序版本的信息：

```
aws elasticbeanstalk describe-application-versions --application-name my-app --
version-label "v2"
```

输出：

```
{
    "ApplicationVersions": [
        {
            "ApplicationName": "my-app",
            "VersionLabel": "v2",
            "Description": "update cover page",
            "DateCreated": "2015-07-23T01:32:26.079Z",
            "DateUpdated": "2015-07-23T01:32:26.079Z",
            "SourceBundle": {
                "S3Bucket": "elasticbeanstalk-us-west-2-015321684451",
                "S3Key": "my-app/5026-stage-150723_224258.war"
            }
        },
      {
          "ApplicationName": "my-app",
          "VersionLabel": "v1",
          "Description": "initial version",
          "DateCreated": "2015-07-23T22:26:10.816Z",
          "DateUpdated": "2015-07-23T22:26:10.816Z",
          "SourceBundle": {
```

```
            "S3Bucket": "elasticbeanstalk-us-west-2-015321684451",
            "S3Key": "my-app/5026-stage-150723_222618.war"
        }
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeApplicationVersions。

describe-applications

以下代码示例演示了如何使用 describe-applications。

Amazon CLI

查看应用程序列表

以下命令检索有关当前区域中应用程序的信息：

```
aws elasticbeanstalk describe-applications
```

输出：

```
{
    "Applications": [
        {
            "ApplicationName": "ruby",
            "ConfigurationTemplates": [],
            "DateUpdated": "2015-08-13T21:05:44.376Z",
            "Versions": [
                "Sample Application"
            ],
            "DateCreated": "2015-08-13T21:05:44.376Z"
        },
        {
            "ApplicationName": "pythonsample",
            "Description": "Application created from the EB CLI using \"eb init\"",
            "Versions": [
                "Sample Application"
            ],
            "DateCreated": "2015-08-13T19:05:43.637Z",
            "ConfigurationTemplates": [],
```

```
            "DateUpdated": "2015-08-13T19:05:43.637Z"
        },
        {
            "ApplicationName": "nodejs-example",
            "ConfigurationTemplates": [],
            "DateUpdated": "2015-08-06T17:50:02.486Z",
            "Versions": [
                "add elasticache",
                "First Release"
            ],
            "DateCreated": "2015-08-06T17:50:02.486Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeApplications。

describe-configuration-options

以下代码示例演示了如何使用 describe-configuration-options。

Amazon CLI

查看环境的配置选项

以下命令检索名为 my-env 的环境的所有可用配置选项的描述：

```
aws elasticbeanstalk describe-configuration-options --environment-name my-env --
application-name my-app
```

输出（缩减）：

```
{
    "Options": [
        {
            "Name": "JVMOptions",
            "UserDefined": false,
            "DefaultValue": "Xms=256m,Xmx=256m,XX:MaxPermSize=64m,JVM Options=",
            "ChangeSeverity": "RestartApplicationServer",
            "Namespace": "aws:cloudformation:template:parameter",
            "ValueType": "KeyValueList"
        },
```

```
        {
            "Name": "Interval",
            "UserDefined": false,
            "DefaultValue": "30",
            "ChangeSeverity": "NoInterruption",
            "Namespace": "aws:elb:healthcheck",
            "MaxValue": 300,
            "MinValue": 5,
            "ValueType": "Scalar"
        },
        ...
        {
            "Name": "LowerThreshold",
            "UserDefined": false,
            "DefaultValue": "2000000",
            "ChangeSeverity": "NoInterruption",
            "Namespace": "aws:autoscaling:trigger",
            "MinValue": 0,
            "ValueType": "Scalar"
        },
        {
            "Name": "ListenerEnabled",
            "UserDefined": false,
            "DefaultValue": "true",
            "ChangeSeverity": "Unknown",
            "Namespace": "aws:elb:listener",
            "ValueType": "Boolean"
        }
    ]
}
```

可用配置选项因平台和配置版本而异。有关命名空间和支持的选项的更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigurationOptions。

`describe-configuration-settings`

以下代码示例演示了如何使用 `describe-configuration-settings`。

Amazon CLI

查看环境的配置设置

以下命令检索名为 my-env 的环境的配置设置：

```
aws elasticbeanstalk describe-configuration-settings --environment-name my-env --
application-name my-app
```

输出（缩减）：

```
{
    "ConfigurationSettings": [
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "Description": "Environment created from the EB CLI using \"eb create
\"",
            "DeploymentStatus": "deployed",
            "DateCreated": "2015-08-13T19:16:25Z",
            "OptionSettings": [
                {
                    "OptionName": "Availability Zones",
                    "ResourceName": "AWSEBAutoScalingGroup",
                    "Namespace": "aws:autoscaling:asg",
                    "Value": "Any"
                },
                {
                    "OptionName": "Cooldown",
                    "ResourceName": "AWSEBAutoScalingGroup",
                    "Namespace": "aws:autoscaling:asg",
                    "Value": "360"
                },
                ...
                {
                    "OptionName": "ConnectionDrainingTimeout",
                    "ResourceName": "AWSEBLoadBalancer",
                    "Namespace": "aws:elb:policies",
                    "Value": "20"
                },
                {
                    "OptionName": "ConnectionSettingIdleTimeout",
                    "ResourceName": "AWSEBLoadBalancer",
                    "Namespace": "aws:elb:policies",
                    "Value": "60"
                }
            ],
```

```
            "DateUpdated": "2015-08-13T23:30:07Z",
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8
 Java 8"
        }
    ]
}
```

有关命名空间和支持的选项的更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeConfigurationSettings。

`describe-environment-health`

以下代码示例演示了如何使用 `describe-environment-health`。

Amazon CLI

查看环境运行状况

以下命令检索名为 `my-env` 的环境的整体运行状况信息：

```
aws elasticbeanstalk describe-environment-health --environment-name my-env --
attribute-names All
```

输出：

```
{
    "Status": "Ready",
    "EnvironmentName": "my-env",
    "Color": "Green",
    "ApplicationMetrics": {
        "Duration": 10,
        "Latency": {
            "P99": 0.004,
            "P75": 0.002,
            "P90": 0.003,
            "P95": 0.004,
            "P85": 0.003,
            "P10": 0.001,
            "P999": 0.004,
            "P50": 0.001
        },
```

```
        "RequestCount": 45,
        "StatusCodes": {
            "Status3xx": 0,
            "Status2xx": 45,
            "Status5xx": 0,
            "Status4xx": 0
        }
    },
    "RefreshedAt": "2015-08-20T21:09:18Z",
    "HealthStatus": "Ok",
    "InstancesHealth": {
        "Info": 0,
        "Ok": 1,
        "Unknown": 0,
        "Severe": 0,
        "Warning": 0,
        "Degraded": 0,
        "NoData": 0,
        "Pending": 0
    },
    "Causes": []
}
```

运行状况信息仅适用于启用了增强型运行状况报告的环境。有关更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“增强型运行状况报告和监控”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironmentHealth。

`describe-environment-resources`

以下代码示例演示了如何使用 `describe-environment-resources`。

Amazon CLI

查看有关环境中 Amazon 资源的信息

以下命令检索有关名为 `my-env` 的环境中资源的信息：

```
aws elasticbeanstalk describe-environment-resources --environment-name my-env
```

输出：

```
{
```

```
    "EnvironmentResources": {
        "EnvironmentName": "my-env",
        "AutoScalingGroups": [
            {
                "Name": "awseb-e-qu3fyyjyjs-stack-AWSEBAutoScalingGroup-
QSB2ZO88SXZT"
            }
        ],
        "Triggers": [],
        "LoadBalancers": [
            {
                "Name": "awseb-e-q-AWSEBLoa-1EEPZ0K98BIF0"
            }
        ],
        "Queues": [],
        "Instances": [
            {
                "Id": "i-0c91c786"
            }
        ],
        "LaunchConfigurations": [
            {
                "Name": "awseb-e-qu3fyyjyjs-stack-
AWSEBAutoScalingLaunchConfiguration-1UUVQIBC96TQ2"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironmentResources。

describe-environments

以下代码示例演示了如何使用 describe-environments。

Amazon CLI

查看有关环境的信息

以下命令检索有关名为 my-env 的环境的信息：

```
aws elasticbeanstalk describe-environments --environment-names my-env
```

输出：

```
{
    "Environments": [
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "VersionLabel": "7f58-stage-150812_025409",
            "Status": "Ready",
            "EnvironmentId": "e-rpqsewtp2j",
            "EndpointURL": "awseb-e-w-AWSEBLoa-1483140XB0Q4L-109QXY8121.us-
west-2.elb.amazonaws.com",
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8
 Java 8",
            "CNAME": "my-env.elasticbeanstalk.com",
            "Health": "Green",
            "AbortableOperationInProgress": false,
            "Tier": {
                "Version": " ",
                "Type": "Standard",
                "Name": "WebServer"
            },
            "DateUpdated": "2015-08-12T18:16:55.019Z",
            "DateCreated": "2015-08-07T20:48:49.599Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEnvironments。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

查看环境的事件

以下命令检索名为 my-env 的环境的事件：

```
aws elasticbeanstalk describe-events --environment-name my-env
```

输出（缩减）：

```
{
    "Events": [
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "Message": "Environment health has transitioned from Info to Ok.",
            "EventDate": "2015-08-20T07:06:53.535Z",
            "Severity": "INFO"
        },
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "Severity": "INFO",
            "RequestId": "b7f3960b-4709-11e5-ba1e-07e16200da41",
            "Message": "Environment update completed successfully.",
            "EventDate": "2015-08-20T07:06:02.049Z"
        },
        ...
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "Severity": "INFO",
            "RequestId": "ca8dfbf6-41ef-11e5-988b-651aa638f46b",
            "Message": "Using elasticbeanstalk-us-west-2-012445113685 as Amazon S3
 storage bucket for environment data.",
            "EventDate": "2015-08-13T19:16:27.561Z"
        },
        {
            "ApplicationName": "my-app",
            "EnvironmentName": "my-env",
            "Severity": "INFO",
            "RequestId": "cdfba8f6-41ef-11e5-988b-65638f41aa6b",
            "Message": "createEnvironment is starting.",
            "EventDate": "2015-08-13T19:16:26.581Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-instances-health

以下代码示例演示了如何使用 describe-instances-health。

Amazon CLI

查看环境运行状况

以下命令检索名为 my-env 的环境中实例的运行状况信息：

```
aws elasticbeanstalk describe-instances-health --environment-name my-env --
attribute-names All
```

输出：

```
{
    "InstanceHealthList": [
        {
            "InstanceId": "i-08691cc7",
            "ApplicationMetrics": {
                "Duration": 10,
                "Latency": {
                    "P99": 0.006,
                    "P75": 0.002,
                    "P90": 0.004,
                    "P95": 0.005,
                    "P85": 0.003,
                    "P10": 0.0,
                    "P999": 0.006,
                    "P50": 0.001
                },
                "RequestCount": 48,
                "StatusCodes": {
                    "Status3xx": 0,
                    "Status2xx": 47,
                    "Status5xx": 0,
                    "Status4xx": 1
                }
            },
            "System": {
                "LoadAverage": [
                    0.0,
                    0.02,
```

```
                    0.05
                ],
                "CPUUtilization": {
                    "SoftIRQ": 0.1,
                    "IOWait": 0.2,
                    "System": 0.3,
                    "Idle": 97.8,
                    "User": 1.5,
                    "IRQ": 0.0,
                    "Nice": 0.1
                }
            },
            "Color": "Green",
            "HealthStatus": "Ok",
            "LaunchedAt": "2015-08-13T19:17:09Z",
            "Causes": []
        }
    ],
    "RefreshedAt": "2015-08-20T21:09:08Z"
}
```

运行状况信息仅适用于启用了增强型运行状况报告的环境。有关更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“增强型运行状况报告和监控”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstancesHealth。

list-available-solution-stacks

以下代码示例演示了如何使用 list-available-solution-stacks。

Amazon CLI

查看解决方案堆栈

以下命令列出了所有当前可用平台配置以及您过去使用过的所有和任何平台配置的解决方案堆栈：

```
aws elasticbeanstalk list-available-solution-stacks
```

输出（缩减）：

```
{
    "SolutionStacks": [
```

```
        "64bit Amazon Linux 2015.03 v2.0.0 running Node.js",
        "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.6",
        "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.5",
        "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.4",
        "64bit Amazon Linux 2015.03 v2.0.0 running Python 3.4",
        "64bit Amazon Linux 2015.03 v2.0.0 running Python 2.7",
        "64bit Amazon Linux 2015.03 v2.0.0 running Python",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Puma)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Passenger Standalone)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.1 (Puma)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.1 (Passenger Standalone)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.0 (Puma)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.0 (Passenger Standalone)",
        "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 1.9.3",
        "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8",
        "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 7 Java 7",
        "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 7 Java 6",
        "64bit Windows Server Core 2012 R2 running IIS 8.5",
        "64bit Windows Server 2012 R2 running IIS 8.5",
        "64bit Windows Server 2012 running IIS 8",
        "64bit Windows Server 2008 R2 running IIS 7.5",
        "64bit Amazon Linux 2015.03 v2.0.0 running Docker 1.6.2",
        "64bit Amazon Linux 2015.03 v2.0.0 running Multi-container Docker 1.6.2
 (Generic)",
        "64bit Debian jessie v2.0.0 running GlassFish 4.1 Java 8 (Preconfigured -
 Docker)",
        "64bit Debian jessie v2.0.0 running GlassFish 4.0 Java 7 (Preconfigured -
 Docker)",
        "64bit Debian jessie v2.0.0 running Go 1.4 (Preconfigured - Docker)",
        "64bit Debian jessie v2.0.0 running Go 1.3 (Preconfigured - Docker)",
        "64bit Debian jessie v2.0.0 running Python 3.4 (Preconfigured - Docker)",
    ],
    "SolutionStackDetails": [
        {
            "PermittedFileTypes": [
                "zip"
            ],
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Node.js"
        },
        ...
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAvailableSolutionStacks。

rebuild-environment

以下代码示例演示了如何使用 rebuild-environment。

Amazon CLI

重建环境

以下命令在名为 my-env 的环境中终止并重新创建资源：

```
aws elasticbeanstalk rebuild-environment --environment-name my-env
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebuildEnvironment。

request-environment-info

以下代码示例演示了如何使用 request-environment-info。

Amazon CLI

请求结尾日志

以下命令从名为 my-env 的环境中请求日志：

```
aws elasticbeanstalk request-environment-info --environment-name my-env --info-
type tail
```

请求日志后，使用 retrieve-environment-info 检索其位置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestEnvironmentInfo。

restart-app-server

以下代码示例演示了如何使用 restart-app-server。

Amazon CLI

重启应用程序服务器

以下命令将重启名为 my-env 的环境中所有实例上的应用程序服务器：

```
aws elasticbeanstalk restart-app-server --environment-name my-env
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestartAppServer。

retrieve-environment-info

以下代码示例演示了如何使用 retrieve-environment-info。

Amazon CLI

检索结尾日志

以下命令从名为 my-env 的环境中检索指向日志的链接：

```
aws elasticbeanstalk retrieve-environment-info --environment-name my-env --info-
type tail
```

输出：

```
{
    "EnvironmentInfo": [
        {
            "SampleTimestamp": "2015-08-20T22:23:17.703Z",
            "Message": "https://elasticbeanstalk-us-
west-2-0123456789012.s3.amazonaws.com/resources/environments/
logs/tail/e-fyqyju3yjs/i-09c1c867/TailLogs-1440109397703.out?
AWSAccessKeyId=AKGPT4J56IAJ2EUBL5CQ&Expires=1440195891&Signature=n
%2BEalOV6A2HIOx4Rcfb7LT16bBM%3D",
            "InfoType": "tail",
            "Ec2InstanceId": "i-09c1c867"
        }
    ]
}
```

在浏览器中查看链接。在检索之前，必须使用 request-environment-info 请求日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetrieveEnvironmentInfo。

swap-environment-cnames

以下代码示例演示了如何使用 swap-environment-cnames。

Amazon CLI

交换环境 CNAME

以下命令交换两个环境的已分配子域：

```
aws elasticbeanstalk swap-environment-cnames --source-environment-name my-env-blue
 --destination-environment-name my-env-green
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SwapEnvironmentCnames。

terminate-environment

以下代码示例演示了如何使用 terminate-environment。

Amazon CLI

终止环境

以下命令将终止名为 my-env 的 Elastic Beanstalk 环境：

```
aws elasticbeanstalk terminate-environment --environment-name my-env
```

输出：

```
{
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Status": "Terminating",
    "EnvironmentId": "e-fh2eravpns",
    "EndpointURL": "awseb-e-f-AWSEBLoa-1I9XUMP4-8492WNUP202574.us-
west-2.elb.amazonaws.com",
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java
 8",
    "CNAME": "my-env.elasticbeanstalk.com",
    "Health": "Grey",
    "AbortableOperationInProgress": false,
    "Tier": {
        "Version": " ",
        "Type": "Standard",
        "Name": "WebServer"
```

```
    },
    "DateUpdated": "2015-08-12T19:05:54.744Z",
    "DateCreated": "2015-08-12T18:52:53.622Z"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateEnvironment。

update-application-version

以下代码示例演示了如何使用 update-application-version。

Amazon CLI

更改应用程序版本的描述

以下命令更新名为 22a0-stage-150819_185942 的应用程序版本的描述：

```
aws elasticbeanstalk update-application-version --version-label 22a0-
stage-150819_185942 --application-name my-app --description "new description"
```

输出：

```
{
    "ApplicationVersion": {
        "ApplicationName": "my-app",
        "VersionLabel": "22a0-stage-150819_185942",
        "Description": "new description",
        "DateCreated": "2015-08-19T18:59:17.646Z",
        "DateUpdated": "2015-08-20T22:53:28.871Z",
        "SourceBundle": {
            "S3Bucket": "elasticbeanstalk-us-west-2-0123456789012",
            "S3Key": "my-app/22a0-stage-150819_185942.war"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApplicationVersion。

update-application

以下代码示例演示了如何使用 update-application。

Amazon CLI

更改应用程序的描述

以下命令更新名为 my-app 的应用程序的描述：

```
aws elasticbeanstalk update-application --application-name my-app --description "my
 Elastic Beanstalk application"
```

输出：

```
{
    "Application": {
        "ApplicationName": "my-app",
        "Description": "my Elastic Beanstalk application",
        "Versions": [
            "2fba-stage-150819_234450",
            "bf07-stage-150820_214945",
            "93f8",
            "fd7c-stage-150820_000431",
            "22a0-stage-150819_185942"
        ],
        "DateCreated": "2015-08-13T19:15:50.449Z",
        "ConfigurationTemplates": [],
        "DateUpdated": "2015-08-20T22:34:56.195Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateApplication。

update-configuration-template

以下代码示例演示了如何使用 update-configuration-template。

Amazon CLI

更新配置模板

以下命令将从名为 my-template 的已保存配置模板中删除已配置的 CloudWatch 自定义运行状况指标配置 ConfigDocument：

```
aws elasticbeanstalk update-configuration-template --template-
name my-template --application-name my-app --options-to-
remove Namespace=aws:elasticbeanstalk:healthreporting:system,OptionName=ConfigDocument
```

输出：

```
{
    "ApplicationName": "my-app",
    "TemplateName": "my-template",
    "DateCreated": "2015-08-20T22:39:31Z",
    "DateUpdated": "2015-08-20T22:43:11Z",
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
}
```

有关命名空间和支持的选项的更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConfigurationTemplate。

update-environment

以下代码示例演示了如何使用 update-environment。

Amazon CLI

将环境更新为新版本

以下命令将名为“my-env”的环境更新为其所属应用程序的版本“v2”：

```
aws elasticbeanstalk update-environment --environment-name my-env --version-label v2
```

此命令要求“my-env”环境已存在，并且属于具有标签为“v2”的有效应用程序版本的应用程序。

输出：

```
{
  "ApplicationName": "my-app",
  "EnvironmentName": "my-env",
  "VersionLabel": "v2",
  "Status": "Updating",
  "EnvironmentId": "e-szqipays4h",
```

```
  "EndpointURL": "awseb-e-i-AWSEBLoa-1RDLX6TC9VUAO-0123456789.us-
west-2.elb.amazonaws.com",
  "SolutionStackName": "64bit Amazon Linux running Tomcat 7",
  "CNAME": "my-env.elasticbeanstalk.com",
  "Health": "Grey",
  "Tier": {
      "Version": " ",
      "Type": "Standard",
      "Name": "WebServer"
  },
  "DateUpdated": "2015-02-03T23:12:29.119Z",
  "DateCreated": "2015-02-03T23:04:54.453Z"
}
```

设置环境变量

以下命令将“my-env”环境中的“PARAM1”变量的值设置为“ParamValue”：

```
aws elasticbeanstalk update-environment --environment-name my-env --option-
settings Namespace=aws:elasticbeanstalk:application:environment,OptionName=PARAM1,Value=Para
```

option-settings 参数除了该变量的名称和值之外，还采用命名空间。除了环境变量之外，Elastic Beanstalk 还支持多个选项命名空间。

从文件配置选项设置

以下命令从文件配置 aws:elb:loadbalancer 命名空间中的多个选项：

```
aws elasticbeanstalk update-environment --environment-name my-env --option-
settings file://options.json
```

options.json 是一个 JSON 对象，可定义多项设置：

```
[
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "Interval",
    "Value": "15"
  },
  {
    "Namespace": "aws:elb:healthcheck",
```

```
    "OptionName": "Timeout",
    "Value": "8"
  },
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "HealthyThreshold",
    "Value": "2"
  },
  {
    "Namespace": "aws:elb:healthcheck",
    "OptionName": "UnhealthyThreshold",
    "Value": "3"
  }
]
```

输出：

```
{
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "VersionLabel": "7f58-stage-150812_025409",
    "Status": "Updating",
    "EnvironmentId": "e-wtp2rpqsej",
    "EndpointURL": "awseb-e-w-AWSEBLoa-14XB83101Q4L-104QXY80921.sa-
east-1.elb.amazonaws.com",
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java
 8",
    "CNAME": "my-env.elasticbeanstalk.com",
    "Health": "Grey",
    "AbortableOperationInProgress": true,
    "Tier": {
        "Version": " ",
        "Type": "Standard",
        "Name": "WebServer"
    },
    "DateUpdated": "2015-08-12T18:15:23.804Z",
    "DateCreated": "2015-08-07T20:48:49.599Z"
}
```

有关命名空间和支持的选项的更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEnvironment。

validate-configuration-settings

以下代码示例演示了如何使用 validate-configuration-settings。

Amazon CLI

验证配置设置

以下命令可验证 CloudWatch 自定义指标配置文档：

```
aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings file://options.json
```

options.json 是一个 JSON 文档，其中包含一个或多个要验证的配置设置：

```
[
    {
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "OptionName": "ConfigDocument",
        "Value": "{\"CloudWatchMetrics\": {\"Environment\":
 {\"ApplicationLatencyP99.9\": null,\"InstancesSevere\": 60,
\"ApplicationLatencyP90\": 60,\"ApplicationLatencyP99\": null,
\"ApplicationLatencyP95\": 60,\"InstancesUnknown\": 60,\"ApplicationLatencyP85\":
 60,\"InstancesInfo\": null,\"ApplicationRequests2xx\": null,\"InstancesDegraded
\": null,\"InstancesWarning\": 60,\"ApplicationLatencyP50\": 60,
\"ApplicationRequestsTotal\": null,\"InstancesNoData\": null,\"InstancesPending
\": 60,\"ApplicationLatencyP10\": null,\"ApplicationRequests5xx\": null,
\"ApplicationLatencyP75\": null,\"InstancesOk\": 60,\"ApplicationRequests3xx\":
 null,\"ApplicationRequests4xx\": null},\"Instance\": {\"ApplicationLatencyP99.9\":
 null,\"ApplicationLatencyP90\": 60,\"ApplicationLatencyP99\": null,
\"ApplicationLatencyP95\": null,\"ApplicationLatencyP85\": null,\"CPUUser\": 60,
\"ApplicationRequests2xx\": null,\"CPUIdle\": null,\"ApplicationLatencyP50\":
 null,\"ApplicationRequestsTotal\": 60,\"RootFilesystemUtil\": null,
\"LoadAverage1min\": null,\"CPUIrq\": null,\"CPUNice\": 60,\"CPUIowait\": 60,
\"ApplicationLatencyP10\": null,\"LoadAverage5min\": null,\"ApplicationRequests5xx
\": null,\"ApplicationLatencyP75\": 60,\"CPUSystem\": 60,\"ApplicationRequests3xx\":
 60,\"ApplicationRequests4xx\": null,\"InstanceHealth\": null,\"CPUSoftirq\": 60}},
\"Version\": 1}"
    }
]
```

如果您指定的选项对指定环境有效，则 Elastic Beanstalk 会返回一个空的消息数组：

```
{
    "Messages": []
}
```

如果验证失败，则响应将包含有关错误的信息：

```
{
    "Messages": [
        {
            "OptionName": "ConfigDocumet",
            "Message": "Invalid option specification (Namespace:
 'aws:elasticbeanstalk:healthreporting:system', OptionName: 'ConfigDocumet'):
 Unknown configuration setting.",
            "Namespace": "aws:elasticbeanstalk:healthreporting:system",
            "Severity": "error"
        }
    ]
}
```

有关命名空间和支持的选项的更多信息，请参阅《Amazon Elastic Beanstalk 开发人员指南》中的“选项值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateConfigurationSettings。

ELB – 使用 Amazon CLI 的版本 1 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 ELB – 版本 1 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-tags

以下代码示例演示了如何使用 add-tags。

Amazon CLI

将标签添加到负载均衡器

此示例将标签添加到指定负载均衡器。

命令:

```
aws elb add-tags --load-balancer-name my-load-balancer --
tags "Key=project,Value=lima" "Key=department,Value=digital-media"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTags。

apply-security-groups-to-load-balancer

以下代码示例演示了如何使用 apply-security-groups-to-load-balancer。

Amazon CLI

将安全组与 VPC 中的负载均衡器相关联

此示例将安全组与 VPC 中的指定负载均衡器相关联。

命令:

```
aws elb apply-security-groups-to-load-balancer --load-balancer-name my-load-balancer
 --security-groups sg-fc448899
```

输出：

```
{
  "SecurityGroups": [
      "sg-fc448899"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ApplySecurityGroupsToLoadBalancer。

attach-load-balancer-to-subnets

以下代码示例演示了如何使用 attach-load-balancer-to-subnets。

Amazon CLI

将子网连接到负载均衡器

此示例将指定的子网添加到指定负载均衡器的已配置子网集中。

命令:

```
aws elb attach-load-balancer-to-subnets --load-balancer-name my-load-balancer --
subnets subnet-0ecac448
```

输出：

```
{
   "Subnets": [
       "subnet-15aaab61",
       "subnet-0ecac448"
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachLoadBalancerToSubnets。

configure-health-check

以下代码示例演示了如何使用 configure-health-check。

Amazon CLI

为后端 EC2 实例指定运行状况检查设置

此示例指定用于评估后端 EC2 实例的运行状况的运行状况检查设置。

命令:

```
aws elb configure-health-check --load-balancer-name my-load-balancer --health-
check Target=HTTP:80/
png,Interval=30,UnhealthyThreshold=2,HealthyThreshold=2,Timeout=3
```

输出：

```
{
   "HealthCheck": {
       "HealthyThreshold": 2,
       "Interval": 30,
       "Target": "HTTP:80/png",
       "Timeout": 3,
       "UnhealthyThreshold": 2
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfigureHealthCheck。

create-app-cookie-stickiness-policy

以下代码示例演示了如何使用 create-app-cookie-stickiness-policy。

Amazon CLI

为 HTTPS 负载均衡器生成粘性策略

此示例生成一个粘性策略，该策略遵循应用程序生成的 Cookie 的粘性会话生命周期。

命令:

```
aws elb create-app-cookie-stickiness-policy --load-balancer-name my-load-balancer --
policy-name my-app-cookie-policy --cookie-name my-app-cookie
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAppCookieStickinessPolicy。

create-lb-cookie-stickiness-policy

以下代码示例演示了如何使用 create-lb-cookie-stickiness-policy。

Amazon CLI

为 HTTPS 负载均衡器生成基于持续时间的粘性策略

此示例生成一个粘性策略，其粘性会话生命周期由指定的有效期限控制。

命令:

```
aws elb create-lb-cookie-stickiness-policy --load-balancer-name my-load-balancer --
policy-name my-duration-cookie-policy --cookie-expiration-period 60
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLbCookieStickinessPolicy。

create-load-balancer-listeners

以下代码示例演示了如何使用 create-load-balancer-listeners。

Amazon CLI

为负载均衡器创建 HTTP 侦听器

此示例使用 HTTP 协议在端口 80 上为您的负载均衡器创建侦听器。

命令:

```
aws elb create-load-balancer-listeners --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"
```

为负载均衡器创建 HTTPS 侦听器

此示例使用 HTTPS 协议在端口 443 上为您的负载均衡器创建侦听器。

命令:

```
aws elb create-load-balancer-listeners --load-balancer-name my-load-balancer --
listeners "Protocol=HTTPS,LoadBalancerPort=443,InstanceProtocol=HTTP,InstancePort=80"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancerListeners。

create-load-balancer-policy

以下代码示例演示了如何使用 create-load-balancer-policy。

Amazon CLI

创建在负载均衡器上启用代理协议的策略

此示例创建一个策略，该策略在指定的负载均衡器上启用代理协议。

命令:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-ProxyProtocol-policy --policy-type-name ProxyProtocolPolicyType --policy-attributes AttributeName=ProxyProtocol,AttributeValue=true
```

使用推荐的安全策略创建 SSL 协商策略

此示例使用推荐的安全策略为指定的 HTTPS 负载均衡器创建 SSL 协商策略。

命令:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-SSLNegotiation-policy --policy-type-name SSLNegotiationPolicyType --policy-attributes AttributeName=Reference-Security-Policy,AttributeValue=ELBSecurityPolicy-2015-03
```

使用自定义安全策略创建 SSL 协商策略

此示例通过启用协议和密码使用自定义安全策略为您的 HTTPS 负载均衡器创建 SSL 协商策略。

命令:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-SSLNegotiation-policy --policy-type-name SSLNegotiationPolicyType --policy-attributes AttributeName=Protocol-SSLv3,AttributeValue=true AttributeName=Protocol-TLSv1.1,AttributeValue=true AttributeName=DHE-RSA-AES256-SHA256,AttributeValue=true AttributeName=Server-Defined-Cipher-Order,AttributeValue=true
```

创建公有密钥策略

此示例创建一个公有密钥策略。

命令:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-PublicKey-policy --policy-type-name PublicKeyPolicyType --policy-attributes AttributeName=PublicKey,AttributeValue=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQ
+dS74kj//c6x7ROtusUaeQCTgIUkayttRDWchuqo1pHC1u
+n5xxXnBBe2ejbb2WRsKIQ5rXEeixsjFpFsojpSQKkzhVGI6mJVZBJDVKSHmswnwLBdofLhzvllpovBPTHe
```

```
+o4haAWvDBALJU0pkSI1FecPHcs2hwxf14zHoXy1e2k36A64nXW43wtfx5qcVSIxtCEOjnYRg7RPvybaGfQ
+v6Iaxb/+7J5kEvZhTFQId+bSiJImF1FSUT1W1xwzBZPUbcUkkXDj45vC2s3Z8E
+Lk7a3uZhvsQHLZnrfuWjBWGWvZ/MhZYgEXAMPLE
```

创建后端服务器身份验证策略

此示例创建一个后端服务器身份验证策略，该策略使用公有密钥策略在您的后端实例上启用身份验证。

命令:

```
aws elb create-load-balancer-policy --load-balancer-name my-
load-balancer --policy-name my-authentication-policy --policy-
type-name BackendServerAuthenticationPolicyType --policy-
attributes AttributeName=PublicKeyPolicyName,AttributeValue=my-PublicKey-policy
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancerPolicy。

create-load-balancer

以下代码示例演示了如何使用 create-load-balancer。

Amazon CLI

创建 HTTP 负载均衡器

此示例在 VPC 中创建带有 HTTP 侦听器的负载均衡器。

命令:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"
 --subnets subnet-15aaab61 --security-groups sg-a61988c3
```

输出：

```
{
    "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com"
}
```

此示例在 EC2-Classic 中创建带有 HTTP 侦听器的负载均衡器。

命令:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"
 --availability-zones us-west-2a us-west-2b
```

输出：

```
{
    "DNSName": "my-load-balancer-123456789.us-west-2.elb.amazonaws.com"
}
```

创建 HTTPS 负载均衡器

此示例在 VPC 中创建带有 HTTPS 侦听器的负载均衡器。

命令:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" "Protoco
certificate/my-server-cert" --subnets subnet-15aaab61 --security-groups sg-a61988c3
```

输出：

```
{
    "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com"
}
```

此示例在 EC2-Classic 中创建带有 HTTPS 侦听器的负载均衡器。

命令:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" "Protoco
certificate/my-server-cert" --availability-zones us-west-2a us-west-2b
```

输出：

```
{
    "DNSName": "my-load-balancer-123456789.us-west-2.elb.amazonaws.com"
}
```

创建内部负载均衡器

此示例在 VPC 中创建带有 HTTP 侦听器的内部负载均衡器。

命令:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"
 --scheme internal --subnets subnet-a85db0df --security-groups sg-a61988c3
```

输出：

```
{
    "DNSName": "internal-my-load-balancer-123456789.us-west-2.elb.amazonaws.com"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancer。

delete-load-balancer-listeners

以下代码示例演示了如何使用 delete-load-balancer-listeners。

Amazon CLI

从负载均衡器中删除侦听器

此示例从指定的负载均衡器中删除指定端口的侦听器。

命令:

```
aws elb delete-load-balancer-listeners --load-balancer-name my-load-balancer --load-
balancer-ports 80
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancerListeners。

delete-load-balancer-policy

以下代码示例演示了如何使用 delete-load-balancer-policy。

Amazon CLI

从负载均衡器中删除策略

此示例从指定的负载均衡器中删除指定策略。不得在任何侦听器上启用该策略。

命令:

```
aws elb delete-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-duration-cookie-policy
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancerPolicy。

delete-load-balancer

以下代码示例演示了如何使用 delete-load-balancer。

Amazon CLI

删除负载均衡器

此示例删除指定的负载均衡器。

命令:

```
aws elb delete-load-balancer --load-balancer-name my-load-balancer
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancer。

deregister-instances-from-load-balancer

以下代码示例演示了如何使用 deregister-instances-from-load-balancer。

Amazon CLI

从负载均衡器中取消注册实例

此示例从指定负载均衡器中取消注册指定实例。

命令:

```
aws elb deregister-instances-from-load-balancer --load-balancer-name my-load-balancer --instances i-d6f6fae3
```

输出：

```
{
    "Instances": [
        {
            "InstanceId": "i-207d9717"
        },
        {
            "InstanceId": "i-afefb49b"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterInstancesFromLoadBalancer。

describe-account-limits

以下代码示例演示了如何使用 describe-account-limits。

Amazon CLI

描述经典负载均衡器限制

以下 describe-account-limits 示例显示有关 Amazon 账户的经典负载均衡器限制的详细信息。

```
aws elb describe-account-limits
```

输出：

```
{
    "Limits": [
        {
            "Name": "classic-load-balancers",
            "Max": "20"
        },
        {
            "Name": "classic-listeners",
            "Max": "100"
        },
        {
            "Name": "classic-registered-instances",
            "Max": "1000"
```

```
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountLimits。

describe-instance-health

以下代码示例演示了如何使用 describe-instance-health。

Amazon CLI

描述负载均衡器多个实例的运行状况

此示例描述指定负载均衡器多个实例的运行状况。

命令:

```
aws elb describe-instance-health --load-balancer-name my-load-balancer
```

输出：

```
{
  "InstanceStates": [
      {
          "InstanceId": "i-207d9717",
          "ReasonCode": "N/A",
          "State": "InService",
          "Description": "N/A"
      },
      {
          "InstanceId": "i-afefb49b",
          "ReasonCode": "N/A",
          "State": "InService",
          "Description": "N/A"
      }
  ]
}
```

描述负载均衡器单个实例的运行状况

此示例描述指定负载均衡器指定实例的运行状况。

命令:

```
aws elb describe-instance-health --load-balancer-name my-load-balancer --
instances i-7299c809
```

以下是正在注册的实例的示例响应。

输出：

```
{
  "InstanceStates": [
      {
          "InstanceId": "i-7299c809",
          "ReasonCode": "ELB",
          "State": "OutOfService",
          "Description": "Instance registration is still in progress."
    }
  ]
}
```

以下是运行状况不佳的实例的示例响应。

输出：

```
{
  "InstanceStates": [
      {
          "InstanceId": "i-7299c809",
          "ReasonCode": "Instance",
          "State": "OutOfService",
          "Description": "Instance has failed at least the UnhealthyThreshold number
 of health checks consecutively."
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceHealth。

describe-load-balancer-attributes

以下代码示例演示了如何使用 describe-load-balancer-attributes。

Amazon CLI

描述负载均衡器的属性

此示例描述指定负载均衡器的属性。

命令:

```
aws elb describe-load-balancer-attributes --load-balancer-name my-load-balancer
```

输出：

```
{
  "LoadBalancerAttributes": {
      "ConnectionDraining": {
          "Enabled": false,
          "Timeout": 300
      },
      "CrossZoneLoadBalancing": {
          "Enabled": true
      },
      "ConnectionSettings": {
          "IdleTimeout": 30
      },
      "AccessLog": {
          "Enabled": false
    }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancerAttributes。

describe-load-balancer-policies

以下代码示例演示了如何使用 describe-load-balancer-policies。

Amazon CLI

描述与负载均衡器关联的所有策略

此示例描述与指定负载均衡器关联的所有策略。

命令:

```
aws elb describe-load-balancer-policies --load-balancer-name my-load-balancer
```

输出：

```
{
  "PolicyDescriptions": [
    {
      "PolicyAttributeDescriptions": [
        {
          "AttributeName": "ProxyProtocol",
          "AttributeValue": "true"
        }
      ],
      "PolicyName": "my-ProxyProtocol-policy",
      "PolicyTypeName": "ProxyProtocolPolicyType"
    },
    {
        "PolicyAttributeDescriptions": [
            {
                "AttributeName": "CookieName",
                "AttributeValue": "my-app-cookie"
            }
        ],
        "PolicyName": "my-app-cookie-policy",
        "PolicyTypeName": "AppCookieStickinessPolicyType"
    },
    {
      "PolicyAttributeDescriptions": [
        {
          "AttributeName": "CookieExpirationPeriod",
          "AttributeValue": "60"
        }
      ],
      "PolicyName": "my-duration-cookie-policy",
      "PolicyTypeName": "LBCookieStickinessPolicyType"
    },
    .
    .
    .
  ]
}
```

描述与负载均衡器关联的特定策略

此示例描述与指定负载均衡器关联的指定策略。

命令:

```
aws elb describe-load-balancer-policies --load-balancer-name my-load-balancer --
policy-name my-authentication-policy
```

输出：

```
{
  "PolicyDescriptions": [
      {
          "PolicyAttributeDescriptions": [
              {
                  "AttributeName": "PublicKeyPolicyName",
                  "AttributeValue": "my-PublicKey-policy"
              }
          ],
          "PolicyName": "my-authentication-policy",
          "PolicyTypeName": "BackendServerAuthenticationPolicyType"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancerPolicies。

describe-load-balancer-policy-types

以下代码示例演示了如何使用 describe-load-balancer-policy-types。

Amazon CLI

描述由 Elastic Load Balancing 定义的负载均衡器策略类型

此示例描述可用于为负载均衡器创建策略配置的负载均衡器策略类型。

命令:

```
aws elb describe-load-balancer-policy-types
```

输出：

```
{
  "PolicyTypeDescriptions": [
      {
          "PolicyAttributeTypeDescriptions": [
              {
                  "Cardinality": "ONE",
                  "AttributeName": "ProxyProtocol",
                  "AttributeType": "Boolean"
              }
          ],
          "PolicyTypeName": "ProxyProtocolPolicyType",
          "Description": "Policy that controls whether to include the IP address and
 port of the originating request for TCP messages. This policy operates on TCP/SSL
 listeners only"
      },
      {
          "PolicyAttributeTypeDescriptions": [
              {
                  "Cardinality": "ONE",
                  "AttributeName": "PublicKey",
                  "AttributeType": "String"
              }
          ],
          "PolicyTypeName": "PublicKeyPolicyType",
          "Description": "Policy containing a list of public keys to
 accept when authenticating the back-end server(s). This policy cannot be
 applied directly to back-end servers or listeners but must be part of a
 BackendServerAuthenticationPolicyType."
      },
      {
          "PolicyAttributeTypeDescriptions": [
              {
                  "Cardinality": "ONE",
                  "AttributeName": "CookieName",
                  "AttributeType": "String"
              }
          ],
          "PolicyTypeName": "AppCookieStickinessPolicyType",
          "Description": "Stickiness policy with session lifetimes controlled by the
 lifetime of the application-generated cookie. This policy can be associated only
 with HTTP/HTTPS listeners."
      },
```

```
      {
          "PolicyAttributeTypeDescriptions": [
              {
                  "Cardinality": "ZERO_OR_ONE",
                  "AttributeName": "CookieExpirationPeriod",
                  "AttributeType": "Long"
              }
          ],
          "PolicyTypeName": "LBCookieStickinessPolicyType",
          "Description": "Stickiness policy with session lifetimes controlled by
 the browser (user-agent) or a specified expiration period. This policy can be
 associated only with HTTP/HTTPS listeners."
      },
      {
          "PolicyAttributeTypeDescriptions": [
              .
              .
              .
          ],
          "PolicyTypeName": "SSLNegotiationPolicyType",
          "Description": "Listener policy that defines the ciphers and protocols
 that will be accepted by the load balancer. This policy can be associated only with
 HTTPS/SSL listeners."
      },
      {
          "PolicyAttributeTypeDescriptions": [
              {
                  "Cardinality": "ONE_OR_MORE",
                  "AttributeName": "PublicKeyPolicyName",
                  "AttributeType": "PolicyName"
              }
          ],
          "PolicyTypeName": "BackendServerAuthenticationPolicyType",
          "Description": "Policy that controls authentication to back-end server(s)
 and contains one or more policies, such as an instance of a PublicKeyPolicyType.
 This policy can be associated only with back-end servers that are using HTTPS/SSL."
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancerPolicyTypes。

describe-load-balancers

以下代码示例演示了如何使用 describe-load-balancers。

Amazon CLI

描述负载均衡器

此示例描述您的所有负载均衡器。

命令:

```
aws elb describe-load-balancers
```

描述其中一个负载均衡器

此示例描述指定的负载均衡器。

命令:

```
aws elb describe-load-balancers --load-balancer-name my-load-balancer
```

以下示例响应适用于 VPC 中的 HTTPS 负载均衡器。

输出：

```
{
  "LoadBalancerDescriptions": [
    {
      "Subnets": [
          "subnet-15aaab61"
      ],
      "CanonicalHostedZoneNameID": "Z3DZXE0EXAMPLE",
      "CanonicalHostedZoneName": "my-load-balancer-1234567890.us-
west-2.elb.amazonaws.com",
      "ListenerDescriptions": [
          {
              "Listener": {
                  "InstancePort": 80,
                  "LoadBalancerPort": 80,
                  "Protocol": "HTTP",
                  "InstanceProtocol": "HTTP"
              },
              "PolicyNames": []
```

```
            },
            {
                "Listener": {
                    "InstancePort": 443,
                    "SSLCertificateId": "arn:aws:iam::123456789012:server-certificate/
my-server-cert",
                    "LoadBalancerPort": 443,
                    "Protocol": "HTTPS",
                    "InstanceProtocol": "HTTPS"
                },
                "PolicyNames": [
                    "ELBSecurityPolicy-2015-03"
                ]
            }
        ],
        "HealthCheck": {
            "HealthyThreshold": 2,
            "Interval": 30,
            "Target": "HTTP:80/png",
            "Timeout": 3,
            "UnhealthyThreshold": 2
        },
        "VPCId": "vpc-a01106c2",
        "BackendServerDescriptions": [
            {
                "InstancePort": 80,
                "PolicyNames": [
                    "my-ProxyProtocol-policy"
                ]
            }
        ],
        "Instances": [
            {
                "InstanceId": "i-207d9717"
            },
            {
                "InstanceId": "i-afefb49b"
            }
        ],
        "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com",
        "SecurityGroups": [
            "sg-a61988c3"
        ],
        "Policies": {
```

```
            "LBCookieStickinessPolicies": [
                {
                    "PolicyName": "my-duration-cookie-policy",
                    "CookieExpirationPeriod": 60
                }
            ],
            "AppCookieStickinessPolicies": [],
            "OtherPolicies": [
                "my-PublicKey-policy",
                "my-authentication-policy",
                "my-SSLNegotiation-policy",
                "my-ProxyProtocol-policy",
                "ELBSecurityPolicy-2015-03"
            ]
        },
        "LoadBalancerName": "my-load-balancer",
        "CreatedTime": "2015-03-19T03:24:02.650Z",
        "AvailabilityZones": [
            "us-west-2a"
        ],
        "Scheme": "internet-facing",
        "SourceSecurityGroup": {
            "OwnerAlias": "123456789012",
            "GroupName": "my-elb-sg"
        }
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancers。

describe-tags

以下代码示例演示了如何使用 `describe-tags`。

Amazon CLI

描述分配给负载均衡器的标签

此示例描述分配给指定负载均衡器的标签。

命令:

```
aws elb describe-tags --load-balancer-name my-load-balancer
```

输出：

```
{
  "TagDescriptions": [
      {
          "Tags": [
              {
                  "Value": "lima",
                  "Key": "project"
              },
              {
                  "Value": "digital-media",
                  "Key": "department"
              }
          ],
          "LoadBalancerName": "my-load-balancer"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

detach-load-balancer-from-subnets

以下代码示例演示了如何使用 detach-load-balancer-from-subnets。

Amazon CLI

将负载均衡器与子网分离

此示例将指定的负载均衡器与指定的子网分离。

命令:

```
aws elb detach-load-balancer-from-subnets --load-balancer-name my-load-balancer --
subnets subnet-0ecac448
```

输出：

```
{
```

```
    "Subnets": [
        "subnet-15aaab61"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachLoadBalancerFromSubnets。

disable-availability-zones-for-load-balancer

以下代码示例演示了如何使用 disable-availability-zones-for-load-balancer。

Amazon CLI

禁用负载均衡器的可用区

此示例从指定负载均衡器的可用区集中删除指定的可用区。

命令:

```
aws elb disable-availability-zones-for-load-balancer --load-balancer-name my-load-balancer --availability-zones us-west-2a
```

输出：

```
{
    "AvailabilityZones": [
        "us-west-2b"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableAvailabilityZonesForLoadBalancer。

enable-availability-zones-for-load-balancer

以下代码示例演示了如何使用 enable-availability-zones-for-load-balancer。

Amazon CLI

为负载均衡器启用可用区

此示例将指定的可用区添加到指定的负载均衡器。

命令:

```
aws elb enable-availability-zones-for-load-balancer --load-balancer-name my-load-balancer --availability-zones us-west-2b
```

输出：

```
{
    "AvailabilityZones": [
        "us-west-2a",
        "us-west-2b"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableAvailabilityZonesForLoadBalancer。

modify-load-balancer-attributes

以下代码示例演示了如何使用 modify-load-balancer-attributes。

Amazon CLI

修改负载均衡器的属性

此示例修改指定负载均衡器的 CrossZoneLoadBalancing 属性。

命令:

```
aws elb modify-load-balancer-attributes --load-balancer-name my-load-balancer --load-balancer-attributes "{\"CrossZoneLoadBalancing\":{\"Enabled\":true}}"
```

输出：

```
{
    "LoadBalancerAttributes": {
        "CrossZoneLoadBalancing": {
            "Enabled": true
        }
```

```
    },
    "LoadBalancerName": "my-load-balancer"
}
```

此示例修改指定负载均衡器的 ConnectionDraining 属性。

命令:

```
aws elb modify-load-balancer-attributes --load-balancer-name my-load-balancer
 --load-balancer-attributes "{\"ConnectionDraining\":{\"Enabled\":true,\"Timeout
\":300}}"
```

输出：

```
{
    "LoadBalancerAttributes": {
        "ConnectionDraining": {
            "Enabled": true,
            "Timeout": 300
        }
    },
    "LoadBalancerName": "my-load-balancer"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyLoadBalancerAttributes。

register-instances-with-load-balancer

以下代码示例演示了如何使用 register-instances-with-load-balancer。

Amazon CLI

将实例注册到负载均衡器

此示例将指定的实例注册到指定的负载均衡器。

命令:

```
aws elb register-instances-with-load-balancer --load-balancer-name my-load-balancer
 --instances i-d6f6fae3
```

输出：

```
{
   "Instances": [
       {
           "InstanceId": "i-d6f6fae3"
       },
       {
           "InstanceId": "i-207d9717"
       },
       {
           "InstanceId": "i-afefb49b"
       }
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterInstancesWithLoadBalancer。

`remove-tags`

以下代码示例演示了如何使用 `remove-tags`。

Amazon CLI

从负载均衡器中删除标签

此示例从指定的负载均衡器中删除标签。

命令:

```
aws elb remove-tags --load-balancer-name my-load-balancer --tags project
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTags。

`set-load-balancer-listener-ssl-certificate`

以下代码示例演示了如何使用 `set-load-balancer-listener-ssl-certificate`。

Amazon CLI

更新 HTTPS 负载均衡器的 SSL 证书

此示例替换指定 HTTPS 负载均衡器的现有 SSL 证书。

命令:

```
aws elb set-load-balancer-listener-ssl-certificate --load-balancer-
name my-load-balancer --load-balancer-port 443 --ssl-certificate-
id arn:aws:iam::123456789012:server-certificate/new-server-cert
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetLoadBalancerListenerSslCertificate。

set-load-balancer-policies-for-backend-server

以下代码示例演示了如何使用 set-load-balancer-policies-for-backend-server。

Amazon CLI

替换与后端实例的端口关联的策略

此示例替换当前与指定端口关联的策略。

命令:

```
aws elb set-load-balancer-policies-for-backend-server --load-balancer-name my-load-
balancer --instance-port 80 --policy-names my-ProxyProtocol-policy
```

删除当前与后端实例上的端口关联的所有策略

此示例删除与指定端口关联的所有策略。

命令:

```
aws elb set-load-balancer-policies-for-backend-server --load-balancer-name my-load-
balancer --instance-port 80 --policy-names []
```

要确认已删除策略，请使用 describe-load-balancer-policies 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetLoadBalancerPoliciesForBackendServer。

set-load-balancer-policies-of-listener

以下代码示例演示了如何使用 set-load-balancer-policies-of-listener。

Amazon CLI

替换与侦听器关联的策略

此示例替换当前与指定侦听器关联的策略。

命令:

```
aws elb set-load-balancer-policies-of-listener --load-balancer-name my-load-balancer
 --load-balancer-port 443 --policy-names my-SSLNegotiation-policy
```

删除与侦听器关联的所有策略

此示例删除当前与指定侦听器关联的所有策略。

命令:

```
aws elb set-load-balancer-policies-of-listener --load-balancer-name my-load-balancer
 --load-balancer-port 443 --policy-names []
```

要确认已从负载均衡器中删除策略，请使用 `describe-load-balancer-policies` 命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetLoadBalancerPoliciesOfListener。

ELB – 使用 Amazon CLI 的版本 2 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 ELB – 版本 2 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-listener-certificates

以下代码示例演示了如何使用 add-listener-certificates。

Amazon CLI

将证书添加到安全侦听器

此示例将指定的证书添加到指定的安全侦听器。

命令:

```
aws elbv2 add-listener-certificates --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 --certificates CertificateArn=arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705
```

输出：

```
{
  "Certificates": [
      {
          "CertificateArn": "arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705",
          "IsDefault": false
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddListenerCertificates。

add-tags

以下代码示例演示了如何使用 add-tags。

Amazon CLI

将标签添加到负载均衡器

以下 add-tags 示例将 project 和 department 标签添加到指定的负载均衡器。

```
aws elbv2 add-tags \
    --resource-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --tags "Key=project,Value=lima" "Key=department,Value=digital-media"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTags。

create-listener

以下代码示例演示了如何使用 create-listener。

Amazon CLI

示例 1：创建 HTTP 侦听器

以下 create-listener 示例为指定的应用程序负载均衡器创建一个 HTTP 侦听器，用于将请求转发到指定的目标组。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --protocol HTTP \
    --port 80 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的教程：使用 Amazon CLI 创建应用程序负载均衡器。

示例 1：创建 HTTPS 侦听器

以下 create-listener 示例为指定的应用程序负载均衡器创建一个 HTTPS 侦听器，用于将请求转发到指定的目标组。您必须指定 HTTPS 侦听器的 SSL 证书。您可以使用 Amazon Certificate Manager（ACM）创建和管理证书。此外，还可以使用 SSL/TLS 工具创建证书，获取证书颁发机构（CA）签名的证书，并将证书上传至 Amazon Identity and Access Management（IAM）。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --protocol HTTPS \
    --port 443 \
```

```
    --certificates CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557 \
    --ssl-policy ELBSecurityPolicy-2016-08 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的添加 HTTPS 侦听器。

示例 3：创建 TCP 侦听器

以下 `create-listener` 示例为指定的网络负载均衡器创建一个 TCP 侦听器，用于将请求转发到指定的目标组。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-tcp-targets/b6bba954d1361c78
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的教程：使用 Amazon CLI 创建网络负载均衡器。

示例 4：创建 TLS 侦听器

以下 `create-listener` 示例为指定的网络负载均衡器创建一个 TLS 侦听器，用于将请求转发到指定的目标组。您必须指定 TLS 侦听器的 SSL 证书。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --protocol TLS \
    --port 443 \
    --certificates CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557 \
    --ssl-policy ELBSecurityPolicy-2016-08 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的 网络负载均衡器的 TLS 侦听器。

示例 5：创建 UDP 侦听器

以下 create-listener 示例为指定的网络负载均衡器创建一个 UDP 侦听器，用于将请求转发到指定的目标组。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e \
    --protocol UDP \
    --port 53 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-tcp-targets/b6bba954d1361c78
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的教程：使用 Amazon CLI 创建网络负载均衡器。

示例 6：为指定的网关和转发创建侦听器

以下 create-listener 示例为指定的网关负载均衡器创建一个侦听器，用于将请求转发到指定的目标组。

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
east-1:850631746142:loadbalancer/gwy/my-gateway-load-balancer/e0f9b3d5c7f7d3d6 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
east-1:850631746142:targetgroup/my-glb-targets/007ca469fae3bb1615
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:listener/gwy/my-agw-lb-example2/e0f9b3d5c7f7d3d6/
afc127db15f925de",
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:loadbalancer/gwy/my-agw-lb-example2/e0f9b3d5c7f7d3d6",
            "DefaultActions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:targetgroup/test-tg-agw-2/007ca469fae3bb1615",
                    "ForwardConfig": {
                        "TargetGroups": [
```

```
                            {
                                "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:targetgroup/test-tg-agw-2/007ca469fae3bb1615"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《网关负载均衡器用户指南》中的通过 Amazon CLI 使用网关负载均衡器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateListener。

`create-load-balancer`

以下代码示例演示了如何使用 `create-load-balancer`。

Amazon CLI

示例 1：创建面向 Internet 的负载均衡器

以下 `create-load-balancer` 示例创建一个面向 Internet 的应用程序负载均衡器，并为指定的子网启用可用区。

```
aws elbv2 create-load-balancer \
    --name my-load-balancer \
    --subnets subnet-b7d581c0 subnet-8360a9e7
```

输出：

```
{
    "LoadBalancers": [
        {
            "Type": "application",
            "Scheme": "internet-facing",
            "IpAddressType": "ipv4",
            "VpcId": "vpc-3ac0fb5f",
            "AvailabilityZones": [
```

```
                {
                    "ZoneName": "us-west-2a",
                    "SubnetId": "subnet-8360a9e7"
                },
                {
                    "ZoneName": "us-west-2b",
                    "SubnetId": "subnet-b7d581c0"
                }
            ],
            "CreatedTime": "2017-08-25T21:26:12.920Z",
            "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",
            "DNSName": "my-load-balancer-424835706.us-west-2.elb.amazonaws.com",
            "SecurityGroups": [
                "sg-5943793c"
            ],
            "LoadBalancerName": "my-load-balancer",
            "State": {
                "Code": "provisioning"
            },
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188"
        }
    ]
}
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的教程：使用 Amazon CLI 创建应用程序负载均衡器。

示例 2：创建内部负载均衡器

以下 `create-load-balancer` 示例创建一个内部应用程序负载均衡器，并为指定的子网启用可用区。

```
aws elbv2 create-load-balancer \
    --name my-internal-load-balancer \
    --scheme internal \
    --subnets subnet-b7d581c0 subnet-8360a9e7
```

输出：

```
{
    "LoadBalancers": [
        {
```

```
            "Type": "application",
            "Scheme": "internal",
            "IpAddressType": "ipv4",
            "VpcId": "vpc-3ac0fb5f",
            "AvailabilityZones": [
                {
                    "ZoneName": "us-west-2a",
                    "SubnetId": "subnet-8360a9e7"
                },
                {
                    "ZoneName": "us-west-2b",
                    "SubnetId": "subnet-b7d581c0"
                }
            ],
            "CreatedTime": "2016-03-25T21:29:48.850Z",
            "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",
            "DNSName": "internal-my-internal-load-balancer-1529930873.us-
west-2.elb.amazonaws.com",
            "SecurityGroups": [
                "sg-5943793c"
            ],
            "LoadBalancerName": "my-internal-load-balancer",
            "State": {
                "Code": "provisioning"
            },
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-internal-load-balancer/5b49b8d4303115c2"
        }
    ]
}
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的教程：使用 Amazon CLI 创建应用程序负载均衡器。

示例 3：创建网络负载均衡器

以下 `create-load-balancer` 示例创建一个面向 Internet 的网络负载均衡器，并为指定的子网启用可用区。它使用子网映射，将指定的弹性 IP 地址与可用区的负载均衡器节点使用的网络接口相关联。

```
aws elbv2 create-load-balancer \
    --name my-network-load-balancer \
    --type network \
```

```
    --subnet-mappings SubnetId=subnet-b7d581c0,AllocationId=eipalloc-64d5890a
```

输出：

```
{
    "LoadBalancers": [
        {
            "Type": "network",
            "Scheme": "internet-facing",
            "IpAddressType": "ipv4",
            "VpcId": "vpc-3ac0fb5f",
            "AvailabilityZones": [
                {
                    "LoadBalancerAddresses": [
                        {
                            "IpAddress": "35.161.207.171",
                            "AllocationId": "eipalloc-64d5890a"
                        }
                    ],
                    "ZoneName": "us-west-2b",
                    "SubnetId": "subnet-5264e837"
                }
            ],
            "CreatedTime": "2017-10-15T22:41:25.657Z",
            "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",
            "DNSName": "my-network-load-balancer-5d1b75f4f1cee11e.elb.us-
west-2.amazonaws.com",
            "LoadBalancerName": "my-network-load-balancer",
            "State": {
                "Code": "provisioning"
            },
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e"
        }
    ]
}
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的教程：使用 Amazon CLI 创建网络负载均衡器。

示例 4：创建网关负载均衡器

以下 `create-load-balancer` 示例创建一个网关负载均衡器，并为指定的子网启用可用区。

```
aws elbv2 create-load-balancer \
    --name my-gateway-load-balancer \
    --type gateway \
    --subnets subnet-dc83f691 subnet-a62583f9
```

输出：

```
{
    "LoadBalancers": [
        {
            "Type": "gateway",
            "VpcId": "vpc-838475fe",
            "AvailabilityZones": [
                {
                    "ZoneName": "us-east-1b",
                    "SubnetId": "subnet-a62583f9"
                },
            {
                    "ZoneName": "us-east-1a",
                    "SubnetId": "subnet-dc83f691"
                }
            ],
            "CreatedTime": "2021-07-14T19:33:43.324000+00:00",
            "LoadBalancerName": "my-gateway-load-balancer",
            "State": {
                "Code": "provisioning"
            },
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:loadbalancer/gwy/my-gateway-load-balancer/dfbb5a7d32cdee79"
        }
    ]
}
```

有关更多信息，请参阅《网关负载均衡器用户指南》中的通过 Amazon CLI 使用网关负载均衡器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancer。

create-rule

以下代码示例演示了如何使用 create-rule。

Amazon CLI

示例 1：使用路径条件和转发操作创建规则

以下 create-rule 示例创建一个规则，当 URL 包含指定的模式时，会将请求转发到指定的目标组。

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
    --priority 5 \
    --conditions file://conditions-pattern.json
    --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

的内容conditions-pattern.json：

```
[
    {
        "Field": "path-pattern",
        "PathPatternConfig": {
            "Values": ["/images/*"]
        }
    }
]
```

示例 2：使用主机条件和固定响应创建规则

以下 create-rule 示例创建一个规则，当主机标头中的主机名与指定的主机名相匹配时，会提供固定响应。

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
    --priority 10 \
    --conditions file://conditions-host.json \
    --actions file://actions-fixed-response.json
```

conditions-host.json 的内容

```
[
```

```
  {
      "Field": "host-header",
      "HostHeaderConfig": {
          "Values": ["*.example.com"]
      }
  }
]
```

actions-fixed-response.json 的内容

```
[
    {
        "Type": "fixed-response",
        "FixedResponseConfig": {
            "MessageBody": "Hello world",
            "StatusCode": "200",
            "ContentType": "text/plain"
        }
    }
]
```

示例 3：使用源 IP 地址条件、身份验证操作和转发操作创建规则

以下 create-rule 示例创建一个规则，该规则用于在源 IP 地址与指定的 IP 地址相匹配时对用户进行身份验证，如果身份验证成功，则将请求转发到指定的目标组。

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
    --priority 20 \
    --conditions file://conditions-source-ip.json \
    --actions file://actions-authenticate.json
```

conditions-source-ip.json 的内容

```
[
    {
        "Field": "source-ip",
        "SourceIpConfig": {
            "Values": ["192.0.2.0/24", "198.51.100.10/32"]
        }
    }
```

```
]
```

actions-authenticate.json 的内容

```
[
    {
        "Type": "authenticate-oidc",
        "AuthenticateOidcConfig": {
            "Issuer": "https://idp-issuer.com",
            "AuthorizationEndpoint": "https://authorization-endpoint.com",
            "TokenEndpoint": "https://token-endpoint.com",
            "UserInfoEndpoint": "https://user-info-endpoint.com",
            "ClientId": "abcdefghijklmnopqrstuvwxyz123456789",
            "ClientSecret": "12345678901234567890123456789O",
            "SessionCookieName": "my-cookie",
            "SessionTimeout": 3600,
            "Scope": "email",
            "AuthenticationRequestExtraParams": {
                "display": "page",
                "prompt": "login"
            },
            "OnUnauthenticatedRequest": "deny"
        },
        "Order": 1
    },
    {
        "Type": "forward",
        "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:880185128111:targetgroup/cli-test/642a97ecb0e0f26b",
        "Order": 2
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRule。

create-target-group

以下代码示例演示了如何使用 create-target-group。

Amazon CLI

示例 1：为应用程序负载均衡器创建目标组

以下 create-target-group 示例为应用程序负载均衡器创建目标组，以便您按实例 ID（目标类型为 instance）注册目标。此目标组使用 HTTP 协议、端口 80 和 HTTP 目标组的默认运行状况检查设置。

```
aws elbv2 create-target-group \
    --name my-targets \
    --protocol HTTP \
    --port 80 \
    --target-type instance \
    --vpc-id vpc-3ac0fb5f
```

输出：

```
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
            "TargetGroupName": "my-targets",
            "Protocol": "HTTP",
            "Port": 80,
            "VpcId": "vpc-3ac0fb5f",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "HealthCheckPath": "/",
            "Matcher": {
                "HttpCode": "200"
            },
            "TargetType": "instance",
            "ProtocolVersion": "HTTP1",
            "IpAddressType": "ipv4"
        }
    ]
}
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的创建目标组。

示例 2：创建目标组以将流量从应用程序负载均衡器路由到 Lambda 函数

以下 create-target-group 示例为应用程序负载均衡器创建目标组，在其中目标为 Lambda 函数（目标类型为 lambda）。默认情况下，为此目标组禁用运行状况检查。

```
aws elbv2 create-target-group \
    --name my-lambda-target \
    --target-type lambda
```

输出：

```
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-lambda-target/a3003e085dbb8ddc",
            "TargetGroupName": "my-lambda-target",
            "HealthCheckEnabled": false,
            "HealthCheckIntervalSeconds": 35,
            "HealthCheckTimeoutSeconds": 30,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "HealthCheckPath": "/",
            "Matcher": {
                "HttpCode": "200"
            },
            "TargetType": "lambda",
            "IpAddressType": "ipv4"
        }
    ]
}
```

有关更多信息，请参阅应用程序负载均衡器用户指南中的 Lambda 函数作为目标。

示例 3：为网络负载均衡器创建目标组

以下 create-target-group 示例为网络负载均衡器创建目标组，以便您按 IP 地址（目标类型为 ip）注册目标。此目标组使用 TCP 协议、端口 80 和 TCP 目标组的默认运行状况检查设置。

```
aws elbv2 create-target-group \
    --name my-ip-targets \
    --protocol TCP \
```

```
    --port 80 \
    --target-type ip \
    --vpc-id vpc-3ac0fb5f
```

输出：

```
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-ip-targets/b6bba954d1361c78",
            "TargetGroupName": "my-ip-targets",
            "Protocol": "TCP",
            "Port": 80,
            "VpcId": "vpc-3ac0fb5f",
            "HealthCheckEnabled": true,
            "HealthCheckProtocol": "TCP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 10,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "TargetType": "ip",
            "IpAddressType": "ipv4"
        }
    ]
}
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的创建目标组。

示例 4：创建目标组以将流量从网络负载均衡器路由到应用程序负载均衡器

以下 `create-target-group` 示例为网络负载均衡器创建目标组，您将在其中将应用程序负载均衡器注册为目标（目标类型为 `alb`）。

aws elbv2 create-target-group --name my-alb-target --protocol TCP --port 80 --target-type alb --vpc-id vpc-3ac0fb5f

输出：

```
{
    "TargetGroups": [
        {
```

```
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-alb-target/a3003e085dbb8ddc",
            "TargetGroupName": "my-alb-target",
            "Protocol": "TCP",
            "Port": 80,
            "VpcId": "vpc-838475fe",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 6,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "HealthCheckPath": "/",
            "Matcher": {
                "HttpCode": "200-399"
            },
            "TargetType": "alb",
            "IpAddressType": "ipv4"
        }
    ]
}
```

有关更多信息，请参阅《网络负载均衡器用户指南》中的以应用程序负载均衡器作为目标创建目标组。

示例 5：为网关负载均衡器创建目标组

以下 `create-target-group` 示例为网关负载均衡器创建目标组，其中目标为实例，目标组协议为 GENEVE。

```
aws elbv2 create-target-group \
    --name my-glb-targetgroup \
    --protocol GENEVE \
    --port 6081 \
    --target-type instance \
    --vpc-id vpc-838475fe
```

输出：

```
{
    "TargetGroups": [
        {
```

```
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-glb-targetgroup/00c3d57eacd6f40b6f",
            "TargetGroupName": "my-glb-targetgroup",
            "Protocol": "GENEVE",
            "Port": 6081,
            "VpcId": "vpc-838475fe",
            "HealthCheckProtocol": "TCP",
            "HealthCheckPort": "80",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 10,
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "TargetType": "instance"
        }
    ]
}
```

有关更多信息，请参阅《网关负载均衡器用户指南》中的“创建目标组”<https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/create-target-group.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTargetGroup。

delete-listener

以下代码示例演示了如何使用 delete-listener。

Amazon CLI

删除侦听器

以下 delete-listener 示例删除指定的侦听器。

```
aws elbv2 delete-listener \
    --listener-arn arn:aws:elasticloadbalancing:ua-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteListener。

delete-load-balancer

以下代码示例演示了如何使用 delete-load-balancer。

Amazon CLI

删除负载均衡器

以下 delete-load-balancer 示例将删除指定的负载均衡器。

```
aws elbv2 delete-load-balancer \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancer。

delete-rule

以下代码示例演示了如何使用 delete-rule。

Amazon CLI

删除规则

以下 delete-rule 示例将删除指定的规则。

```
aws elbv2 delete-rule \
    --rule-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/
app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/1291d13826f405c3
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRule。

delete-target-group

以下代码示例演示了如何使用 delete-target-group。

Amazon CLI

删除目标组

以下 delete-target-group 示例删除指定的目标组。

```
aws elbv2 delete-target-group \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

此命令不生成任何输出。

有关更多信息，请参阅《应用程序负载均衡器指南》中的删除负载均衡器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTargetGroup。

deregister-targets

以下代码示例演示了如何使用 deregister-targets。

Amazon CLI

示例 1：从目标组中取消注册目标

以下 deregister-targets 示例从指定的目标组中删除指定的实例。

```
aws elbv2 deregister-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
    --targets Id=i-1234567890abcdef0
```

示例 2：取消注册使用端口覆盖注册的目标

以下 deregister-targets 示例从使用端口覆盖注册的目标组中删除实例。

```
aws elbv2 deregister-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-internal-targets/3bb63f11dfb0faf9 \
    --targets Id=i-1234567890abcdef0,Port=80 Id=i-1234567890abcdef0,Port=766
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTargets。

describe-account-limits

以下代码示例演示了如何使用 describe-account-limits。

Amazon CLI

描述 Elastic Load Balancing 限制

以下 describe-account-limits 示例显示了当前区域中 Amazon 账户的 Elastic Load Balancing 限制。

```
aws elbv2 describe-account-limits
```

输出：

```
{
    "Limits": [
        {
            "Name": "target-groups",
            "Max": "3000"
        },
        {
            "Name": "targets-per-application-load-balancer",
            "Max": "1000"
        },
        {
            "Name": "listeners-per-application-load-balancer",
            "Max": "50"
        },
        {
            "Name": "rules-per-application-load-balancer",
            "Max": "100"
        },
        {
            "Name": "network-load-balancers",
            "Max": "50"
        },
        {
            "Name": "targets-per-network-load-balancer",
            "Max": "3000"
        },
        {
            "Name": "targets-per-availability-zone-per-network-load-balancer",
            "Max": "500"
        },
        {
            "Name": "listeners-per-network-load-balancer",
            "Max": "50"
        },
        {
            "Name": "condition-values-per-alb-rule",
            "Max": "5"
        },
        {
```

```
            "Name": "condition-wildcards-per-alb-rule",
            "Max": "5"
        },
        {
            "Name": "target-groups-per-application-load-balancer",
            "Max": "100"
        },
        {
            "Name": "target-groups-per-action-on-application-load-balancer",
            "Max": "5"
        },
        {
            "Name": "target-groups-per-action-on-network-load-balancer",
            "Max": "1"
        },
        {
            "Name": "certificates-per-application-load-balancer",
            "Max": "25"
        },
        {
            "Name": "certificates-per-network-load-balancer",
            "Max": "25"
        },
        {
            "Name": "targets-per-target-group",
            "Max": "1000"
        },
        {
            "Name": "target-id-registrations-per-application-load-balancer",
            "Max": "1000"
        },
        {
            "Name": "network-load-balancer-enis-per-vpc",
            "Max": "1200"
        },
        {
            "Name": "application-load-balancers",
            "Max": "50"
        },
        {
            "Name": "gateway-load-balancers",
            "Max": "100"
        },
        {
```

```
            "Name": "gateway-load-balancers-per-vpc",
            "Max": "100"
        },
        {
            "Name": "geneve-target-groups",
            "Max": "100"
        },
        {
            "Name": "targets-per-availability-zone-per-gateway-load-balancer",
            "Max": "300"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 一般参考》中的配额。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountLimits。

describe-listener-certificates

以下代码示例演示了如何使用 describe-listener-certificates。

Amazon CLI

描述安全侦听器的证书

此示例描述指定的安全侦听器的证书。

命令:

```
aws elbv2 describe-listener-certificates --listener-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-
balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

输出：

```
{
  "Certificates": [
      {
          "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705",
          "IsDefault": false
```

```
      },
      {
          "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557",
          "IsDefault": false
      },
      {
          "CertificateArn": "arn:aws:acm:us-west-2:123456789012:certificate/
fe59da96-6f58-4a22-8eed-6d0d50477e1d",
          "IsDefault": true
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeListenerCertificates。

`describe-listeners`

以下代码示例演示了如何使用 `describe-listeners`。

Amazon CLI

描述侦听器

此示例描述指定的侦听器。

命令:

```
aws elbv2 describe-listeners --listener-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

输出：

```
{
  "Listeners": [
      {
          "Port": 80,
          "Protocol": "HTTP",
          "DefaultActions": [
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
```

```
                    "Type": "forward"
                }
            ],
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"
        }
    ]
}
```

描述负载均衡器的侦听器

此示例描述指定负载均衡器的侦听器。

命令:

```
aws elbv2 describe-listeners --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

输出：

```
{
    "Listeners": [
        {
            "Port": 443,
            "Protocol": "HTTPS",
            "DefaultActions": [
                {
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
                    "Type": "forward"
                }
            ],
            "SslPolicy": "ELBSecurityPolicy-2015-05",
            "Certificates": [
                {
                    "CertificateArn": "arn:aws:iam::123456789012:server-certificate/
my-server-cert"
                }
            ],
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
```

```
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/0467ef3c8400ae65"
        },
        {
            "Port": 80,
            "Protocol": "HTTP",
            "DefaultActions": [
                {
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
                    "Type": "forward"
                }
            ],
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeListeners。

describe-load-balancer-attributes

以下代码示例演示了如何使用 describe-load-balancer-attributes。

Amazon CLI

描述负载均衡器属性

以下 describe-load-balancer-attributes 示例显示指定负载均衡器的属性。

```
aws elbv2 describe-load-balancer-attributes \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

以下输出示例显示了应用程序负载均衡器的属性。

```
{
    "Attributes": [
        {
```

```
            "Value": "false",
            "Key": "access_logs.s3.enabled"
        },
        {
            "Value": "",
            "Key": "access_logs.s3.bucket"
        },
        {
            "Value": "",
            "Key": "access_logs.s3.prefix"
        },
        {
            "Value": "60",
            "Key": "idle_timeout.timeout_seconds"
        },
        {
            "Value": "false",
            "Key": "deletion_protection.enabled"
        },
        {
            "Value": "true",
            "Key": "routing.http2.enabled"
        }
    ]
}
```

以下输出示例包含网络负载均衡器的属性。

```
{
    "Attributes": [
        {
            "Value": "false",
            "Key": "access_logs.s3.enabled"
        },
        {
            "Value": "",
            "Key": "access_logs.s3.bucket"
        },
        {
            "Value": "",
            "Key": "access_logs.s3.prefix"
        },
        {
```

```
            "Value": "false",
            "Key": "deletion_protection.enabled"
        },
        {
            "Value": "false",
            "Key": "load_balancing.cross_zone.enabled"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancerAttributes。

describe-load-balancers

以下代码示例演示了如何使用 describe-load-balancers。

Amazon CLI

描述负载均衡器

此示例描述指定的负载均衡器。

命令:

```
aws elbv2 describe-load-balancers --load-balancer-
arns arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188
```

输出：

```
{
  "LoadBalancers": [
      {
          "Type": "application",
          "Scheme": "internet-facing",
          "IpAddressType": "ipv4",
          "VpcId": "vpc-3ac0fb5f",
          "AvailabilityZones": [
              {
                  "ZoneName": "us-west-2a",
                  "SubnetId": "subnet-8360a9e7"
              },
```

```
            {
                "ZoneName": "us-west-2b",
                "SubnetId": "subnet-b7d581c0"
            }
        ],
        "CreatedTime": "2016-03-25T21:26:12.920Z",
        "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",
        "DNSName": "my-load-balancer-424835706.us-west-2.elb.amazonaws.com",
        "SecurityGroups": [
            "sg-5943793c"
        ],
        "LoadBalancerName": "my-load-balancer",
        "State": {
            "Code": "active"
        },
        "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188"
      }
  ]
}
```

描述所有负载均衡器

此示例描述您的所有负载均衡器。

命令:

```
aws elbv2 describe-load-balancers
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoadBalancers。

describe-rules

以下代码示例演示了如何使用 `describe-rules`。

Amazon CLI

示例 1：描述规则

以下 `describe-rules` 示例显示指定规则的详细信息。

```
aws elbv2 describe-rules \
```

```
    --rule-arns arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/
app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee
```

示例 2：描述侦听器的规则

以下 describe-rules 示例显示指定侦听器的规则的详细信息。输出包括默认规则和您添加的任何其他规则。

```
aws elbv2 describe-rules \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRules。

describe-ssl-policies

以下代码示例演示了如何使用 describe-ssl-policies。

Amazon CLI

示例 1：按负载均衡器类型列出用于 SSL 协商的策略

以下 describe-ssl-policies 示例显示可用于与应用程序负载均衡器进行 SSL 协商的策略的名称。该示例使用 --query 参数仅显示策略的名称。

```
aws elbv2 describe-ssl-policies \
    --load-balancer-type application \
    --query SslPolicies[*].Name
```

输出：

```
[
    "ELBSecurityPolicy-2016-08",
    "ELBSecurityPolicy-TLS13-1-2-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Res-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06",
    "ELBSecurityPolicy-TLS13-1-1-2021-06",
    "ELBSecurityPolicy-TLS13-1-0-2021-06",
    "ELBSecurityPolicy-TLS13-1-3-2021-06",
    "ELBSecurityPolicy-TLS-1-2-2017-01",
```

```
    "ELBSecurityPolicy-TLS-1-1-2017-01",
    "ELBSecurityPolicy-TLS-1-2-Ext-2018-06",
    "ELBSecurityPolicy-FS-2018-06",
    "ELBSecurityPolicy-2015-05",
    "ELBSecurityPolicy-TLS-1-0-2015-04",
    "ELBSecurityPolicy-FS-1-2-Res-2019-08",
    "ELBSecurityPolicy-FS-1-1-2019-08",
    "ELBSecurityPolicy-FS-1-2-2019-08",
    "ELBSecurityPolicy-FS-1-2-Res-2020-10"
]
```

示例 2：列出支持特定协议的策略

以下 describe-ssl-policies 示例显示支持 TLS 1.3 协议的策略的名称。该示例使用 --query 参数仅显示策略的名称。

```
aws elbv2 describe-ssl-policies \
    --load-balancer-type application \
    --query SslPolicies[?contains(SslProtocols,'TLSv1.3')].Name
```

输出：

```
[
    "ELBSecurityPolicy-TLS13-1-2-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Res-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06",
    "ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06",
    "ELBSecurityPolicy-TLS13-1-1-2021-06",
    "ELBSecurityPolicy-TLS13-1-0-2021-06",
    "ELBSecurityPolicy-TLS13-1-3-2021-06"
]
```

示例 3：显示策略的密码

以下 describe-ssl-policies 示例显示指定策略的密码名称。该示例使用 --query 参数仅显示密码名称。列表中第一个密码的优先级为 1，其余密码按优先级顺序排列。

```
aws elbv2 describe-ssl-policies \
    --names ELBSecurityPolicy-TLS13-1-2-2021-06 \
    --query SslPolicies[*].Ciphers[*].Name
```

输出：

```
[
    "TLS_AES_128_GCM_SHA256",
    "TLS_AES_256_GCM_SHA384",
    "TLS_CHACHA20_POLY1305_SHA256",
    "ECDHE-ECDSA-AES128-GCM-SHA256",
    "ECDHE-RSA-AES128-GCM-SHA256",
    "ECDHE-ECDSA-AES128-SHA256",
    "ECDHE-RSA-AES128-SHA256",
    "ECDHE-ECDSA-AES256-GCM-SHA384",
    "ECDHE-RSA-AES256-GCM-SHA384",
    "ECDHE-ECDSA-AES256-SHA384",
    "ECDHE-RSA-AES256-SHA384"
]
```

有关更多信息，请参阅《应用程序负载均衡器用户指南》中的安全策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSslPolicies。

`describe-tags`

以下代码示例演示了如何使用 `describe-tags`。

Amazon CLI

描述分配给负载均衡器的标签

此示例描述分配给指定负载均衡器的标签。

命令:

```
aws elbv2 describe-tags --resource-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

输出：

```
{
  "TagDescriptions": [
      {
          "ResourceArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
          "Tags": [
              {
                  "Value": "lima",
```

```
                "Key": "project"
            },
            {
                "Value": "digital-media",
                "Key": "department"
            }
        ]
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

describe-target-group-attributes

以下代码示例演示了如何使用 describe-target-group-attributes。

Amazon CLI

描述目标组属性

以下 describe-target-group-attributes 示例显示指定目标组的属性。

```
aws elbv2 describe-target-group-attributes \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

如果协议为 HTTP 或 HTTPS，且目标类型为 instance 或 ip，则输出包含属性。

```
{
    "Attributes": [
        {
            "Value": "false",
            "Key": "stickiness.enabled"
        },
        {
            "Value": "300",
            "Key": "deregistration_delay.timeout_seconds"
        },
        {
            "Value": "lb_cookie",
            "Key": "stickiness.type"
        },
```

```
        {
            "Value": "86400",
            "Key": "stickiness.lb_cookie.duration_seconds"
        },
        {
            "Value": "0",
            "Key": "slow_start.duration_seconds"
        }
    ]
}
```

如果协议为 HTTP 或 HTTPS，且目标类型为 `lambda`，则以下输出包含属性。

```
{
    "Attributes": [
        {
            "Value": "false",
            "Key": "lambda.multi_value_headers.enabled"
        }
    ]
}
```

如果协议为 TCP、TLS、UDP 或 TCP_UDP，则以下输出将包含属性。

```
{
    "Attributes": [
        {
            "Value": "false",
            "Key": "proxy_protocol_v2.enabled"
        },
        {
            "Value": "300",
            "Key": "deregistration_delay.timeout_seconds"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTargetGroupAttributes。

`describe-target-groups`

以下代码示例演示了如何使用 `describe-target-groups`。

Amazon CLI

示例 1：描述目标组

以下 describe-target-groups 示例显示指定目标组的详细信息。

```
aws elbv2 describe-target-groups \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

输出：

```
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
            "TargetGroupName": "my-targets",
            "Protocol": "HTTP",
            "Port": 80,
            "VpcId": "vpc-3ac0fb5f",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "HealthCheckPath": "/",
            "Matcher": {
                "HttpCode": "200"
            },
            "LoadBalancerArns": [
                "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/
app/my-load-balancer/50dc6c495c0c9188"
            ],
            "TargetType": "instance",
            "ProtocolVersion": "HTTP1",
            "IpAddressType": "ipv4"
        }
    ]
}
```

示例 2：描述负载均衡器的所有目标组

以下 describe-target-groups 示例显示指定负载均衡器所有目标组的详细信息。该示例使用 --query 参数仅显示目标组名称。

```
aws elbv2 describe-target-groups \
    --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --query TargetGroups[*].TargetGroupName
```

输出：

```
[
    "my-instance-targets",
    "my-ip-targets",
    "my-lambda-target"
]
```

有关更多信息，请参阅《应用程序负载均衡器指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTargetGroups。

describe-target-health

以下代码示例演示了如何使用 describe-target-health。

Amazon CLI

示例 1：描述目标组中目标的运行状况

以下 describe-target-health 示例显示指定目标组中目标的运行状况详细信息。这些目标运行状况良好。

```
aws elbv2 describe-target-health \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

输出：

```
{
```

```
    "TargetHealthDescriptions": [
        {
            "HealthCheckPort": "80",
            "Target": {
                "Id": "i-ceddcd4d",
                "Port": 80
            },
            "TargetHealth": {
                "State": "healthy"
            }
        },
        {
            "HealthCheckPort": "80",
            "Target": {
                "Id": "i-0f76fade",
                "Port": 80
            },
            "TargetHealth": {
                "State": "healthy"
            }
        }
    ]
}
```

示例 2：描述目标的运行状况

以下 describe-target-health 示例显示指定目标的运行状况详细信息。此目标运行正常。

```
aws elbv2 describe-target-health \
    --targets Id=i-0f76fade,Port=80 \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

输出：

```
{
    "TargetHealthDescriptions": [
        {
            "HealthCheckPort": "80",
            "Target": {
                "Id": "i-0f76fade",
                "Port": 80
            },
```

```
            "TargetHealth": {
                "State": "healthy"
            }
        }
    ]
}
```

以下示例输出适用于未在侦听器的操作中指定目标组的目标。此目标无法接收来自负载均衡器的流量。

```
{
    "TargetHealthDescriptions": [
    {
        "HealthCheckPort": "80",
        "Target": {
            "Id": "i-0f76fade",
            "Port": 80
        },
            "TargetHealth": {
                "State": "unused",
                "Reason": "Target.NotInUse",
                "Description": "Target group is not configured to receive traffic
 from the load balancer"
            }
        }
    ]
}
```

以下示例输出适用于仅在侦听器的操作中指定目标组的目标。该目标仍在注册中。

```
{
    "TargetHealthDescriptions": [
        {
            "HealthCheckPort": "80",
            "Target": {
                "Id": "i-0f76fade",
                "Port": 80
            },
            "TargetHealth": {
                "State": "initial",
                "Reason": "Elb.RegistrationInProgress",
                "Description": "Target registration is in progress"
            }
```

```
        }
    ]
}
```

以下示例输出适用于运行状况不佳的目标。

```
{
    "TargetHealthDescriptions": [
        {
            "HealthCheckPort": "80",
            "Target": {
                "Id": "i-0f76fade",
                "Port": 80
            },
            "TargetHealth": {
                "State": "unhealthy",
                "Reason": "Target.Timeout",
                "Description": "Connection to target timed out"
            }
        }
    ]
}
```

以下示例输出针对的目标是 Lambda 函数且运行状况检查已禁用。

```
{
    "TargetHealthDescriptions": [
        {
            "Target": {
                "Id": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
                "AvailabilityZone": "all",
            },
            "TargetHealth": {
                "State": "unavailable",
                "Reason": "Target.HealthCheckDisabled",
                "Description": "Health checks are not enabled for this target"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTargetHealth。

modify-listener

以下代码示例演示了如何使用 modify-listener。

Amazon CLI

示例 1：将默认操作更改为转发操作

以下 modify-listener 示例将指定侦听器的默认操作更改为 forward 操作。

```
aws elbv2 modify-listener \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
    --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2",
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
            "Protocol": "HTTP",
            "Port": 80,
            "DefaultActions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f"
                }
            ]
        }
    ]
}
```

示例 2：将默认操作更改为重定向操作

以下 modify-listener 示例将指定侦听器的默认操作更改为 redirect 操作。

```
aws elbv2 modify-listener \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
    --default-actions Type=redirect,
 RedirectConfig='{Protocol=HTTPS,StatusCode=HTTP_302}'
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/
f2f7dc8efc522ab2",
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
            "Protocol": "HTTP",
            "Port": 80,
            "DefaultActions": [
                {
                    "Type": "redirect",
                    "RedirectConfig": {
                        "Protocol": "HTTPS",
                        "Port": "#{port}",
                        "Host": "#{host}",
                        "Path": "/#{path}",
                        "Query": "#{query}",
                        "StatusCode": "HTTP_302",
                    }
                }
            ]
        }
    ]
}
```

示例 3：更改服务器证书

以下 modify-listener 示例更改指定 HTTPS 侦听器的服务器证书。

```
aws elbv2 modify-listener \
    --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/0467ef3c8400ae65 \
```

```
    --certificates CertificateArn=arn:aws:iam::123456789012:server-certificate/my-
new-server-cert
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-
balancer/50dc6c495c0c9188/0467ef3c8400ae65",
            "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
            "Protocol": "HTTPS",
            "Port": 443,
            "DefaultActions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067"
                }
            ],
            "SslPolicy": "ELBSecurityPolicy-2015-05",
            "Certificates": [
                {
                    "CertificateArn": "arn:aws:iam::123456789012:server-certificate/
my-new-server-cert"
                }
            ],
        }
    ]
}
```

有关更多信息，请参阅《Application Load Balancers User Guide》中的 Listener rules。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyListener。

modify-load-balancer-attributes

以下代码示例演示了如何使用 modify-load-balancer-attributes。

Amazon CLI

启用删除保护

此示例为指定负载均衡器启用删除保护。

命令:

```
aws elbv2 modify-load-balancer-attributes --load-balancer-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188 --attributes Key=deletion_protection.enabled,Value=true
```

输出：

```
{
  "Attributes": [
      {
          "Value": "true",
          "Key": "deletion_protection.enabled"
      },
      {
          "Value": "false",
          "Key": "access_logs.s3.enabled"
      },
      {
          "Value": "60",
          "Key": "idle_timeout.timeout_seconds"
      },
      {
          "Value": "",
          "Key": "access_logs.s3.prefix"
      },
      {
          "Value": "",
          "Key": "access_logs.s3.bucket"
      }
  ]
}
```

更改空闲超时

此示例更改指定负载均衡器的空闲超时值。

命令:

```
aws elbv2 modify-load-balancer-attributes --load-balancer-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188 --attributes Key=idle_timeout.timeout_seconds,Value=30
```

输出：

```
{
  "Attributes": [
      {
          "Value": "30",
          "Key": "idle_timeout.timeout_seconds"
      },
      {
          "Value": "false",
          "Key": "access_logs.s3.enabled"
      },
      {
          "Value": "",
          "Key": "access_logs.s3.prefix"
      },
      {
          "Value": "true",
          "Key": "deletion_protection.enabled"
      },
      {
          "Value": "",
          "Key": "access_logs.s3.bucket"
      }
  ]
}
```

启用访问日志

此示例为指定的负载均衡器启用访问日志。请注意，S3 存储桶必须与负载均衡器位于同一区域，并且必须附加一个授予对 Elastic Load Balancing 服务访问权限的策略。

命令:

```
aws elbv2 modify-load-balancer-attributes --load-balancer-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
```

```
balancer/50dc6c495c0c9188 --
attributes Key=access_logs.s3.enabled,Value=true Key=access_logs.s3.bucket,Value=my-
loadbalancer-logs Key=access_logs.s3.prefix,Value=myapp
```

输出：

```
{
  "Attributes": [
      {
          "Value": "true",
          "Key": "access_logs.s3.enabled"
      },
      {
          "Value": "my-load-balancer-logs",
          "Key": "access_logs.s3.bucket"
      },
      {
          "Value": "myapp",
          "Key": "access_logs.s3.prefix"
      },
      {
          "Value": "60",
          "Key": "idle_timeout.timeout_seconds"
      },
      {
          "Value": "false",
          "Key": "deletion_protection.enabled"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyLoadBalancerAttributes。

modify-rule

以下代码示例演示了如何使用 modify-rule。

Amazon CLI

修改规则

以下 modify-rule 示例更新指定规则的操作和条件。

```
aws elbv2 modify-rule \
  --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
  --conditions Field=path-pattern,Values='/images/*'
  --rule-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee
```

输出：

```
{
    "Rules": [
        {
            "Priority": "10",
            "Conditions": [
                {
                    "Field": "path-pattern",
                    "Values": [
                        "/images/*"
                    ]
                }
            ],
            "RuleArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-load-balancer/50dc6c495c0c9188/
f2f7dc8efc522ab2/9683b2d02a6cabee",
            "IsDefault": false,
            "Actions": [
                {
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
                    "Type": "forward"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyRule。

modify-target-group-attributes

以下代码示例演示了如何使用 modify-target-group-attributes。

Amazon CLI

修改取消注册延迟超时

此示例将指定目标组的取消注册延迟超时设置为指定值。

命令:

```
aws elbv2 modify-target-group-attributes --target-group-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/73e2d6bc24d8a067 --
attributes Key=deregistration_delay.timeout_seconds,Value=600
```

输出：

```
{
  "Attributes": [
      {
          "Value": "false",
          "Key": "stickiness.enabled"
      },
      {
          "Value": "600",
          "Key": "deregistration_delay.timeout_seconds"
      },
      {
          "Value": "lb_cookie",
          "Key": "stickiness.type"
      },
      {
          "Value": "86400",
          "Key": "stickiness.lb_cookie.duration_seconds"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTargetGroupAttributes。

modify-target-group

以下代码示例演示了如何使用 modify-target-group。

Amazon CLI

修改目标组的运行状况检查配置

以下 modify-target-group 示例更改用于评估指定目标组的目标运行状况的运行状况检查的配置。请注意，由于 CLI 解析逗号的方式不同，您必须用单引号（而不是双引号）将 --matcher 选项的范围引起来。

```
aws elbv2 modify-target-group \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-https-targets/2453ed029918f21f \
    --health-check-protocol HTTPS \
    --health-check-port 443 \
    --matcher HttpCode='200,299'
```

输出：

```
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-https-targets/2453ed029918f21f",
            "TargetGroupName": "my-https-targets",
            "Protocol": "HTTPS",
            "Port": 443,
            "VpcId": "vpc-3ac0fb5f",
            "HealthCheckProtocol": "HTTPS",
            "HealthCheckPort": "443",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 5,
            "UnhealthyThresholdCount": 2,
            "Matcher": {
                "HttpCode": "200,299"
            },
            "LoadBalancerArns": [
                "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/
app/my-load-balancer/50dc6c495c0c9188"
            ],
            "TargetType": "instance",
            "ProtocolVersion": "HTTP1",
            "IpAddressType": "ipv4"
```

```
        }
    ]
}
```

有关更多信息，请参阅《应用程序负载均衡器指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyTargetGroup。

register-targets

以下代码示例演示了如何使用 register-targets。

Amazon CLI

示例 1：按实例 ID 将目标注册到目标组

以下 register-targets 示例将指定的实例注册到目标组。该目标组必须有一个目标类型 instance。

```
aws elbv2 register-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
    --targets Id=i-1234567890abcdef0 Id=i-0abcdef1234567890
```

示例 2：使用端口覆盖将目标注册到目标组

以下 register-targets 示例使用多个端口将指定的实例注册到目标组。这让您能够在与目标组中的目标相同的实例上注册容器。

```
aws elbv2 register-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-internal-targets/3bb63f11dfb0faf9 \
    --targets Id=i-0598c7d356eba48d7,Port=80 Id=i-0598c7d356eba48d7,Port=766
```

示例 3：按 IP 地址将目标注册到目标组

以下 register-targets 示例将指定的 IP 地址注册到目标组。该目标组必须有一个目标类型 ip。

```
aws elbv2 register-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-tcp-ip-targets/8518e899d173178f \
```

```
    --targets Id=10.0.1.15 Id=10.0.1.23
```

示例 4：将 Lambda 函数注册为目标

以下 `register-targets` 示例将指定的 IP 地址注册到目标组。该目标组必须有一个目标类型 `lambda`。您必须向 Elastic Load Balancing 授予调用 Lambda 函数的权限。

```
aws elbv2 register-targets \
    --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-tcp-ip-targets/8518e899d173178f \
    --targets Id=arn:aws:lambda:us-west-2:123456789012:function:my-function
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTargets。

remove-listener-certificates

以下代码示例演示了如何使用 `remove-listener-certificates`。

Amazon CLI

从安全侦听器中删除证书

此示例从指定的安全侦听器中删除指定的证书。

命令:

```
aws elbv2 remove-listener-certificates --listener-
arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/
app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 --
certificates CertificateArn=arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-
f4a3-4072-80be-05b9ba72f705
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveListenerCertificates。

remove-tags

以下代码示例演示了如何使用 `remove-tags`。

Amazon CLI

从负载均衡器中删除标签

以下 remove-tags 示例从指定的负载均衡器中删除 project 和 department 标签。

```
aws elbv2 remove-tags \
    --resource-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
    --tag-keys project department
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTags。

set-ip-address-type

以下代码示例演示了如何使用 set-ip-address-type。

Amazon CLI

设置负载均衡器的地址类型

此示例将指定负载均衡器的地址类型设置为 dualstack。负载均衡器子网必须具有关联的 IPv6 CIDR 块。

命令:

```
aws elbv2 set-ip-address-type --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --ip-address-
type dualstack
```

输出：

```
{
    "IpAddressType": "dualstack"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetIpAddressType。

set-rule-priorities

以下代码示例演示了如何使用 set-rule-priorities。

Amazon CLI

设置规则优先级

此示例设置指定规则的优先级。

命令:

```
aws elbv2 set-rule-priorities --rule-
priorities RuleArn=arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-
rule/app/my-load-balancer/50dc6c495c0c9188/
f2f7dc8efc522ab2/1291d13826f405c3,Priority=5
```

输出：

```
{
  "Rules": [
      {
          "Priority": "5",
          "Conditions": [
              {
                  "Field": "path-pattern",
                  "Values": [
                      "/img/*"
                  ]
              }
          ],
          "RuleArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-
rule/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/1291d13826f405c3",
          "IsDefault": false,
          "Actions": [
              {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
                  "Type": "forward"
              }
          ]
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetRulePriorities。

set-security-groups

以下代码示例演示了如何使用 set-security-groups。

Amazon CLI

将安全组与负载均衡器相关联

此示例将指定的安全组与指定的负载均衡器相关联。

命令:

```
aws elbv2 set-security-groups --load-balancer-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --security-groups sg-5943793c
```

输出：

```
{
  "SecurityGroupIds": [
      "sg-5943793c"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetSecurityGroups。

set-subnets

以下代码示例演示了如何使用 set-subnets。

Amazon CLI

为负载均衡器启用可用区

此示例为用于指定负载均衡器的指定子网启用可用区。

命令:

```
aws elbv2 set-subnets --load-balancer-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --subnets subnet-8360a9e7 subnet-b7d581c0
```

输出：

```
{
  "AvailabilityZones": [
      {
```

```
            "SubnetId": "subnet-8360a9e7",
            "ZoneName": "us-west-2a"
        },
        {
            "SubnetId": "subnet-b7d581c0",
            "ZoneName": "us-west-2b"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetSubnets。

使用 Amazon CLI 的 ElastiCache 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 ElastiCache 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-resource`

以下代码示例演示了如何使用 `add-tags-to-resource`。

Amazon CLI

为资源添加标签

以下 `add-tags-to-resource` 示例向集群或快照资源添加最多 10 个标签（键值对）。

```
aws elasticache add-tags-to-resource \
    --resource-name "arn:aws:elasticache:us-east-1:1234567890:cluster:my-mem-
cluster" \
```

```
    --tags '{"20150202":15, "ElastiCache":"Service"}'
```

输出：

```
{
    "TagList": [
        {
            "Value": "20150202",
            "Key": "APIVersion"
        },
        {
            "Value": "ElastiCache",
            "Key": "Service"
        }
    ]
}
```

有关更多信息，请参阅《Elasticache 用户指南》中的使用成本分配标签监控成本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToResource。

authorize-cache-security-group-ingress

以下代码示例演示了如何使用 authorize-cache-security-group-ingress。

Amazon CLI

为入口授权缓存安全组

以下 authorize-cache-security-group-ingress 示例允许网络进入缓存安全组。

```
aws elasticache authorize-cache-security-group-ingress \
     --cache-security-group-name  "my-sec-grp" \
     --ec2-security-group-name "my-ec2-sec-grp" \
     --ec2-security-group-owner-id "1234567890"
```

此命令不生成任何输出。

有关更多信息，请参阅《ElastiCache 用户指南》中的 Amazon ElastiCache 中的自助更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeCacheSecurityGroupIngress。

batch-apply-update-action

以下代码示例演示了如何使用 batch-apply-update-action。

Amazon CLI

应用服务更新

以下 batch-apply-update-action 示例将服务更新应用于 Redis 集群。

```
aws elasticache batch-apply-update-action \
    --service-update-name elc-xxxxx406-xxx \
    --replication-group-ids test-cluster
```

输出：

```
{
    "ProcessedUpdateActions": [
        {
            "ReplicationGroupId": "pat-cluster",
            "ServiceUpdateName": "elc-xxxxx406-xxx",
            "UpdateActionStatus": "waiting-to-start"
        }
    ],
    "UnprocessedUpdateActions": []
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 Amazon ElastiCache 中的自助更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchApplyUpdateAction。

batch-stop-update-action

以下代码示例演示了如何使用 batch-stop-update-action。

Amazon CLI

停止服务更新

以下 batch-stop-update-action 示例将服务更新应用于 Redis 集群。

```
aws elasticache batch-stop-update-action \
```

```
    --service-update-name elc-xxxxx406-xxx \
    --replication-group-ids test-cluster
```

输出：

```
{
    "ProcessedUpdateActions": [
        {
            "ReplicationGroupId": "pat-cluster",
            "ServiceUpdateName": "elc-xxxxx406-xxx",
            "UpdateActionStatus": "stopping"
        }
    ],
    "UnprocessedUpdateActions": []
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 Amazon ElastiCache 中的自助更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchStopUpdateAction。

copy-snapshot

以下代码示例演示了如何使用 copy-snapshot。

Amazon CLI

复制快照

以下 copy-snapshot 示例复制现有快照。

```
aws elasticache copy-snapshot \
    --source-snapshot-name "my-snapshot" \
    --target-snapshot-name "my-snapshot-copy"
```

输出：

```
{
    "Snapshot":{
        "Engine": "redis",
        "CacheParameterGroupName": "default.redis3.2",
        "VpcId": "vpc-3820329f3",
```

```
        "CacheClusterId": "my-redis4",
        "SnapshotRetentionLimit": 7,
        "NumCacheNodes": 1,
        "SnapshotName": "my-snapshot-copy",
        "CacheClusterCreateTime": "2016-12-21T22:24:04.955Z",
        "AutoMinorVersionUpgrade": true,
        "PreferredAvailabilityZone": "us-east-1c",
        "SnapshotStatus": "creating",
        "SnapshotSource": "manual",
        "SnapshotWindow": "07:00-08:00",
        "EngineVersion": "3.2.4",
        "NodeSnapshots": [
            {
                "CacheSize": "3 MB",
                "SnapshotCreateTime": "2016-12-28T07:00:52Z",
                "CacheNodeId": "0001",
                "CacheNodeCreateTime": "2016-12-21T22:24:04.955Z"
            }
        ],
        "CacheSubnetGroupName": "default",
        "Port": 6379,
        "PreferredMaintenanceWindow": "tue:09:30-tue:10:30",
        "CacheNodeType": "cache.m3.large"
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的导出备份。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopySnapshot。

create-cache-cluster

以下代码示例演示了如何使用 create-cache-cluster。

Amazon CLI

创建缓存集群

以下 create-cache-cluster 示例使用 Redis 引擎创建缓存集群。

```
aws elasticache create-cache-cluster \
    --cache-cluster-id "cluster-test" \
    --engine redis \
```

```
    --cache-node-type cache.m5.large \
    --num-cache-nodes 1
```

输出：

```
{
    "CacheCluster": {
        "CacheClusterId": "cluster-test",
        "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
        "CacheNodeType": "cache.m5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "CacheClusterStatus": "creating",
        "NumCacheNodes": 1,
        "PreferredMaintenanceWindow": "sat:13:00-sat:14:00",
        "PendingModifiedValues": {},
        "CacheSecurityGroups": [],
        "CacheParameterGroup": {
            "CacheParameterGroupName": "default.redis5.0",
            "ParameterApplyStatus": "in-sync",
            "CacheNodeIdsToReboot": []
        },
        "CacheSubnetGroupName": "default",
        "AutoMinorVersionUpgrade": true,
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "06:30-07:30",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的创建集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCacheCluster。

create-cache-parameter-group

以下代码示例演示了如何使用 create-cache-parameter-group。

Amazon CLI

创建缓存参数组

以下 create-cache-parameter-group 示例创建一个新的 Amazon ElastiCache 缓存参数组。

```
aws elasticache create-cache-parameter-group \
    --cache-parameter-group-family "redis5.0" \
    --cache-parameter-group-name "mygroup" \
    --description "mygroup"
```

输出：

```
{
    "CacheParameterGroup": {
        "CacheParameterGroupName": "mygroup",
        "CacheParameterGroupFamily": "redis5.0",
        "Description": "my group"
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的创建参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCacheParameterGroup。

create-cache-subnet-group

以下代码示例演示了如何使用 create-cache-subnet-group。

Amazon CLI

创建缓存子网组

以下 create-cache-subnet-group 示例创建一个新的缓存子网组。

```
aws elasticache create-cache-subnet-group \
    --cache-subnet-group-name "mygroup" \
    --cache-subnet-group-description "my subnet group" \
    --subnet-ids "subnet-xxxxec4f"
```

输出：

```
{
    "CacheSubnetGroup": {
        "CacheSubnetGroupName": "mygroup",
        "CacheSubnetGroupDescription": "my subnet group",
```

```
        "VpcId": "vpc-a3e97cdb",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-xxxxec4f",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2d"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的创建缓存子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCacheSubnetGroup。

create-global-replication-group

以下代码示例演示了如何使用 create-global-replication-group。

Amazon CLI

创建全局复制组

以下 create-global-replication-group 示例创建一个新的全局复制组。

```
aws elasticache create-global-replication-group \
    --global-replication-group-id-suffix my-global-replication-group \
    --primary-replication-group-id my-primary-cluster
```

输出：

```
{
    "GlobalReplicationGroup": {
        "GlobalReplicationGroupId": "sgaui-my-global-replication-group",
        "GlobalReplicationGroupDescription": " ",
        "Status": "creating",
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.6",
        "Members": [
            {
                "ReplicationGroupId": "my-primary-cluster",
```

```
                "ReplicationGroupRegion": "us-west-2",
                "Role": "PRIMARY",
                "AutomaticFailover": "enabled",
                "Status": "associating"
            }
        ],
        "ClusterEnabled": true,
        "GlobalNodeGroups": [
            {
                "GlobalNodeGroupId": "sgaui-my-global-replication-group-0001",
                "Slots": "0-16383"
            }
        ],
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGlobalReplicationGroup。

create-replication-group

以下代码示例演示了如何使用 create-replication-group。

Amazon CLI

创建复制组

以下 create-replication-group 示例创建 Redis（已禁用集群模式）或 Redis（已启用集群模式）复制组。此操作仅对 Redis 有效。

```
aws elasticache create-replication-group \
    --replication-group-id "mygroup" \
    --replication-group-description "my group" \
    --engine "redis" \
    --cache-node-type "cache.m5.large"
```

输出：

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "mygroup",
        "Description": "my group",
        "Status": "creating",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "mygroup-001"
        ],
        "AutomaticFailover": "disabled",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "06:00-07:00",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.m5.large",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的创建 Redis 复制组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplicationGroup。

`create-snapshot`

以下代码示例演示了如何使用 `create-snapshot`。

Amazon CLI

创建快照

以下 `create-snapshot` 示例使用 Redis 引擎创建快照。

```
aws elasticache create-snapshot \
    --snapshot-name mysnapshot \
    --cache-cluster-id cluster-test
```

输出：

```
{
    "Snapshot": {
        "SnapshotName": "mysnapshot",
```

```
        "CacheClusterId": "cluster-test",
        "SnapshotStatus": "creating",
        "SnapshotSource": "manual",
        "CacheNodeType": "cache.m5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "NumCacheNodes": 1,
        "PreferredAvailabilityZone": "us-west-2b",
        "CacheClusterCreateTime": "2020-03-19T03:12:01.483Z",
        "PreferredMaintenanceWindow": "sat:13:00-sat:14:00",
        "Port": 6379,
        "CacheParameterGroupName": "default.redis5.0",
        "CacheSubnetGroupName": "default",
        "VpcId": "vpc-a3e97cdb",
        "AutoMinorVersionUpgrade": true,
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "06:30-07:30",
        "NodeSnapshots": [
            {
                "CacheNodeId": "0001",
                "CacheSize": "",
                "CacheNodeCreateTime": "2020-03-19T03:12:01.483Z"
            }
        ]
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 ElastiCache for Redis 备份和还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshot。

create-user-group

以下代码示例演示了如何使用 create-user-group。

Amazon CLI

创建用户组

以下 create-user-group 示例创建一个新的用户组。

```
aws elasticache create-user-group \
    --user-group-id myusergroup \
    --engine redis \
```

```
    --user-ids default
```

输出：

```
{
    "UserGroupId": "myusergroup",
    "Status": "creating",
    "Engine": "redis",
    "UserIds": [
        "default"
    ],
    "ReplicationGroups": [],
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUserGroup。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

创建用户

以下 create-user 示例创建一个新的用户。

```
aws elasticache create-user \
    --user-id user1 \
    --user-name myUser \
    --passwords mYnuUzrpAxXw2rdzx \
    --engine redis \
    --access-string "on ~app::* -@all +@read"
```

输出：

```
{
    "UserId": "user2",
    "UserName": "myUser",
```

```
    "Status": "active",
    "Engine": "redis",
    "AccessString": "on ~app::* -@all +@read +@hash +@bitmap +@geo -setbit -bitfield
 -hset -hsetnx -hmset -hincrby -hincrbyfloat -hdel -bitop -geoadd -georadius -
georadiusbymember",
    "UserGroupIds": [],
    "Authentication": {
        "Type": "password",
        "PasswordCount": 1
    },
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

decrease-node-groups-in-global-replication-group

以下代码示例演示了如何使用 decrease-node-groups-in-global-replication-group。

Amazon CLI

减少全局复制组中节点组的数量

以下 decrease-node-groups-in-global-replication-group 使用 Redis 引擎减少节点组计数。

```
aws elasticache decrease-node-groups-in-global-replication-group \
    --global-replication-group-id sgaui-test \
    --node-group-count 1 \
    --apply-immediately \
    --global-node-groups-to-retain sgaui-test-0003
```

输出：

```
{
    "GlobalReplicationGroup":
    {
        "GlobalReplicationGroupId": "sgaui-test",
        "GlobalReplicationGroupDescription": "test",
        "Status": "modifying",
```

```
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.6",
        "Members": [
            {
                "ReplicationGroupId": "test-2",
                "ReplicationGroupRegion": "us-east-1",
                "Role": "SECONDARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            },
            {
                "ReplicationGroupId": "test-1",
                "ReplicationGroupRegion": "us-west-2",
                "Role": "PRIMARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            }
        ],
        "ClusterEnabled": true,
        "GlobalNodeGroups": [
            {
                "GlobalNodeGroupId": "sgaui-test-0001",
                "Slots": "0-449,1816-5461"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-0002",
                "Slots": "6827-10922"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-0003",
                "Slots": "10923-14052,15418-16383"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-0004",
                "Slots": "450-1815,5462-6826,14053-15417"
            }
        ],
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecreaseNodeGroupsInGlobalReplicationGroup。

decrease-replica-count

以下代码示例演示了如何使用 decrease-replica-count。

Amazon CLI

减少副本计数

以下 decrease-replica-count 示例动态减少 Redis（已禁用集群模式）复制组中的副本数量或 Redis（已启用集群模式）复制组的一个或多个节点组（分片）中的副本节点数量。此操作在无需集群停机的情况下执行。

```
aws elasticache decrease-replica-count \
    --replication-group-id my-cluster \
    --apply-immediately  \
    --new-replica-count 2
```

输出：

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "my-cluster",
        "Description": " ",
        "Status": "modifying",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "myrepliace",
            "my-cluster-001",
            "my-cluster-002",
            "my-cluster-003"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
                "Status": "modifying",
                "PrimaryEndpoint": {
                    "Address": "my-cluster.xxxxx.ng.0001.usw2.cache.amazonaws.com",
```

```
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "my-cluster-
ro.xxxxx.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "myrepliace",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address":
 "myrepliace.xxxxx.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "my-cluster-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "my-
cluster-001.xxxxx.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "my-cluster-002",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "my-
cluster-002.xxxxx.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "my-cluster-003",
                        "CacheNodeId": "0001",
```

```
                        "ReadEndpoint": {
                            "Address": "my-
cluster-003.xxxxx.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "replica"
                    }
                ]
            }
        ],
        "AutomaticFailover": "disabled",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "07:30-08:30",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.xlarge",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的更改副本数量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecreaseReplicaCount。

`delete-cache-cluster`

以下代码示例演示了如何使用 `delete-cache-cluster`。

Amazon CLI

删除缓存集群

以下 `delete-cache-cluster` 示例删除指定的之前预置的集群。该命令将删除所有关联的缓存节点、节点端点和集群本身。当您收到此操作的成功响应时，Amazon ElastiCache 会立即开始删除该集群；您无法取消或恢复此操作。

此操作对以下情况无效：

Redis（已启用集群模式）集群 作为复制组的上一个只读副本的集群 启用了多可用区模式的节点组（分片） 来自 Redis（已启用集群模式）复制组的集群 未处于可用状态的集群

```
aws elasticache delete-cache-cluster \
```

```
    --cache-cluster-id "my-cluster-002"
```

输出：

```
{
    "CacheCluster": {
        "CacheClusterId": "my-cluster-002",
        "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
        "CacheNodeType": "cache.r5.xlarge",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "CacheClusterStatus": "deleting",
        "NumCacheNodes": 1,
        "PreferredAvailabilityZone": "us-west-2a",
        "CacheClusterCreateTime": "2019-11-26T03:35:04.546Z",
        "PreferredMaintenanceWindow": "mon:04:05-mon:05:05",
        "PendingModifiedValues": {},
        "NotificationConfiguration": {
            "TopicArn": "arn:aws:sns:us-west-x:xxxxxxx4152:My_Topic",
            "TopicStatus": "active"
        },
        "CacheSecurityGroups": [],
        "CacheParameterGroup": {
            "CacheParameterGroupName": "mygroup",
            "ParameterApplyStatus": "in-sync",
            "CacheNodeIdsToReboot": []
        },
        "CacheSubnetGroupName": "kxkxk",
        "AutoMinorVersionUpgrade": true,
        "SecurityGroups": [
            {
                "SecurityGroupId": "sg-xxxxxxxxxx9836",
                "Status": "active"
            },
            {
                "SecurityGroupId": "sg-xxxxxxxxxxxx7b",
                "Status": "active"
            }
        ],
        "ReplicationGroupId": "my-cluster",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "07:30-08:30",
```

```
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的删除集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCacheCluster。

delete-cache-parameter-group

以下代码示例演示了如何使用 delete-cache-parameter-group。

Amazon CLI

删除缓存参数组

以下 delete-cache-parameter-group 示例删除指定的缓存参数组。如果缓存参数组与任何缓存集群关联，则无法删除该缓存参数组。

```
aws elasticache delete-cache-parameter-group \
    --cache-parameter-group-name myparamgroup
```

此命令不生成任何输出。

有关更多信息，请参阅《ElastiCache 用户指南》中的删除参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCacheParameterGroup。

delete-cache-subnet-group

以下代码示例演示了如何使用 delete-cache-subnet-group。

Amazon CLI

删除缓存子网组

以下 delete-cache-subnet-group 示例删除指定的缓存子网组。如果缓存子网组与任何集群关联，则无法删除该缓存子网组。

```
aws elasticache delete-cache-subnet-group \
    --cache-subnet-group-name "mygroup"
```

此命令不生成任何输出。

有关更多信息，请参阅《ElastiCache 用户指南》中的删除子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCacheSubnetGroup。

delete-global-replication-group

以下代码示例演示了如何使用 delete-global-replication-group。

Amazon CLI

删除全局复制组

以下 delete-global-replication-group 示例删除新的全局复制组。

```
aws elasticache delete-global-replication-group \
    --global-replication-group-id my-global-replication-group \
    --retain-primary-replication-group
```

输出：

```
{
    "GlobalReplicationGroup": {
        "GlobalReplicationGroupId": "sgaui-my-grg",
        "GlobalReplicationGroupDescription": "my-grg",
        "Status": "deleting",
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.6",
        "Members": [
            {
                "ReplicationGroupId": "my-cluster-grg",
                "ReplicationGroupRegion": "us-west-2",
                "Role": "PRIMARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            }
        ],
        "ClusterEnabled": false,
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
```

```
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGlobalReplicationGroup。

delete-replication-group

以下代码示例演示了如何使用 delete-replication-group。

Amazon CLI

删除复制组

以下 delete-replication-group 示例删除现有复制组。默认情况下，此操作会删除整个复制组，包括单个主副本/多个主副本和所有只读副本。如果复制组只有一个主副本，则您可以选择仅删除只读副本，同时通过设置 RetainPrimaryCluster=true 来保留主副本。

当您收到此操作的成功响应时，Amazon ElastiCache 会立即开始删除选定的资源；您无法取消或恢复此操作。仅对 Redis 有效。

```
aws elasticache delete-replication-group \
    --replication-group-id "mygroup"
```

输出：

```
{
     "ReplicationGroup": {
         "ReplicationGroupId": "mygroup",
         "Description": "my group",
         "Status": "deleting",
         "PendingModifiedValues": {},
         "AutomaticFailover": "disabled",
         "SnapshotRetentionLimit": 0,
         "SnapshotWindow": "06:00-07:00",
         "TransitEncryptionEnabled": false,
         "AtRestEncryptionEnabled": false
     }
 }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReplicationGroup。

delete-snapshot

以下代码示例演示了如何使用 delete-snapshot。

Amazon CLI

删除快照

以下 delete-snapshot 示例已使用 Redis 引擎删除快照。

```
aws elasticache delete-snapshot \
    --snapshot-name mysnapshot
```

输出：

```
{
    "Snapshot": {
        "SnapshotName": "my-cluster-snapshot",
        "ReplicationGroupId": "mycluster",
        "ReplicationGroupDescription": "mycluster",
        "SnapshotStatus": "deleting",
        "SnapshotSource": "manual",
        "CacheNodeType": "cache.r5.xlarge",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "PreferredMaintenanceWindow": "thu:12:00-thu:13:00",
        "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxxxx152:My_Topic",
        "Port": 6379,
        "CacheParameterGroupName": "default.redis5.0.cluster.on",
        "CacheSubnetGroupName": "default",
        "VpcId": "vpc-a3e97cdb",
        "AutoMinorVersionUpgrade": true,
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "13:00-14:00",
        "NumNodeGroups": 4,
        "AutomaticFailover": "enabled",
        "NodeSnapshots": [
            {
                "CacheClusterId": "mycluster-0002-003",
                "NodeGroupId": "0002",
                "CacheNodeId": "0001",
```

```
                "CacheSize": "6 MB",
                "CacheNodeCreateTime": "2020-06-18T00:05:44.719000+00:00",
                "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
            },
            {
                "CacheClusterId": "mycluster-0003-003",
                "NodeGroupId": "0003",
                "CacheNodeId": "0001",
                "CacheSize": "6 MB",
                "CacheNodeCreateTime": "2019-12-05T19:13:15.912000+00:00",
                "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
            },
            {
                "CacheClusterId": "mycluster-0004-002",
                "NodeGroupId": "0004",
                "CacheNodeId": "0001",
                "CacheSize": "6 MB",
                "CacheNodeCreateTime": "2019-12-09T19:44:34.324000+00:00",
                "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
            },
            {
                "CacheClusterId": "mycluster-0005-003",
                "NodeGroupId": "0005",
                "CacheNodeId": "0001",
                "CacheSize": "6 MB",
                "CacheNodeCreateTime": "2020-06-18T00:05:44.775000+00:00",
                "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
            }
        ]
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 ElastiCache for Redis 备份和还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSnapshot。

`delete-user-group`

以下代码示例演示了如何使用 `delete-user-group`。

Amazon CLI

删除用户组

以下 delete-user-group 示例删除用户组。

```
aws elasticache delete-user-group \
    --user-group-id myusergroup
```

输出：

```
{
    "UserGroupId": "myusergroup",
    "Status": "deleting",
    "Engine": "redis",
    "UserIds": [
        "default"
    ],
    "ReplicationGroups": [],
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserGroup。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

以下 delete-user 示例删除用户。

```
aws elasticache delete-user \
    --user-id user2
```

输出：

```
{
    "UserId": "user1",
    "UserName": "myUser",
    "Status": "deleting",
```

```
    "Engine": "redis",
    "AccessString": "on ~* +@all",
    "UserGroupIds": [
        "myusergroup"
    ],
    "Authentication": {
        "Type": "password",
        "PasswordCount": 1
    },
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user1"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-cache-clusters

以下代码示例演示了如何使用 describe-cache-clusters。

Amazon CLI

描述缓存集群

以下 describe-cache-clusters 示例描述了一个缓存集群。

```
aws elasticache describe-cache-clusters
```

输出：

```
{
    "CacheClusters": [
    {
            "CacheClusterId": "my-cluster-003",
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/
elasticache/home#client-download:",
            "CacheNodeType": "cache.r5.large",
            "Engine": "redis",
            "EngineVersion": "5.0.5",
            "CacheClusterStatus": "available",
            "NumCacheNodes": 1,
            "PreferredAvailabilityZone": "us-west-2a",
```

```
            "CacheClusterCreateTime": "2019-11-26T01:22:52.396Z",
            "PreferredMaintenanceWindow": "mon:17:30-mon:18:30",
            "PendingModifiedValues": {},
            "NotificationConfiguration": {
                "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxxx152:My_Topic",
                "TopicStatus": "active"
            },
            "CacheSecurityGroups": [],
            "CacheParameterGroup": {
                "CacheParameterGroupName": "default.redis5.0",
                "ParameterApplyStatus": "in-sync",
                "CacheNodeIdsToReboot": []
            },
            "CacheSubnetGroupName": "kxkxk",
            "AutoMinorVersionUpgrade": true,
            "SecurityGroups": [
                {
                    "SecurityGroupId": "sg-xxxxxd7b",
                    "Status": "active"
                }
            ],
            "ReplicationGroupId": "my-cluster",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "06:30-07:30",
            "AuthTokenEnabled": false,
            "TransitEncryptionEnabled": false,
            "AtRestEncryptionEnabled": false,
            "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxxx152:cluster:my-cache-
cluster",
            "ReplicationGroupLogDeliveryEnabled": false,
            "LogDeliveryConfigurations": [
                {
                    "LogType": "slow-log",
                    "DestinationType": "cloudwatch-logs",
                    "DestinationDetails": {
                        "CloudWatchLogsDetails": {
                            "LogGroup": "test-log"
                        }
                    },
                    "LogFormat": "text",
                    "Status": "active"
                }
            ]
        }
```

```
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的管理集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCacheClusters。

describe-cache-engine-versions

以下代码示例演示了如何使用 describe-cache-engine-versions。

Amazon CLI

描述缓存引擎版本

以下 describe-cache-engine-versions 示例返回可用缓存引擎及其版本的列表。

```
aws elasticache describe-cache-engine-versions \
    --engine "Redis"
```

输出：

```
{
    "CacheEngineVersions": [
        {
            "Engine": "redis",
            "EngineVersion": "2.6.13",
            "CacheParameterGroupFamily": "redis2.6",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.6.13"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.19",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.8.19"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.21",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
```

```
            "CacheEngineVersionDescription": "redis version 2.8.21"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.22",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.8.22"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.23",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.8.23"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.24",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.8.24"
        },
        {
            "Engine": "redis",
            "EngineVersion": "2.8.6",
            "CacheParameterGroupFamily": "redis2.8",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 2.8.6"
        },
        {
            "Engine": "redis",
            "EngineVersion": "3.2.10",
            "CacheParameterGroupFamily": "redis3.2",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 3.2.10"
        },
        {
            "Engine": "redis",
            "EngineVersion": "3.2.4",
            "CacheParameterGroupFamily": "redis3.2",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 3.2.4"
        },
```

```
        {
            "Engine": "redis",
            "EngineVersion": "3.2.6",
            "CacheParameterGroupFamily": "redis3.2",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 3.2.6"
        },
        {
            "Engine": "redis",
            "EngineVersion": "4.0.10",
            "CacheParameterGroupFamily": "redis4.0",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 4.0.10"
        },
        {
            "Engine": "redis",
            "EngineVersion": "5.0.0",
            "CacheParameterGroupFamily": "redis5.0",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 5.0.0"
        },
        {
            "Engine": "redis",
            "EngineVersion": "5.0.3",
            "CacheParameterGroupFamily": "redis5.0",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 5.0.3"
        },
        {
            "Engine": "redis",
            "EngineVersion": "5.0.4",
            "CacheParameterGroupFamily": "redis5.0",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 5.0.4"
        },
        {
            "Engine": "redis",
            "EngineVersion": "5.0.5",
            "CacheParameterGroupFamily": "redis5.0",
            "CacheEngineDescription": "Redis",
            "CacheEngineVersionDescription": "redis version 5.0.5"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCacheEngineVersions。

describe-cache-parameter-groups

以下代码示例演示了如何使用 describe-cache-parameter-groups。

Amazon CLI

描述缓存参数组

以下 describe-cache-parameter-groups 示例返回缓存参数组描述的列表。

```
aws elasticache describe-cache-parameter-groups \
    --cache-parameter-group-name "mygroup"
```

输出：

```
{
    "CacheParameterGroups": [
        {
            "CacheParameterGroupName": "mygroup",
            "CacheParameterGroupFamily": "redis5.0",
            "Description": " "
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用参数组配置引擎参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCacheParameterGroups。

describe-cache-parameters

以下代码示例演示了如何使用 describe-cache-parameters。

Amazon CLI

描述缓存参数

以下“describe-cache-parameters”示例返回指定缓存参数组的详细参数列表。

```
aws elasticache describe-cache-parameters \
    --cache-parameter-group-name "myparamgroup"
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "activedefrag",
            "ParameterValue": "yes",
            "Description": "Enabled active memory defragmentation",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-cycle-max",
            "ParameterValue": "75",
            "Description": "Maximal effort for defrag in CPU percentage",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-75",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-cycle-min",
            "ParameterValue": "5",
            "Description": "Minimal effort for defrag in CPU percentage",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-75",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-ignore-bytes",
            "ParameterValue": "104857600",
```

```
            "Description": "Minimum amount of fragmentation waste to start active
defrag",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1048576-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-max-scan-fields",
            "ParameterValue": "1000",
            "Description": "Maximum number of set/hash/zset/list fields that will be
processed from the main dictionary scan",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-1000000",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-threshold-lower",
            "ParameterValue": "10",
            "Description": "Minimum percentage of fragmentation to start active
defrag",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-100",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "active-defrag-threshold-upper",
            "ParameterValue": "100",
            "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-100",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "activerehashing",
            "ParameterValue": "yes",
            "Description": "Apply rehashing or not.",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "requires-reboot"
        },
        {
            "ParameterName": "appendfsync",
            "ParameterValue": "everysec",
            "Description": "fsync policy for AOF persistence",
            "Source": "system",
            "DataType": "string",
            "AllowedValues": "always,everysec,no",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "appendonly",
            "ParameterValue": "no",
            "Description": "Enable Redis persistence.",
            "Source": "system",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-output-buffer-limit-normal-hard-limit",
            "ParameterValue": "0",
            "Description": "Normal client output buffer hard limit in bytes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "client-output-buffer-limit-normal-soft-limit",
            "ParameterValue": "0",
            "Description": "Normal client output buffer soft limit in bytes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-output-buffer-limit-normal-soft-seconds",
            "ParameterValue": "0",
            "Description": "Normal client output buffer soft limit in seconds.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-output-buffer-limit-pubsub-hard-limit",
            "ParameterValue": "33554432",
            "Description": "Pubsub client output buffer hard limit in bytes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-output-buffer-limit-pubsub-soft-limit",
            "ParameterValue": "8388608",
            "Description": "Pubsub client output buffer soft limit in bytes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "client-output-buffer-limit-pubsub-soft-seconds",
            "ParameterValue": "60",
            "Description": "Pubsub client output buffer soft limit in seconds.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-output-buffer-limit-replica-soft-seconds",
            "ParameterValue": "60",
            "Description": "Replica client output buffer soft limit in seconds.",
            "Source": "system",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "client-query-buffer-limit",
            "ParameterValue": "1073741824",
            "Description": "Max size of a single client query buffer",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1048576-1073741824",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "close-on-replica-write",
            "ParameterValue": "yes",
            "Description": "If enabled, clients who attempt to write to a read-only
replica will be disconnected. Applicable to 2.8.23 and higher.",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
```

```
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "cluster-enabled",
            "ParameterValue": "no",
            "Description": "Enable cluster mode",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "requires-reboot"
        },
        {
            "ParameterName": "cluster-require-full-coverage",
            "ParameterValue": "no",
            "Description": "Whether cluster becomes unavailable if one or more slots
are not covered",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "databases",
            "ParameterValue": "16",
            "Description": "Set the number of databases.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-1200000",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "requires-reboot"
        },
        {
            "ParameterName": "hash-max-ziplist-entries",
            "ParameterValue": "512",
            "Description": "The maximum number of hash entries in order for the
dataset to be compressed.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
```

```
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "hash-max-ziplist-value",
            "ParameterValue": "64",
            "Description": "The threshold of biggest hash entries in order for the
dataset to be compressed.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "hll-sparse-max-bytes",
            "ParameterValue": "3000",
            "Description": "HyperLogLog sparse representation bytes limit",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-16000",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lazyfree-lazy-eviction",
            "ParameterValue": "no",
            "Description": "Perform an asynchronous delete on evictions",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lazyfree-lazy-expire",
            "ParameterValue": "no",
            "Description": "Perform an asynchronous delete on expired keys",
            "Source": "user",
            "DataType": "string",
```

```
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lazyfree-lazy-server-del",
            "ParameterValue": "no",
            "Description": "Perform an asynchronous delete on key updates",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lfu-decay-time",
            "ParameterValue": "1",
            "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policy",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lfu-log-factor",
            "ParameterValue": "10",
            "Description": "The log factor for incrementing key counter for LFU
eviction policy",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "list-compress-depth",
            "ParameterValue": "0",
```

```
            "Description": "Number of quicklist ziplist nodes from each side of
the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "list-max-ziplist-size",
            "ParameterValue": "-2",
            "Description": "The number of entries allowed per internal list node can
be specified as a fixed maximum size or a maximum number of elements",
            "Source": "system",
            "DataType": "integer",
            "AllowedValues": "-5,-4,-3,-2,-1,1-",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lua-replicate-commands",
            "ParameterValue": "yes",
            "Description": "Always enable Lua effect replication or not",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "lua-time-limit",
            "ParameterValue": "5000",
            "Description": "Max execution time of a Lua script in milliseconds. 0
for unlimited execution without warnings.",
            "Source": "system",
            "DataType": "integer",
            "AllowedValues": "5000",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "maxclients",
            "ParameterValue": "65000",
            "Description": "The maximum number of Redis clients.",
            "Source": "system",
            "DataType": "integer",
            "AllowedValues": "1-65000",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "requires-reboot"
        },
        {
            "ParameterName": "maxmemory-policy",
            "ParameterValue": "volatile-lru",
            "Description": "Max memory policy.",
            "Source": "user",
            "DataType": "string",
            "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-
lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "maxmemory-samples",
            "ParameterValue": "3",
            "Description": "Max memory samples.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "min-replicas-max-lag",
            "ParameterValue": "10",
            "Description": "The maximum amount of replica lag in seconds beyond
 which the master would stop taking writes. A value of 0 means the master always
 takes writes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
```

```
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "min-replicas-to-write",
            "ParameterValue": "0",
            "Description": "The minimum number of replicas that must be present with
lag no greater than min-replicas-max-lag for master to take writes. Setting this to
0 means the master always takes writes.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "notify-keyspace-events",
            "Description": "The keyspace events for Redis to notify Pub/Sub clients
about. By default all notifications are disabled",
            "Source": "user",
            "DataType": "string",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "proto-max-bulk-len",
            "ParameterValue": "536870912",
            "Description": "Max size of a single element request",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "1048576-536870912",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "rename-commands",
            "ParameterValue": "",
            "Description": "Redis commands that can be dynamically renamed by the
customer",
            "Source": "user",
```

```
            "DataType": "string",
            "AllowedValues":
"APPEND,BITCOUNT,BITFIELD,BITOP,BITPOS,BLPOP,BRPOP,BRPOPLPUSH,BZPOPMIN,BZPOPMAX,CLIENT,COMM
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.3",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "repl-backlog-size",
            "ParameterValue": "1048576",
            "Description": "The replication backlog size in bytes for PSYNC. This is
the size of the buffer which accumulates slave data when slave is disconnected for
some time, so that when slave reconnects again, only transfer the portion of data
which the slave missed. Minimum value is 16K.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "16384-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "repl-backlog-ttl",
            "ParameterValue": "3600",
            "Description": "The amount of time in seconds after the master no longer
have any slaves connected for the master to free the replication backlog. A value
of 0 means to never release the backlog.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "replica-allow-chaining",
            "ParameterValue": "no",
            "Description": "Configures if chaining of replicas is allowed",
            "Source": "system",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "replica-ignore-maxmemory",
            "ParameterValue": "yes",
            "Description": "Determines if replica ignores maxmemory setting by not
evicting items independent from the master",
            "Source": "system",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "replica-lazy-flush",
            "ParameterValue": "no",
            "Description": "Perform an asynchronous flushDB during replica sync",
            "Source": "system",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "IsModifiable": false,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "reserved-memory-percent",
            "ParameterValue": "25",
            "Description": "The percent of memory reserved for non-cache memory
usage. You may want to increase this parameter for nodes with read replicas, AOF
enabled, etc, to reduce swap usage.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-100",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "set-max-intset-entries",
            "ParameterValue": "512",
            "Description": "The limit in the size of the set in order for the
dataset to be compressed.",
            "Source": "user",
            "DataType": "integer",
```

```
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "slowlog-log-slower-than",
            "ParameterValue": "10000",
            "Description": "The execution time, in microseconds, to exceed in order
for the command to get logged. Note that a negative number disables the slow log,
while a value of zero forces the logging of every command.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "slowlog-max-len",
            "ParameterValue": "128",
            "Description": "The length of the slow log. There is no limit to this
length. Just be aware that it will consume memory. You can reclaim memory used by
the slow log with SLOWLOG RESET.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "stream-node-max-bytes",
            "ParameterValue": "4096",
            "Description": "The maximum size of a single node in a stream in bytes",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "stream-node-max-entries",
```

```
            "ParameterValue": "100",
            "Description": "The maximum number of items a single node in a stream
can contain",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "tcp-keepalive",
            "ParameterValue": "300",
            "Description": "If non-zero, send ACKs every given number of seconds.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "timeout",
            "ParameterValue": "0",
            "Description": "Close connection if client is idle for a given number of
seconds, or never if 0.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0,20-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        },
        {
            "ParameterName": "zset-max-ziplist-entries",
            "ParameterValue": "128",
            "Description": "The maximum number of sorted set entries in order for
the dataset to be compressed.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
```

```
        },
        {
            "ParameterName": "zset-max-ziplist-value",
            "ParameterValue": "64",
            "Description": "The threshold of biggest sorted set entries in order for
 the dataset to be compressed.",
            "Source": "user",
            "DataType": "integer",
            "AllowedValues": "0-",
            "IsModifiable": true,
            "MinimumEngineVersion": "5.0.0",
            "ChangeType": "immediate"
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的参数管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCacheParameters。

describe-cache-subnet-groups

以下代码示例演示了如何使用 describe-cache-subnet-groups。

Amazon CLI

描述缓存子网组

以下 describe-cache-subnet-groups 示例返回子网组列表。

```
aws elasticache describe-cache-subnet-groups
```

输出：

```
{
    "CacheSubnetGroups": [
        {
            "CacheSubnetGroupName": "default",
            "CacheSubnetGroupDescription": "Default CacheSubnetGroup",
            "VpcId": "vpc-a3e97cdb",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-8d4bacf5",
```

```
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-dde21380",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-6485ec4f",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    }
                },
                {
                    "SubnetIdentifier": "subnet-b4ebebff",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    }
                }
            ]
        },
        {
            "CacheSubnetGroupName": "kxkxk",
            "CacheSubnetGroupDescription": "mygroup",
            "VpcId": "vpc-a3e97cdb",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-b4ebebff",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    }
                }
            ]
        },
        {
            "CacheSubnetGroupName": "test",
            "CacheSubnetGroupDescription": "test",
            "VpcId": "vpc-a3e97cdb",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-b4ebebff",
```

```
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    }
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的子网和子网组，或《ElastiCache for Memcached 用户指南》中的子网和子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCacheSubnetGroups。

describe-engine-default-parameters

以下代码示例演示了如何使用 describe-engine-default-parameters。

Amazon CLI

描述引擎默认参数

以下 describe-engine-default-parameters 示例返回指定缓存引擎的默认引擎和系统参数信息。

```
aws elasticache describe-engine-default-parameters \
    --cache-parameter-group-family "redis5.0"
```

输出：

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "activedefrag",
                "ParameterValue": "no",
                "Description": "Enabled active memory defragmentation",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
```

```
            },
            {
                "ParameterName": "active-defrag-cycle-max",
                "ParameterValue": "75",
                "Description": "Maximal effort for defrag in CPU percentage",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-75",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "active-defrag-cycle-min",
                "ParameterValue": "5",
                "Description": "Minimal effort for defrag in CPU percentage",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-75",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "active-defrag-ignore-bytes",
                "ParameterValue": "104857600",
                "Description": "Minimum amount of fragmentation waste to start
active defrag",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1048576-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "active-defrag-max-scan-fields",
                "ParameterValue": "1000",
                "Description": "Maximum number of set/hash/zset/list fields that
will be processed from the main dictionary scan",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-1000000",
                "IsModifiable": true,
```

```
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "active-defrag-threshold-lower",
                "ParameterValue": "10",
                "Description": "Minimum percentage of fragmentation to start active
defrag",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-100",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "active-defrag-threshold-upper",
                "ParameterValue": "100",
                "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-100",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "activerehashing",
                "ParameterValue": "yes",
                "Description": "Apply rehashing or not.",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "requires-reboot"
            },
            {
                "ParameterName": "appendfsync",
                "ParameterValue": "everysec",
                "Description": "fsync policy for AOF persistence",
                "Source": "system",
                "DataType": "string",
```

```
                "AllowedValues": "always,everysec,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "appendonly",
                "ParameterValue": "no",
                "Description": "Enable Redis persistence.",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-normal-hard-limit",
                "ParameterValue": "0",
                "Description": "Normal client output buffer hard limit in bytes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-normal-soft-limit",
                "ParameterValue": "0",
                "Description": "Normal client output buffer soft limit in bytes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-normal-soft-seconds",
                "ParameterValue": "0",
                "Description": "Normal client output buffer soft limit in seconds.",
                "Source": "system",
                "DataType": "integer",
```

```
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-pubsub-hard-limit",
                "ParameterValue": "33554432",
                "Description": "Pubsub client output buffer hard limit in bytes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-pubsub-soft-limit",
                "ParameterValue": "8388608",
                "Description": "Pubsub client output buffer soft limit in bytes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-pubsub-soft-seconds",
                "ParameterValue": "60",
                "Description": "Pubsub client output buffer soft limit in seconds.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-output-buffer-limit-replica-soft-seconds",
                "ParameterValue": "60",
                "Description": "Replica client output buffer soft limit in
seconds.",
                "Source": "system",
```

```
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "client-query-buffer-limit",
                "ParameterValue": "1073741824",
                "Description": "Max size of a single client query buffer",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1048576-1073741824",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "close-on-replica-write",
                "ParameterValue": "yes",
                "Description": "If enabled, clients who attempt to write to a read-
only replica will be disconnected. Applicable to 2.8.23 and higher.",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "cluster-enabled",
                "ParameterValue": "no",
                "Description": "Enable cluster mode",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "requires-reboot"
            },
            {
                "ParameterName": "cluster-require-full-coverage",
                "ParameterValue": "no",
```

```
                "Description": "Whether cluster becomes unavailable if one or more
slots are not covered",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "databases",
                "ParameterValue": "16",
                "Description": "Set the number of databases.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-1200000",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "requires-reboot"
            },
            {
                "ParameterName": "hash-max-ziplist-entries",
                "ParameterValue": "512",
                "Description": "The maximum number of hash entries in order for the
dataset to be compressed.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "hash-max-ziplist-value",
                "ParameterValue": "64",
                "Description": "The threshold of biggest hash entries in order for
the dataset to be compressed.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
```

```
            {
                "ParameterName": "hll-sparse-max-bytes",
                "ParameterValue": "3000",
                "Description": "HyperLogLog sparse representation bytes limit",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-16000",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lazyfree-lazy-eviction",
                "ParameterValue": "no",
                "Description": "Perform an asynchronous delete on evictions",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lazyfree-lazy-expire",
                "ParameterValue": "no",
                "Description": "Perform an asynchronous delete on expired keys",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lazyfree-lazy-server-del",
                "ParameterValue": "no",
                "Description": "Perform an asynchronous delete on key updates",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
```

```
            {
                "ParameterName": "lfu-decay-time",
                "ParameterValue": "1",
                "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policy",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lfu-log-factor",
                "ParameterValue": "10",
                "Description": "The log factor for incrementing key counter for LFU
eviction policy",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "list-compress-depth",
                "ParameterValue": "0",
                "Description": "Number of quicklist ziplist nodes from each side
of the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "list-max-ziplist-size",
                "ParameterValue": "-2",
                "Description": "The number of entries allowed per internal list node
can be specified as a fixed maximum size or a maximum number of elements",
                "Source": "system",
                "DataType": "integer",
```

```
                "AllowedValues": "-5,-4,-3,-2,-1,1-",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lua-replicate-commands",
                "ParameterValue": "yes",
                "Description": "Always enable Lua effect replication or not",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "lua-time-limit",
                "ParameterValue": "5000",
                "Description": "Max execution time of a Lua script in milliseconds.
0 for unlimited execution without warnings.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "5000",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "maxclients",
                "ParameterValue": "65000",
                "Description": "The maximum number of Redis clients.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-65000",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "requires-reboot"
            },
            {
                "ParameterName": "maxmemory-policy",
                "ParameterValue": "volatile-lru",
                "Description": "Max memory policy.",
                "Source": "system",
```

```
                "DataType": "string",
                "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-
lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "maxmemory-samples",
                "ParameterValue": "3",
                "Description": "Max memory samples.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "min-replicas-max-lag",
                "ParameterValue": "10",
                "Description": "The maximum amount of replica lag in seconds beyond
 which the master would stop taking writes. A value of 0 means the master always
 takes writes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "min-replicas-to-write",
                "ParameterValue": "0",
                "Description": "The minimum number of replicas that must be present
 with lag no greater than min-replicas-max-lag for master to take writes. Setting
 this to 0 means the master always takes writes.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
```

```
            {
                "ParameterName": "notify-keyspace-events",
                "Description": "The keyspace events for Redis to notify Pub/Sub
clients about. By default all notifications are disabled",
                "Source": "system",
                "DataType": "string",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "proto-max-bulk-len",
                "ParameterValue": "536870912",
                "Description": "Max size of a single element request",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "1048576-536870912",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "rename-commands",
                "ParameterValue": "",
                "Description": "Redis commands that can be dynamically renamed by
the customer",
                "Source": "system",
                "DataType": "string",
                "AllowedValues":
"APPEND,BITCOUNT,BITFIELD,BITOP,BITPOS,BLPOP,BRPOP,BRPOPLPUSH,BZPOPMIN,BZPOPMAX,CLIENT,COMM
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.3",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "repl-backlog-size",
                "ParameterValue": "1048576",
                "Description": "The replication backlog size in bytes for PSYNC.
This is the size of the buffer which accumulates slave data when slave is
disconnected for some time, so that when slave reconnects again, only transfer the
portion of data which the slave missed. Minimum value is 16K.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "16384-",
```

```
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "repl-backlog-ttl",
                "ParameterValue": "3600",
                "Description": "The amount of time in seconds after the master no
longer have any slaves connected for the master to free the replication backlog. A
value of 0 means to never release the backlog.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "replica-allow-chaining",
                "ParameterValue": "no",
                "Description": "Configures if chaining of replicas is allowed",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "replica-ignore-maxmemory",
                "ParameterValue": "yes",
                "Description": "Determines if replica ignores maxmemory setting by
not evicting items independent from the master",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "replica-lazy-flush",
                "ParameterValue": "no",
```

```
                "Description": "Perform an asynchronous flushDB during replica
sync",
                "Source": "system",
                "DataType": "string",
                "AllowedValues": "yes,no",
                "IsModifiable": false,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "reserved-memory-percent",
                "ParameterValue": "25",
                "Description": "The percent of memory reserved for non-cache memory
usage. You may want to increase this parameter for nodes with read replicas, AOF
enabled, etc, to reduce swap usage.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-100",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "set-max-intset-entries",
                "ParameterValue": "512",
                "Description": "The limit in the size of the set in order for the
dataset to be compressed.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "slowlog-log-slower-than",
                "ParameterValue": "10000",
                "Description": "The execution time, in microseconds, to exceed in
order for the command to get logged. Note that a negative number disables the slow
log, while a value of zero forces the logging of every command.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "-",
                "IsModifiable": true,
```

```
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "slowlog-max-len",
                "ParameterValue": "128",
                "Description": "The length of the slow log. There is no limit to
this length. Just be aware that it will consume memory. You can reclaim memory used
by the slow log with SLOWLOG RESET.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "stream-node-max-bytes",
                "ParameterValue": "4096",
                "Description": "The maximum size of a single node in a stream in
bytes",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "stream-node-max-entries",
                "ParameterValue": "100",
                "Description": "The maximum number of items a single node in a
stream can contain",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "tcp-keepalive",
                "ParameterValue": "300",
```

```
                "Description": "If non-zero, send ACKs every given number of
seconds.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "timeout",
                "ParameterValue": "0",
                "Description": "Close connection if client is idle for a given
number of seconds, or never if 0.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0,20-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "zset-max-ziplist-entries",
                "ParameterValue": "128",
                "Description": "The maximum number of sorted set entries in order
for the dataset to be compressed.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
            },
            {
                "ParameterName": "zset-max-ziplist-value",
                "ParameterValue": "64",
                "Description": "The threshold of biggest sorted set entries in order
for the dataset to be compressed.",
                "Source": "system",
                "DataType": "integer",
                "AllowedValues": "0-",
                "IsModifiable": true,
                "MinimumEngineVersion": "5.0.0",
                "ChangeType": "immediate"
```

```
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngineDefaultParameters。

`describe-events`

以下代码示例演示了如何使用 `describe-events`。

Amazon CLI

描述复制组的事件

以下 `describe-events` 示例返回复制组的事件列表。

```
aws elasticache describe-events \
    --source-identifier test-cluster \
    --source-type replication-group
```

输出：

```
{
    "Events": [
        {
            "SourceIdentifier": "test-cluster",
            "SourceType": "replication-group",
            "Message": "Automatic failover has been turned on for replication group
 test-cluster",
            "Date": "2020-03-18T23:51:34.457Z"
        },
        {
            "SourceIdentifier": "test-cluster",
            "SourceType": "replication-group",
            "Message": "Replication group test-cluster created",
            "Date": "2020-03-18T23:50:31.378Z"
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的监控事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-global-replication-groups

以下代码示例演示了如何使用 describe-global-replication-groups。

Amazon CLI

描述全局复制组

以下 describe-global-replication-groups 示例返回全局数据存储的详细信息。

```
aws elasticache describe-global-replication-groups \
    --global-replication-group-id my-grg
```

输出：

```
{
    "GlobalReplicationGroups": [
        {
            "GlobalReplicationGroupId": "my-grg",
            "GlobalReplicationGroupDescription": "my-grg",
            "Status": "creating",
            "CacheNodeType": "cache.r5.large",
            "Engine": "redis",
            "EngineVersion": "5.0.6",
            "ClusterEnabled": false,
            "AuthTokenEnabled": false,
            "TransitEncryptionEnabled": false,
            "AtRestEncryptionEnabled": false
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGlobalReplicationGroups。

describe-replication-groups

以下代码示例演示了如何使用 describe-replication-groups。

Amazon CLI

返回复制组详细信息列表

以下 describe-replication-groups 示例返回复制组。

```
aws elasticache describe-replication-groups
```

输出：

```
{
    "ReplicationGroups": [
        {
            "ReplicationGroupId": "my-cluster",
            "Description": "mycluster",
            "Status": "available",
            "PendingModifiedValues": {},
            "MemberClusters": [
                "pat-cluster-001",
                "pat-cluster-002",
                "pat-cluster-003",
                "pat-cluster-004"
            ],
            "NodeGroups": [
                {
                    "NodeGroupId": "0001",
                    "Status": "available",
                    "PrimaryEndpoint": {
                        "Address": "my-
cluster.xxxxih.ng.0001.usw2.cache.amazonaws.com",
                        "Port": 6379
                    },
                    "ReaderEndpoint": {
                        "Address": "my-cluster-
ro.xxxxih.ng.0001.usw2.cache.amazonaws.com",
                        "Port": 6379
                    },
                    "NodeGroupMembers": [
                        {
                            "CacheClusterId": "my-cluster-001",
```

```
                            "CacheNodeId": "0001",
                            "ReadEndpoint": {
                                "Address": "pat-
cluster-001.xxxih.0001.usw2.cache.amazonaws.com",
                                "Port": 6379
                            },
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CurrentRole": "primary"
                        },
                        {
                            "CacheClusterId": "my-cluster-002",
                            "CacheNodeId": "0001",
                            "ReadEndpoint": {
                                "Address": "pat-
cluster-002.xxxxih.0001.usw2.cache.amazonaws.com",
                                "Port": 6379
                            },
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CurrentRole": "replica"
                        },
                        {
                            "CacheClusterId": "my-cluster-003",
                            "CacheNodeId": "0001",
                            "ReadEndpoint": {
                                "Address": "pat-
cluster-003.xxxxih.0001.usw2.cache.amazonaws.com",
                                "Port": 6379
                            },
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CurrentRole": "replica"
                        },
                        {
                            "CacheClusterId": "my-cluster-004",
                            "CacheNodeId": "0001",
                            "ReadEndpoint": {
                                "Address": "pat-
cluster-004.xxxih.0001.usw2.cache.amazonaws.com",
                                "Port": 6379
                            },
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CurrentRole": "replica"
                        }
                    ]
                }
```

```
            ],
            "AutomaticFailover": "disabled",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "07:30-08:30",
            "ClusterEnabled": false,
            "CacheNodeType": "cache.r5.xlarge",
            "AuthTokenEnabled": false,
            "TransitEncryptionEnabled": false,
            "AtRestEncryptionEnabled": false,
            "ARN": "arn:aws:elasticache:us-
west-2:xxxxxxxxxxx152:replicationgroup:my-cluster",
            "LogDeliveryConfigurations": [
                {
                    "LogType": "slow-log",
                    "DestinationType": "cloudwatch-logs",
                    "DestinationDetails": {
                        "CloudWatchLogsDetails": {
                            "LogGroup": "test-log"
                        }
                    },
                    "LogFormat": "json",
                    "Status": "active"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的管理集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReplicationGroups。

describe-reserved-cache-nodes-offerings

以下代码示例演示了如何使用 describe-reserved-cache-nodes-offerings。

Amazon CLI

描述 reserved-cache-nodes-offerings

以下 describe-reserved-cache-nodes-offerings 示例返回 reserved-cache-node 选项的详细信息。

```
aws elasticache describe-reserved-cache-nodes-offerings
```

输出：

```
{
    "ReservedCacheNodesOfferings": [
        {
            "ReservedCacheNodesOfferingId": "01ce0a19-a476-41cb-8aee-48eacbcdc8e5",
            "CacheNodeType": "cache.t3.small",
            "Duration": 31536000,
            "FixedPrice": 97.0,
            "UsagePrice": 0.0,
            "ProductDescription": "memcached",
            "OfferingType": "Partial Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.011,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
        },
        {
            "ReservedCacheNodesOfferingId": "0443a27b-4da5-4b90-b92d-929fbd7abed2",
            "CacheNodeType": "cache.m3.2xlarge",
            "Duration": 31536000,
            "FixedPrice": 1772.0,
            "UsagePrice": 0.0,
            "ProductDescription": "redis",
            "OfferingType": "Heavy Utilization",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.25,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
        },

        ...

    ]
}
```

有关更多信息，请参阅《ElastiCache Redis 用户指南》中的获取有关预留节点产品的信息，或《ElastiCache Memcached 用户指南》中的获取有关预留节点产品的信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedCacheNodesOfferings。

`describe-reserved-cache-nodes`

以下代码示例演示了如何使用 `describe-reserved-cache-nodes`。

Amazon CLI

描述预留缓存节点

以下 `describe-reserved-cache-nodes` 示例返回有关此账户的预留缓存节点或指定预留缓存节点的信息。

aws elasticache describe-reserved-cache-nodes

输出：

```
{
    "ReservedCacheNodes": [
        {
            "ReservedCacheNodeId": "mynode",
            "ReservedCacheNodesOfferingId": "xxxxxxxxx-xxxxx-xxxxx-xxxx-xxxxxxxx71",
            "CacheNodeType": "cache.t3.small",
            "StartTime": "2019-12-06T02:50:44.003Z",
            "Duration": 31536000,
            "FixedPrice": 0.0,
            "UsagePrice": 0.0,
            "CacheNodeCount": 1,
            "ProductDescription": "redis",
            "OfferingType": "No Upfront",
            "State": "payment-pending",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.023,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "ReservationARN": "arn:aws:elasticache:us-
west-2:xxxxxxxxxxx52:reserved-instance:mynode"
        }
```

```
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用预留节点管理成本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedCacheNodes。

describe-service-updates

以下代码示例演示了如何使用 describe-service-updates。

Amazon CLI

描述服务更新

以下 describe-service-updates 示例返回有关服务更新的详细信息。

```
aws elasticache describe-service-updates
```

输出：

```
{
     "ServiceUpdates": [
         {
             "ServiceUpdateName": "elc-xxxxxxxx7-001",
             "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
             "ServiceUpdateEndDate": "2020-02-09T15:59:59Z",
             "ServiceUpdateSeverity": "important",
             "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
             "ServiceUpdateStatus": "available",
             "ServiceUpdateDescription": "Upgrades to improve the security,
 reliability, and operational performance of your ElastiCache nodes",
             "ServiceUpdateType": "security-update",
             "Engine": "redis, memcached",
             "EngineVersion": "redis 2.6.13 and onwards, memcached 1.4.5 and
 onwards",
             "AutoUpdateAfterRecommendedApplyByDate": false,
             "EstimatedUpdateTime": "30 minutes per node"
         },
         {
             "ServiceUpdateName": "elc-xxxxxxxx4-001",
             "ServiceUpdateReleaseDate": "2019-06-11T15:00:00Z",
             "ServiceUpdateEndDate": "2019-10-01T09:24:00Z",
```

```
            "ServiceUpdateSeverity": "important",
            "ServiceUpdateRecommendedApplyByDate": "2019-07-11T14:59:59Z",
            "ServiceUpdateStatus": "expired",
            "ServiceUpdateDescription": "Upgrades to improve the security,
 reliability, and operational performance of your ElastiCache nodes",
            "ServiceUpdateType": "security-update",
            "Engine": "redis",
            "EngineVersion": "redis 3.2.6, redis 4.0 and onwards",
            "AutoUpdateAfterRecommendedApplyByDate": false,
            "EstimatedUpdateTime": "30 minutes per node"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeServiceUpdates。

`describe-snapshots`

以下代码示例演示了如何使用 `describe-snapshots`。

Amazon CLI

描述快照

以下“describe-snapshots”示例返回有关您的集群或复制组快照的信息。

```
aws elasticache describe-snapshots
```

输出：

```
{
    "Snapshots": [
        {
            "SnapshotName": "automatic.my-cluster2-002-2019-12-05-06-38",
            "CacheClusterId": "my-cluster2-002",
            "SnapshotStatus": "available",
            "SnapshotSource": "automated",
            "CacheNodeType": "cache.r5.large",
            "Engine": "redis",
            "EngineVersion": "5.0.5",
            "NumCacheNodes": 1,
            "PreferredAvailabilityZone": "us-west-2a",
            "CacheClusterCreateTime": "2019-11-26T01:22:52.396Z",
```

```
            "PreferredMaintenanceWindow": "mon:17:30-mon:18:30",
            "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxx52:My_Topic",
            "Port": 6379,
            "CacheParameterGroupName": "default.redis5.0",
            "CacheSubnetGroupName": "kxkxk",
            "VpcId": "vpc-a3e97cdb",
            "AutoMinorVersionUpgrade": true,
            "SnapshotRetentionLimit": 1,
            "SnapshotWindow": "06:30-07:30",
            "NodeSnapshots": [
                {
                    "CacheNodeId": "0001",
                    "CacheSize": "5 MB",
                    "CacheNodeCreateTime": "2019-11-26T01:22:52.396Z",
                    "SnapshotCreateTime": "2019-12-05T06:38:23Z"
                }
            ]
        },
        {
            "SnapshotName": "myreplica-backup",
            "CacheClusterId": "myreplica",
            "SnapshotStatus": "available",
            "SnapshotSource": "manual",
            "CacheNodeType": "cache.r5.large",
            "Engine": "redis",
            "EngineVersion": "5.0.5",
            "NumCacheNodes": 1,
            "PreferredAvailabilityZone": "us-west-2a",
            "CacheClusterCreateTime": "2019-11-26T00:14:52.439Z",
            "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
            "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxx152:My_Topic",
            "Port": 6379,
            "CacheParameterGroupName": "default.redis5.0",
            "CacheSubnetGroupName": "kxkxk",
            "VpcId": "vpc-a3e97cdb",
            "AutoMinorVersionUpgrade": true,
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "09:00-10:00",
            "NodeSnapshots": [
                {
                    "CacheNodeId": "0001",
                    "CacheSize": "5 MB",
                    "CacheNodeCreateTime": "2019-11-26T00:14:52.439Z",
                    "SnapshotCreateTime": "2019-11-26T00:25:01Z"
```

```
                }
            ]
        },
        {
            "SnapshotName": "my-cluster",
            "CacheClusterId": "my-cluster-003",
            "SnapshotStatus": "available",
            "SnapshotSource": "manual",
            "CacheNodeType": "cache.r5.large",
            "Engine": "redis",
            "EngineVersion": "5.0.5",
            "NumCacheNodes": 1,
            "PreferredAvailabilityZone": "us-west-2a",
            "CacheClusterCreateTime": "2019-11-25T23:56:17.186Z",
            "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
            "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxx152:My_Topic",
            "Port": 6379,
            "CacheParameterGroupName": "default.redis5.0",
            "CacheSubnetGroupName": "kxkxk",
            "VpcId": "vpc-a3e97cdb",
            "AutoMinorVersionUpgrade": true,
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "09:00-10:00",
            "NodeSnapshots": [
                {
                    "CacheNodeId": "0001",
                    "CacheSize": "5 MB",
                    "CacheNodeCreateTime": "2019-11-25T23:56:17.186Z",
                    "SnapshotCreateTime": "2019-11-26T03:08:33Z"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 ElastiCache for Redis 备份和还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshots。

describe-update-actions

以下代码示例演示了如何使用 describe-update-actions。

Amazon CLI

描述更新操作

以下 describe-update-actions 示例返回更新操作的详细信息。

```
aws elasticache describe-update-actions
```

输出：

```
{
    "UpdateActions": [
        {
            "ReplicationGroupId": "mycluster",
            "ServiceUpdateName": "elc-20191007-001",
            "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
            "ServiceUpdateSeverity": "important",
            "ServiceUpdateStatus": "available",
            "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
            "ServiceUpdateType": "security-update",
            "UpdateActionAvailableDate": "2019-12-05T19:15:19.995Z",
            "UpdateActionStatus": "complete",
            "NodesUpdated": "9/9",
            "UpdateActionStatusModifiedDate": "2019-12-05T19:15:20.461Z",
            "SlaMet": "n/a",
            "Engine": "redis"
        },
        {
            "CacheClusterId": "my-memcached-cluster",
            "ServiceUpdateName": "elc-20191007-001",
            "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
            "ServiceUpdateSeverity": "important",
            "ServiceUpdateStatus": "available",
            "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
            "ServiceUpdateType": "security-update",
            "UpdateActionAvailableDate": "2019-12-04T18:26:05.349Z",
            "UpdateActionStatus": "complete",
            "NodesUpdated": "1/1",
            "UpdateActionStatusModifiedDate": "2019-12-04T18:26:05.352Z",
            "SlaMet": "n/a",
            "Engine": "redis"
        },
        {
```

```
            "ReplicationGroupId": "my-cluster",
            "ServiceUpdateName": "elc-20191007-001",
            "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
            "ServiceUpdateSeverity": "important",
            "ServiceUpdateStatus": "available",
            "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
            "ServiceUpdateType": "security-update",
            "UpdateActionAvailableDate": "2019-11-26T03:36:26.320Z",
            "UpdateActionStatus": "complete",
            "NodesUpdated": "4/4",
            "UpdateActionStatusModifiedDate": "2019-12-04T22:11:12.664Z",
            "SlaMet": "n/a",
            "Engine": "redis"
        },
        {
            "ReplicationGroupId": "my-cluster2",
            "ServiceUpdateName": "elc-20191007-001",
            "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
            "ServiceUpdateSeverity": "important",
            "ServiceUpdateStatus": "available",
            "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
            "ServiceUpdateType": "security-update",
            "UpdateActionAvailableDate": "2019-11-26T01:26:01.617Z",
            "UpdateActionStatus": "complete",
            "NodesUpdated": "3/3",
            "UpdateActionStatusModifiedDate": "2019-11-26T01:26:01.753Z",
            "SlaMet": "n/a",
            "Engine": "redis"
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的 Amazon ElastiCache 中的自助更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUpdateActions。

`describe-user-groups`

以下代码示例演示了如何使用 `describe-user-groups`。

Amazon CLI

描述 user-groups

以下 describe-user-groups 示例返回用户组列表。

```
aws elasticache describe-user-groups
```

输出：

```
{
    "UserGroups": [
        {
            "UserGroupId": "myusergroup",
            "Status": "active",
            "Engine": "redis",
            "UserIds": [
                "default"
            ],
            "ReplicationGroups": [],
            "ARN": "arn:aws:elasticache:us-
west-2:xxxxxxxxxx52:usergroup:myusergroup"
        }
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUserGroups。

describe-users

以下代码示例演示了如何使用 describe-users。

Amazon CLI

描述用户

以下 describe-users 示例返回用户列表。

```
aws elasticache describe-users
```

输出：

```
{
```

```
    "Users": [
        {
            "UserId": "default",
            "UserName": "default",
            "Status": "active",
            "Engine": "redis",
            "AccessString": "on ~* +@all",
            "UserGroupIds": [
                "myusergroup"
            ],
            "Authentication": {
                "Type": "no-password"
            },
            "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:default"
        },
        {
            "UserId": "user1",
            "UserName": "myUser",
            "Status": "active",
            "Engine": "redis",
            "AccessString": "on ~* +@all",
            "UserGroupIds": [],
            "Authentication": {
                "Type": "password",
                "PasswordCount": 1
            },
            "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user1"
        },
        {
            "UserId": "user2",
            "UserName": "myUser",
            "Status": "active",
            "Engine": "redis",
            "AccessString": "on ~app::* -@all +@read +@hash +@bitmap +@geo -setbit -
bitfield -hset -hsetnx -hmset -hincrby -hincrbyfloat -hdel -bitop -geoadd -georadius
 -georadiusbymember",
            "UserGroupIds": [],
            "Authentication": {
                "Type": "password",
                "PasswordCount": 1
            },
            "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"
        }
    ]
```

```
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUsers。

disassociate-global-replication-group

以下代码示例演示了如何使用 disassociate-global-replication-group。

Amazon CLI

取消辅助集群与全局复制组的关联

以下 disassociate-global-replication-group 示例从全局数据存储中删除辅助集群。

```
aws elasticache disassociate-global-replication-group \
    --global-replication-group-id my-grg \
    --replication-group-id my-cluster-grg-secondary \
    --replication-group-region us-east-1
```

输出：

```
{
    "GlobalReplicationGroup": {
        "GlobalReplicationGroupId": "my-grg",
        "GlobalReplicationGroupDescription": "my-grg",
        "Status": "modifying",
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.6",
        "Members": [
            {
                "ReplicationGroupId": "my-cluster-grg-secondary",
                "ReplicationGroupRegion": "us-east-1",
                "Role": "SECONDARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            },
            {
                "ReplicationGroupId": "my-cluster-grg",
                "ReplicationGroupRegion": "us-west-2",
```

```
                "Role": "PRIMARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            }
        ],
        "ClusterEnabled": false,
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateGlobalReplicationGroup。

`increase-node-groups-in-global-replication-group`

以下代码示例演示了如何使用 `increase-node-groups-in-global-replication-group`。

Amazon CLI

增加全局复制组中节点组的数量

以下 `increase-node-groups-in-global-replication-group` 使用 Redis 引擎增加节点组计数。

```
aws elasticache increase-node-groups-in-global-replication-group \
    --global-replication-group-id sgaui-pat-test-4 \
    --node-group-count 6 \
    --apply-immediately
```

输出：

```
{
    "GlobalReplicationGroup": {
        "GlobalReplicationGroupId": "sgaui-test-4",
        "GlobalReplicationGroupDescription": "test-4",
        "Status": "modifying",
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
```

```
        "EngineVersion": "5.0.6",
        "Members": [
            {
                "ReplicationGroupId": "my-cluster-b",
                "ReplicationGroupRegion": "us-east-1",
                "Role": "SECONDARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            },
            {
                "ReplicationGroupId": "my-cluster-a",
                "ReplicationGroupRegion": "us-west-2",
                "Role": "PRIMARY",
                "AutomaticFailover": "enabled",
                "Status": "associated"
            }
        ],
        "ClusterEnabled": true,
        "GlobalNodeGroups": [
            {
                "GlobalNodeGroupId": "sgaui-test-4-0001",
                "Slots": "0-234,2420-5461"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-4-0002",
                "Slots": "5462-5904,6997-9830"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-4-0003",
                "Slots": "10923-11190,13375-16383"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-4-0004",
                "Slots": "235-2419,5905-6996"
            },
            {
                "GlobalNodeGroupId": "sgaui-test-4-0005",
                "Slots": "9831-10922,11191-13374"
            }
        ],
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
```

```
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IncreaseNodeGroupsInGlobalReplicationGroup。

`increase-replica-count`

以下代码示例演示了如何使用 `increase-replica-count`。

Amazon CLI

增加副本计数

以下 `increase-replica-count` 示例执行以下两项操作之一。它可以动态增加 Redis（已禁用集群模式）复制组中的副本数量。或者，它可以动态增加 Redis（已启用集群模式）复制组中一个或多个节点组（分区）中的副本节点数量。此操作在无需集群停机的情况下执行。

```
aws elasticache increase-replica-count \
    --replication-group-id "my-cluster" \
    --apply-immediately \
    --new-replica-count 3
```

输出：

```
{
     "ReplicationGroup": {
         "ReplicationGroupId": "my-cluster",
         "Description": " ",
         "Status": "modifying",
         "PendingModifiedValues": {},
         "MemberClusters": [
             "my-cluster-001",
             "my-cluster-002",
             "my-cluster-003",
             "my-cluster-004"
         ],
         "NodeGroups": [
             {
                 "NodeGroupId": "0001",
```

```
                "Status": "modifying",
                "PrimaryEndpoint": {
                    "Address": "my-
cluster.xxxxxih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "my-cluster-
ro.xxxxxxih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "my-cluster-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "my-
cluster-001.xxxxxih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "my-cluster-003",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "my-
cluster-003.xxxxxih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "replica"
                    }
                ]
            }
        ],
        "AutomaticFailover": "disabled",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "07:30-08:30",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.xlarge",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
```

```
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的增加分区中的副本数量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IncreaseReplicaCount。

`list-allowed-node-type-modifications`

以下代码示例演示了如何使用 `list-allowed-node-type-modifications`。

Amazon CLI

列出允许的节点修改

以下 `list-allowed-node-type-modifications` 示例列出了您可以将 Redis 集群或复制组的当前节点类型扩展到的所有可用节点类型。

```
aws elasticache list-allowed-node-type-modifications \
    --replication-group-id "my-replication-group"
```

输出：

```
{
    "ScaleUpModifications": [
        "cache.m5.12xlarge",
        "cache.m5.24xlarge",
        "cache.m5.4xlarge",
        "cache.r5.12xlarge",
        "cache.r5.24xlarge",
        "cache.r5.2xlarge",
        "cache.r5.4xlarge"
    ],
    "ScaleDownModifications": [
        "cache.m3.large",
        "cache.m3.medium",
        "cache.m3.xlarge",
        "cache.m4.large",
        "cache.m4.xlarge",
        "cache.m5.2xlarge",
        "cache.m5.large",
        "cache.m5.xlarge",
        "cache.r3.large",
```

```
        "cache.r4.large",
        "cache.r4.xlarge",
        "cache.r5.large",
        "cache.t2.medium",
        "cache.t2.micro",
        "cache.t2.small",
        "cache.t3.medium",
        "cache.t3.micro",
        "cache.t3.small"
    ]
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的扩展 ElastiCache for Redis 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAllowedNodeTypeModifications。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出了资源的标签。

```
aws elasticache list-tags-for-resource \
    --resource-name "arn:aws:elasticache:us-east-1:123456789012:cluster:my-cluster"
```

输出：

```
{
    "TagList": [
        {
            "Key": "Project",
            "Value": "querySpeedUp"
        },
        {
            "Key": "Environment",
            "Value": "PROD"
        }
    ]
```

```
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用 Amazon CLI 列出标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

modify-cache-cluster

以下代码示例演示了如何使用 modify-cache-cluster。

Amazon CLI

修改缓存集群

以下 modify-cache-cluster 示例修改指定集群的设置。

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id "my-cluster" \
    --num-cache-nodes 1
```

输出：

```
{
    "CacheCluster": {
        "CacheClusterId": "my-cluster",
        "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
        "CacheNodeType": "cache.m5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "CacheClusterStatus": "available",
        "NumCacheNodes": 1,
        "PreferredAvailabilityZone": "us-west-2c",
        "CacheClusterCreateTime": "2019-12-04T18:24:56.652Z",
        "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
        "PendingModifiedValues": {},
        "CacheSecurityGroups": [],
        "CacheParameterGroup": {
            "CacheParameterGroupName": "default.redis5.0",
            "ParameterApplyStatus": "in-sync",
            "CacheNodeIdsToReboot": []
        },
        "CacheSubnetGroupName": "default",
```

```
        "AutoMinorVersionUpgrade": true,
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "07:00-08:00",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的修改 ElastiCache 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCacheCluster。

modify-cache-parameter-group

以下代码示例演示了如何使用 modify-cache-parameter-group。

Amazon CLI

修改缓存参数组

以下 modify-cache-parameter-group 示例修改指定的缓存参数组的参数。

```
aws elasticache modify-cache-parameter-group \
    --cache-parameter-group-name "mygroup" \
    --parameter-name-values "ParameterName=activedefrag, ParameterValue=no"
```

输出：

```
{
    "CacheParameterGroupName": "mygroup"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的修改参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCacheParameterGroup。

modify-cache-subnet-group

以下代码示例演示了如何使用 modify-cache-subnet-group。

Amazon CLI

修改缓存子网组

以下 modify-cache-subnet-group 示例修改指定的缓存子网组。

```
aws elasticache modify-cache-subnet-group \
    --cache-subnet-group-name kxkxk \
    --cache-subnet-group-description "mygroup"
```

输出：

```
{
    "CacheSubnetGroup": {
        "CacheSubnetGroupName": "kxkxk",
        "CacheSubnetGroupDescription": "mygroup",
        "VpcId": "vpc-xxxxcdb",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-xxxxbff",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的修改子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCacheSubnetGroup。

modify-global-replication-group

以下代码示例演示了如何使用 modify-global-replication-group。

Amazon CLI

修改全局复制组

以下 modify-global-replication-group 使用 Redis 引擎修改全局复制组的属性，在本例中禁用自动失效转移。

```
aws elasticache modify-global-replication-group \
    --global-replication-group-id sgaui-pat-group \
```

```
    --apply-immediately \
    --no-automatic-failover-enabled
```

Output

```
{
    "GlobalReplicationGroup": {
        "GlobalReplicationGroupId": "sgaui-test-group",
        "GlobalReplicationGroupDescription": " ",
        "Status": "modifying",
        "CacheNodeType": "cache.r5.large",
        "Engine": "redis",
        "EngineVersion": "5.0.6",
        "ClusterEnabled": false,
        "AuthTokenEnabled": false,
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用全局数据存储跨 Amazon 区域进行复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyGlobalReplicationGroup。

modify-replication-group-shard-configuration

以下代码示例演示了如何使用 modify-replication-group-shard-configuration。

Amazon CLI

修改复制组分区配置

以下 modify-replication-group-shard-configuration 使用 Redis 引擎减少节点组计数。

```
aws elasticache modify-replication-group-shard-configuration \
    --replication-group-id mycluster \
    --node-group-count 3 \
    --apply-immediately \
    --node-groups-to-remove 0002
```

Output


```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "mycluster",
        "Description": "mycluster",
        "GlobalReplicationGroupInfo": {},
        "Status": "modifying",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "mycluster-0002-001",
            "mycluster-0002-002",
            "mycluster-0002-003",
            "mycluster-0003-001",
            "mycluster-0003-002",
            "mycluster-0003-003",
            "mycluster-0003-004",
            "mycluster-0004-001",
            "mycluster-0004-002",
            "mycluster-0004-003",
            "mycluster-0005-001",
            "mycluster-0005-002",
            "mycluster-0005-003"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0002",
                "Status": "modifying",
                "Slots": "894-1767,3134-4443,5149-5461,6827-7332,12570-13662",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0002-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    },
                    {
                        "CacheClusterId": "mycluster-0002-002",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2a"
                    },
                    {
                        "CacheClusterId": "mycluster-0002-003",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
```

```
                    }
                ]
            },
            {
                "NodeGroupId": "0003",
                "Status": "modifying",
                "Slots":
"0-324,5462-5692,6784-6826,7698-8191,10923-11075,12441-12569,13663-16383",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0003-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    },
                    {
                        "CacheClusterId": "mycluster-0003-002",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
                    },
                    {
                        "CacheClusterId": "mycluster-0003-003",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2a"
                    },
                    {
                        "CacheClusterId": "mycluster-0003-004",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    }
                ]
            },
            {
                "NodeGroupId": "0004",
                "Status": "modifying",
                "Slots": "325-336,4706-5148,7333-7697,9012-10922,11076-12440",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0004-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
                    },
                    {
                        "CacheClusterId": "mycluster-0004-002",
                        "CacheNodeId": "0001",
```

```
                    "PreferredAvailabilityZone": "us-west-2a"
                },
                {
                    "CacheClusterId": "mycluster-0004-003",
                    "CacheNodeId": "0001",
                    "PreferredAvailabilityZone": "us-west-2c"
                }
            ]
        },
        {
            "NodeGroupId": "0005",
            "Status": "modifying",
            "Slots": "337-893,1768-3133,4444-4705,5693-6783,8192-9011",
            "NodeGroupMembers": [
                {
                    "CacheClusterId": "mycluster-0005-001",
                    "CacheNodeId": "0001",
                    "PreferredAvailabilityZone": "us-west-2a"
                },
                {
                    "CacheClusterId": "mycluster-0005-002",
                    "CacheNodeId": "0001",
                    "PreferredAvailabilityZone": "us-west-2c"
                },
                {
                    "CacheClusterId": "mycluster-0005-003",
                    "CacheNodeId": "0001",
                    "PreferredAvailabilityZone": "us-west-2b"
                }
            ]
        }
    ],
    "AutomaticFailover": "enabled",
    "MultiAZ": "enabled",
    "ConfigurationEndpoint": {
        "Address": "mycluster.g2xbih.clustercfg.usw2.cache.amazonaws.com",
        "Port": 6379
    },
    "SnapshotRetentionLimit": 1,
    "SnapshotWindow": "13:00-14:00",
    "ClusterEnabled": true,
    "CacheNodeType": "cache.r5.xlarge",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
```

```
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的扩展 ElastiCache for Redis 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReplicationGroupShardConfiguration。

`modify-replication-group`

以下代码示例演示了如何使用 `modify-replication-group`。

Amazon CLI

修改复制组

以下 `modify-replication-group` 使用 Redis 引擎禁用多可用区。

```
aws elasticache modify-replication-group \
    --replication-group-id test-cluster \
    --no-multi-az-enabled \
    --apply-immediately
```

Output

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "test-cluster",
        "Description": "test-cluster",
        "GlobalReplicationGroupInfo": {
            "GlobalReplicationGroupId": "sgaui-pat-group",
            "GlobalReplicationGroupMemberRole": "PRIMARY"
        },
        "Status": "available",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "test-cluster-001",
            "test-cluster-002",
            "test-cluster-003"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
```

```
                "Status": "available",
                "PrimaryEndpoint": {
                    "Address": "test-
cluster.g2xbih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "test-cluster-
ro.g2xbih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "test-cluster-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "test-
cluster-001.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "test-cluster-002",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "test-
cluster-002.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "test-cluster-003",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "test-
cluster-003.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "replica"
```

```
                    }
                ]
            }
        ],
        "SnapshottingClusterId": "test-cluster-002",
        "AutomaticFailover": "enabled",
        "MultiAZ": "disabled",
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "08:00-09:00",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.large",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的修改复制组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyReplicationGroup。

modify-user-group

以下代码示例演示了如何使用 modify-user-group。

Amazon CLI

修改用户组

以下 modify-user-group 示例将用户添加到用户组。

```
aws elasticache modify-user-group \
    --user-group-id myusergroup \
    --user-ids-to-add user1
```

输出：

```
{
    "UserGroupId": "myusergroup",
    "Status": "modifying",
    "Engine": "redis",
    "UserIds": [
        "default"
```

```
    ],
    "PendingChanges": {
        "UserIdsToAdd": [
            "user1"
        ]
    },
    "ReplicationGroups": [],
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyUserGroup。

`modify-user`

以下代码示例演示了如何使用 `modify-user`。

Amazon CLI

修改用户

以下 `modify-user` 示例修改用户的访问字符串。

```
aws elasticache modify-user \
    --user-id user2 \
    --append-access-string "on ~* +@all"
```

输出：

```
{
    "UserId": "user2",
    "UserName": "myUser",
    "Status": "modifying",
    "Engine": "redis",
    "AccessString": "on ~* +@all",
    "UserGroupIds": [],
    "Authentication": {
        "Type": "password",
        "PasswordCount": 1
    },
```

```
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的使用基于角色的访问控制（RBAC）对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyUser。

purchase-reserved-cache-nodes-offering

以下代码示例演示了如何使用 purchase-reserved-cache-nodes-offering。

Amazon CLI

购买 reserved-cache-node-offering

以下 purchase-reserved-cache-nodes-offering 示例允许您购买预留缓存节点产品。

```
aws elasticache purchase-reserved-cache-nodes-offering \
    --reserved-cache-nodes-offering-id xxxxxxx-4da5-4b90-b92d-929fbd7abed2
```

Output

```
{
    "ReservedCacheNode": {
        "ReservedCacheNodeId": "ri-2020-06-30-17-59-40-474",
        "ReservedCacheNodesOfferingId": "xxxxxxx-4da5-4b90-b92d-929fbd7abed2",
        "CacheNodeType": "cache.m3.2xlarge",
        "StartTime": "2020-06-30T17:59:40.474000+00:00",
        "Duration": 31536000,
        "FixedPrice": 1772.0,
        "UsagePrice": 0.0,
        "CacheNodeCount": 1,
        "ProductDescription": "redis",
        "OfferingType": "Heavy Utilization",
        "State": "payment-pending",
        "RecurringCharges": [
            {
                "RecurringChargeAmount": 0.25,
                "RecurringChargeFrequency": "Hourly"
            }
        ]
```

```
    }
}
```

有关更多信息，请参阅《ElastiCache Redis 用户指南》中的获取有关预留节点产品的信息，或《ElastiCache Memcached 用户指南》中的获取有关预留节点产品的信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseReservedCacheNodesOffering。

`reboot-cache-cluster`

以下代码示例演示了如何使用 `reboot-cache-cluster`。

Amazon CLI

重启缓存集群

以下 `reboot-cache-cluster` 示例重启预置集群中的部分或全部缓存节点。此操作将所有修改的缓存参数组应用于该集群。该重启操作会尽快进行，并导致集群短暂中断。在重启期间，集群状态设置为 REBOOTING。

```
aws elasticache reboot-cache-cluster \
    --cache-cluster-id "my-cluster-001" \
    --cache-node-ids-to-reboot "0001"
```

输出：

```
{
    "CacheCluster": {
        "CacheClusterId": "my-cluster-001",
        "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
        "CacheNodeType": "cache.r5.xlarge",
        "Engine": "redis",
        "EngineVersion": "5.0.5",
        "CacheClusterStatus": "rebooting cache cluster nodes",
        "NumCacheNodes": 1,
        "PreferredAvailabilityZone": "us-west-2a",
        "CacheClusterCreateTime": "2019-11-26T03:35:04.546Z",
        "PreferredMaintenanceWindow": "mon:04:05-mon:05:05",
        "PendingModifiedValues": {},
```

```
        "NotificationConfiguration": {
            "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxx152:My_Topic",
            "TopicStatus": "active"
        },
        "CacheSecurityGroups": [],
        "CacheParameterGroup": {
            "CacheParameterGroupName": "mygroup",
            "ParameterApplyStatus": "in-sync",
            "CacheNodeIdsToReboot": []
        },
        "CacheSubnetGroupName": "kxkxk",
        "AutoMinorVersionUpgrade": true,
        "SecurityGroups": [
            {
                "SecurityGroupId": "sg-xxxxxxxxxxxxx836",
                "Status": "active"
            },
            {
                "SecurityGroupId": "sg-xxxxxxxx7b",
                "Status": "active"
            }
        ],
        "ReplicationGroupId": "my-cluster",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "07:30-08:30",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的“重启集群”<https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/Clusters.Rebooting.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootCacheCluster。

`reset-cache-parameter-group`

以下代码示例演示了如何使用 `reset-cache-parameter-group`。

Amazon CLI

重置缓存参数组

以下 reset-cache-parameter-group 示例将缓存参数组的参数修改为引擎或系统默认值。您可以通过提交参数名称列表来重置特定参数。要重置整个缓存参数组，请指定 --reset-all-parameters 和 --cache-parameter-group-name 参数。

```
aws elasticache reset-cache-parameter-group \
    --cache-parameter-group-name "mygroup" \
    --reset-all-parameters
```

输出：

```
{
    "CacheParameterGroupName": "mygroup"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetCacheParameterGroup。

start-migration

以下代码示例演示了如何使用 start-migration。

Amazon CLI

开始迁移

以下 start-migration 使用 Redis 引擎将数据从 Amazon EC2 上的自托管 Redis 迁移到 Amazon ElastiCache。

```
aws elasticache start-migration \
   --replication-group-id test \
   --customer-node-endpoint-
list "Address='test.g2xbih.ng.0001.usw2.cache.amazonaws.com',Port=6379"
```

Output

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "test",
        "Description": "test",
        "GlobalReplicationGroupInfo": {},
        "Status": "modifying",
        "PendingModifiedValues": {},
```

```
        "MemberClusters": [
            "test-001",
            "test-002",
            "test-003"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
                "Status": "available",
                "PrimaryEndpoint": {
                    "Address": "test.g2xbih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "test-ro.g2xbih.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "test-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address":
"test-001.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "test-002",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address":
"test-002.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "test-003",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
```

```
                            "Address":
 "test-003.g2xbih.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    }
                ]
            }
        ],
        "SnapshottingClusterId": "test-002",
        "AutomaticFailover": "enabled",
        "MultiAZ": "enabled",
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "07:30-08:30",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.large",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

有关更多信息，请参阅《ElastiCache 用户指南》中的联机迁移到 ElastiCache。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartMigration。

test-failover

以下代码示例演示了如何使用 test-failover。

Amazon CLI

测试节点组的失效转移

以下 test-failover 示例在复制组（在控制台中称为集群）中的指定节点组（在控制台中称为分区）上测试自动失效转移。

```
aws elasticache test-failover /
    --replication-group-id "mycluster" /
    --node-group-id "0001"
```

输出：

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "mycluster",
        "Description": "My Cluster",
        "Status": "available",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "mycluster-0001-001",
            "mycluster-0001-002",
            "mycluster-0001-003",
            "mycluster-0002-001",
            "mycluster-0002-002",
            "mycluster-0002-003",
            "mycluster-0003-001",
            "mycluster-0003-002",
            "mycluster-0003-003"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
                "Status": "available",
                "Slots": "0-5461",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0001-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
                    },
                    {
                        "CacheClusterId": "mycluster-0001-002",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2a"
                    },
                    {
                        "CacheClusterId": "mycluster-0001-003",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    }
                ]
            },
            {
                "NodeGroupId": "0002",
                "Status": "available",
```

```
                "Slots": "5462-10922",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0002-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2a"
                    },
                    {
                        "CacheClusterId": "mycluster-0002-002",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
                    },
                    {
                        "CacheClusterId": "mycluster-0002-003",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    }
                ]
            },
            {
                "NodeGroupId": "0003",
                "Status": "available",
                "Slots": "10923-16383",
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "mycluster-0003-001",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2c"
                    },
                    {
                        "CacheClusterId": "mycluster-0003-002",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2b"
                    },
                    {
                        "CacheClusterId": "mycluster-0003-003",
                        "CacheNodeId": "0001",
                        "PreferredAvailabilityZone": "us-west-2a"
                    }
                ]
            }
        ],
        "AutomaticFailover": "enabled",
        "ConfigurationEndpoint": {
```

```
            "Address": "mycluster.xxxxih.clustercfg.usw2.cache.amazonaws.com",
            "Port": 6379
        },
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "13:00-14:00",
        "ClusterEnabled": true,
        "CacheNodeType": "cache.r5.large",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestFailover。

使用 Amazon CLI 的 MediaStore 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 MediaConvert 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-container`

以下代码示例演示了如何使用 `create-container`。

Amazon CLI

创建容器

以下 `create-container` 示例创建一个新的空容器。

```
aws mediastore create-container --container-name ExampleContainer
```

输出：

```
{
    "Container": {
        "AccessLoggingEnabled": false,
        "CreationTime": 1563557265,
        "Name": "ExampleContainer",
        "Status": "CREATING",
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的创建容器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateContainer。

`delete-container-policy`

以下代码示例演示了如何使用 `delete-container-policy`。

Amazon CLI

删除容器策略

以下 `delete-container-policy` 示例删除分配给指定容器的策略。删除该策略后，Amazon Elemental MediaStore 会自动将默认策略分配给容器。

```
aws mediastore delete-container-policy \
    --container-name LiveEvents
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore API 参考》中的 DeleteContainerPolicy。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteContainerPolicy。

`delete-container`

以下代码示例演示了如何使用 `delete-container`。

Amazon CLI

删除容器

以下 delete-container 示例删除指定容器。您只能在容器中没有对象时才能将其删除。

```
aws mediastore delete-container \
    --container-name=ExampleLiveDemo
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的删除容器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteContainer。

delete-cors-policy

以下代码示例演示了如何使用 delete-cors-policy。

Amazon CLI

删除 CORS 策略

以下 delete-cors-policy 示例删除分配给指定容器的跨源资源共享（CORS）策略。

```
aws mediastore delete-cors-policy \
    --container-name ExampleContainer
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的删除 CORS 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCorsPolicy。

delete-lifecycle-policy

以下代码示例演示了如何使用 delete-lifecycle-policy。

Amazon CLI

删除对象生命周期策略

以下 delete-lifecycle-policy 示例删除附加到指定容器的对象生命周期策略。此更改最多 20 分钟后生效。

```
aws mediastore delete-lifecycle-policy \
    --container-name LiveEvents
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的删除对象生命周期策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLifecyclePolicy。

describe-container

以下代码示例演示了如何使用 describe-container。

Amazon CLI

查看容器的详细信息

以下 describe-container 示例显示指定容器的详细信息。

```
aws mediastore describe-container \
    --container-name ExampleContainer
```

输出：

```
{
    "Container": {
        "CreationTime": 1563558086,
        "AccessLoggingEnabled": false,
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
        "Status": "ACTIVE",
        "Name": "ExampleContainer",
        "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看容器的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeContainer。

describe-object

以下代码示例演示了如何使用 describe-object。

Amazon CLI

查看特定容器中对象和文件夹的列表

以下 describe-object 示例显示存储在特定容器中的项目（对象和文件夹）的列表。

```
aws mediastore-data describe-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path /folder_name/file1234.jpg
```

输出：

```
{
    "ContentType": "image/jpeg",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ContentLength": "2307346",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看对象的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeObject。

get-container-policy

以下代码示例演示了如何使用 get-container-policy。

Amazon CLI

查看容器策略

以下 get-container-policy 示例显示指定容器的基于资源的策略。

```
aws mediastore get-container-policy \
    --container-name ExampleLiveDemo
```

输出：

```
{
```

```
    "Policy": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Sid": "PublicReadOverHttps",
                "Effect": "Allow",
                "Principal": {
                    "AWS": "arn:aws:iam::111122223333:root"
                },
                "Action": [
                    "mediastore:GetObject",
                    "mediastore:DescribeObject"
                ],
                "Resource": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleLiveDemo/",
                "Condition": {
                    "Bool": {
                        "aws:SecureTransport": "true"
                    }
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看容器策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContainerPolicy。

get-cors-policy

以下代码示例演示了如何使用 get-cors-policy。

Amazon CLI

查看 CORS 策略

以下 get-cors-policy 示例显示分配给指定容器的跨源资源共享（CORS）策略。

```
aws mediastore get-cors-policy \
    --container-name ExampleContainer \
    --region us-west-2
```

输出：

```
{
    "CorsPolicy": [
        {
            "AllowedMethods": [
                "GET",
                "HEAD"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedOrigins": [
                ""
            ],
            "AllowedHeaders": [
                ""
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看 CORS 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCorsPolicy。

get-lifecycle-policy

以下代码示例演示了如何使用 get-lifecycle-policy。

Amazon CLI

查看对象生命周期策略

以下 get-lifecycle-policy 示例显示附加到指定容器的对象生命周期策略。

```
aws mediastore get-lifecycle-policy \
    --container-name LiveEvents
```

输出：

```
{
    "LifecyclePolicy": {
        "rules": [
            {
```

```
                "definition": {
                    "path": [
                        {
                            "prefix": "Football/"
                        },
                        {
                            "prefix": "Baseball/"
                        }
                    ],
                    "days_since_create": [
                        {
                            "numeric": [
                                ">",
                                28
                            ]
                        }
                    ]
                },
                "action": "EXPIRE"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看对象生命周期策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLifecyclePolicy。

get-object

以下代码示例演示了如何使用 get-object。

Amazon CLI

下载对象

以下 get-object 示例将对象下载到指定端点。

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path=/folder_name/README.md README.md
```

输出：

```
{
    "ContentLength": "2307346",
    "ContentType": "image/jpeg",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "StatusCode": 200
}
```

下载对象的一部分

以下 get-object 示例将对象的一部分下载到指定端点。

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path /folder_name/README.md \
    --range="bytes=0-100" README2.md
```

输出：

```
{
    "StatusCode": 206,
    "ContentRange": "bytes 0-100/2307346",
    "ContentLength": "101",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ContentType": "image/jpeg",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的下载对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObject。

list-containers

以下代码示例演示了如何使用 list-containers。

Amazon CLI

查看容器列表

以下 list-containers 示例显示与您的账户关联的所有容器的列表。

```
aws mediastore list-containers
```

输出：

```
{
    "Containers": [
        {
            "CreationTime": 1505317931,
            "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-
west-2.amazonaws.com",
            "Status": "ACTIVE",
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleLiveDemo",
            "AccessLoggingEnabled": false,
            "Name": "ExampleLiveDemo"
        },
        {
            "CreationTime": 1506528818,
            "Endpoint": "https://fffggghhhiiijj.data.mediastore.us-
west-2.amazonaws.com",
            "Status": "ACTIVE",
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
            "AccessLoggingEnabled": false,
            "Name": "ExampleContainer"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看容器列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContainers。

list-items

以下代码示例演示了如何使用 list-items。

Amazon CLI

示例 1：查看特定容器中对象和文件夹的列表

以下 list-items 示例显示存储在指定容器中的项目（对象和文件夹）。

```
aws mediastore-data list-items \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com
```

输出：

```
{
    "Items": [
        {
            "ContentType": "image/jpeg",
            "LastModified": 1563571859.379,
            "Name": "filename.jpg",
            "Type": "OBJECT",
            "ETag":
 "543ab21abcd1a234ab123456a1a2b12345ab12abc12a1234abc1a2bc12345a12",
            "ContentLength": 3784
        },
        {
            "Type": "FOLDER",
            "Name": "ExampleLiveDemo"
        }
    ]
}
```

示例 2：查看特定文件夹中对象和文件夹的列表

以下 list-items 示例显示存储在特定文件夹中的项目（对象和文件夹）。

```
aws mediastore-data list-items \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com
```

输出：

```
{
    "Items": [
        {
            "ContentType": "image/jpeg",
            "LastModified": 1563571859.379,
            "Name": "filename.jpg",
            "Type": "OBJECT",
            "ETag":
 "543ab21abcd1a234ab123456a1a2b12345ab12abc12a1234abc1a2bc12345a12",
            "ContentLength": 3784
```

```
        },
        {
            "Type": "FOLDER",
            "Name": "ExampleLiveDemo"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看对象列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListItems。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出容器的标签

以下 `list-tags-for-resource` 示例显示分配给指定容器的标签键和值。

```
aws mediastore list-tags-for-resource \
    --resource arn:aws:mediastore:us-west-2:1213456789012:container/ExampleContainer
```

输出：

```
{
    "Tags": [
        {
            "Value": "Test",
            "Key": "Environment"
        },
        {
            "Value": "West",
            "Key": "Region"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore API 参考》中的 ListTagsForResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-container-policy

以下代码示例演示了如何使用 put-container-policy。

Amazon CLI

编辑容器策略

以下 put-container-policy 示例将不同的策略分配给指定容器。在此示例中，名为 LiveEventsContainerPolicy.json 的文件中定义了更新的策略。

```
aws mediastore put-container-policy \
    --container-name LiveEvents \
    --policy file://LiveEventsContainerPolicy.json
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的编辑容器策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutContainerPolicy。

put-cors-policy

以下代码示例演示了如何使用 put-cors-policy。

Amazon CLI

示例 1：添加 CORS 策略

以下 put-cors-policy 示例将跨源资源共享（CORS）策略添加到指定容器。CORS 策略的内容位于名为 corsPolicy.json 的文件中。

```
aws mediastore put-cors-policy \
    --container-name ExampleContainer \
    --cors-policy file://corsPolicy.json
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的将 CORS 策略添加到容器中。

示例 2：编辑 CORS 策略

以下 put-cors-policy 示例更新分配给指定容器的跨源资源共享（CORS）策略。更新的 CORS 策略的内容位于名为 corsPolicy2.json 的文件中。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的编辑 CORS 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutCorsPolicy。

put-lifecycle-policy

以下代码示例演示了如何使用 put-lifecycle-policy。

Amazon CLI

创建对象生命周期策略

以下 put-lifecycle-policy 示例将对象生命周期策略附加到指定容器。这使您可以指定该服务应在容器中存储对象多长时间。名为 LiveEventsLifecyclePolicy.json 的文件中的策略指示了对象的到期日，一旦对象到期，MediaStore 就会删除容器中的对象。

```
aws mediastore put-lifecycle-policy \
    --container-name ExampleContainer \
    --lifecycle-policy file://ExampleLifecyclePolicy.json
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的将对象生命周期策略添加到容器中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLifecyclePolicy。

put-object

以下代码示例演示了如何使用 put-object。

Amazon CLI

上传对象

以下 put-object 示例将对象上传到指定容器。您可以指定在容器中保存对象的文件夹路径。如果该文件夹已存在，则 Amazon Elemental MediaStore 会将对象存储在该文件夹中。如果该文件夹不存在，则该服务将创建一个文件夹，然后将对象存储在其中。

```
aws mediastore-data put-object \
```

```
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --body README.md \
    --path /folder_name/README.md \
    --cache-control "max-age=6, public" \
    --content-type binary/octet-stream
```

输出：

```
{
    "ContentSHA256":
 "74b5fdb517f423ed750ef214c44adfe2be36e37d861eafe9c842cbe1bf387a9d",
    "StorageClass": "TEMPORAL",
    "ETag": "af3e4731af032167a106015d1f2fe934e68b32ed1aa297a9e325f5c64979277b"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的上传对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObject。

start-access-logging

以下代码示例演示了如何使用 start-access-logging。

Amazon CLI

在容器上启用访问日志记录

以下 start-access-logging 示例在指定容器上启用访问日志记录。

```
aws mediastore start-access-logging \
    --container-name LiveEvents
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的对容器启用访问日志记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartAccessLogging。

stop-access-logging

以下代码示例演示了如何使用 stop-access-logging。

Amazon CLI

禁用容器上的访问日志记录

以下 stop-access-logging 示例在指定容器上禁用访问日志记录。

```
aws mediastore stop-access-logging \
    --container-name LiveEvents
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的对容器禁用访问日志记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopAccessLogging。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向容器添加标签

以下 tag-resource 示例将标签键和值添加到指定容器。

```
aws mediastore tag-resource \
    --resource arn:aws:mediastore:us-west-2:123456789012:container/ExampleContainer
 \
    --tags '[{"Key": "Region", "Value": "West"}, {"Key": "Environment", "Value":
 "Test"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore API 参考》中的 TagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从容器中移除标签

以下 `untag-resource` 示例从容器中移除指定标签键及其关联值。

```
aws mediastore untag-resource \
    --resource arn:aws:mediastore:us-west-2:123456789012:container/ExampleContainer
 \
    --tag-keys Region
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore API 参考》中的 UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Amazon EMR 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 Amazon EMR 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-instance-fleet`

以下代码示例演示了如何使用 `add-instance-fleet`。

Amazon CLI

将任务实例集添加到集群

此示例将新的任务实例集添加到指定的集群中。

命令:

```
aws emr add-instance-fleet --cluster-id 'j-12ABCDEFGHI34JK' --instance-fleet
 InstanceFleetType=TASK,TargetSpotCapacity=1,LaunchSpecifications={SpotSpecification='{Timeo
```

输出：

```
{
   "ClusterId": "j-12ABCDEFGHI34JK",
   "InstanceFleetId": "if-23ABCDEFGHI45JJ"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddInstanceFleet。

add-steps

以下代码示例演示了如何使用 add-steps。

Amazon CLI

1. 将自定义 JAR 步骤添加到集群

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps
 Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://amzn-s3-demo-
bucket/mytest.jar,Args=arg1,arg2,arg3
 Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://amzn-s3-demo-
bucket/mytest.jar,MainClass=mymainclass,Args=arg1,arg2,arg3
```

必需参数：

```
Jar
```

可选参数：

```
Type, Name, ActionOnFailure, Args
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
        "s-YYYYYYYY"
    ]
}
```

2. 将流式处理步骤添加到集群

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=STREAMING,Name='Streaming
 Program',ActionOnFailure=CONTINUE,Args=[-files,s3://elasticmapreduce/samples/
wordcount/wordSplitter.py,-mapper,wordSplitter.py,-reducer,aggregate,-input,s3://
elasticmapreduce/samples/wordcount/input,-output,s3://amzn-s3-demo-bucket/wordcount/
output]
```

必需参数：

```
Type, Args
```

可选参数：

```
Name, ActionOnFailure
```

JSON 等效值（step.json 的内容）：

```
 [
  {
    "Name": "JSON Streaming Step",
    "Args": ["-files","s3://elasticmapreduce/samples/wordcount/wordSplitter.py","-
mapper","wordSplitter.py","-reducer","aggregate","-input","s3://elasticmapreduce/
samples/wordcount/input","-output","s3://amzn-s3-demo-bucket/wordcount/output"],
    "ActionOnFailure": "CONTINUE",
    "Type": "STREAMING"
  }
]
```

注意：JSON 参数必须将选项和值作为其各自的项目包含在列表中。

命令（使用 step.json）：

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps file://./step.json
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
        "s-YYYYYYYY"
    ]
}
```

3. 将包含多个文件的流式处理步骤添加到集群（仅限 JSON）

JSON（multiplefiles.json）：

```
[
  {
     "Name": "JSON Streaming Step",
     "Type": "STREAMING",
     "ActionOnFailure": "CONTINUE",
     "Args": [
         "-files",
         "s3://amzn-s3-demo-bucket/mapper.py,s3://amzn-s3-demo-bucket/reducer.py",
         "-mapper",
         "mapper.py",
         "-reducer",
         "reducer.py",
         "-input",
         "s3://amzn-s3-demo-bucket/input",
         "-output",
         "s3://amzn-s3-demo-bucket/output"]
  }
]
```

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX  --steps file://./multiplefiles.json
```

必需参数：

```
Type, Args
```

可选参数：

```
Name, ActionOnFailure
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
    ]
}
```

4. 将 Hive 步骤添加到集群

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=HIVE,Name='Hive
 program',ActionOnFailure=CONTINUE,Args=[-f,s3://amzn-s3-demo-bucket/
myhivescript.q,-d,INPUT=s3://amzn-s3-demo-bucket/myhiveinput,-d,OUTPUT=s3://
amzn-s3-demo-bucket/myhiveoutput,arg1,arg2] Type=HIVE,Name='Hive
 steps',ActionOnFailure=TERMINATE_CLUSTER,Args=[-f,s3://elasticmapreduce/samples/
hive-ads/libs/model-build.q,-d,INPUT=s3://elasticmapreduce/samples/hive-ads/tables,-
d,OUTPUT=s3://amzn-s3-demo-bucket/hive-ads/output/2014-04-18/11-07-32,-d,LIBS=s3://
elasticmapreduce/samples/hive-ads/libs]
```

必需参数：

```
Type, Args
```

可选参数：

```
Name, ActionOnFailure
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
        "s-YYYYYYYY"
    ]
}
```

5. 将 Pig 步骤添加到集群

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=PIG,Name='Pig
 program',ActionOnFailure=CONTINUE,Args=[-f,s3://amzn-s3-demo-bucket/
mypigscript.pig,-p,INPUT=s3://amzn-s3-demo-bucket/mypiginput,-p,OUTPUT=s3://
amzn-s3-demo-bucket/mypigoutput,arg1,arg2] Type=PIG,Name='Pig program',Args=[-
f,s3://elasticmapreduce/samples/pig-apache/do-reports2.pig,-p,INPUT=s3://
elasticmapreduce/samples/pig-apache/input,-p,OUTPUT=s3://amzn-s3-demo-bucket/pig-
apache/output,arg1,arg2]
```

必需参数：

```
Type, Args
```

可选参数：

```
Name, ActionOnFailure
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
        "s-YYYYYYYY"
    ]
}
```

6. 将 Impala 步骤添加到集群

命令:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=IMPALA,Name='Impala
 program',ActionOnFailure=CONTINUE,Args=--impala-script,s3://myimpala/input,--
console-output-path,s3://myimpala/output
```

必需参数：

```
Type, Args
```

可选参数：

```
Name, ActionOnFailure
```

输出：

```
{
    "StepIds":[
        "s-XXXXXXXX",
        "s-YYYYYYYY"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddSteps。

add-tags

以下代码示例演示了如何使用 add-tags。

Amazon CLI

1. 将标签添加到集群

命令:

```
aws emr add-tags --resource-id j-xxxxxxx --tags name="John Doe" age=29 sex=male
 address="123 East NW Seattle"
```

输出：

```
None
```

2. 列出集群的标签

--命令：

```
aws emr describe-cluster --cluster-id j-XXXXXXYY --query Cluster.Tags
```

输出：

```
[
    {
        "Value": "male",
        "Key": "sex"
    },
    {
        "Value": "123 East NW Seattle",
        "Key": "address"
    },
    {
        "Value": "John Doe",
        "Key": "name"
    },
    {
        "Value": "29",
        "Key": "age"
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTags。

create-cluster-examples

以下代码示例演示了如何使用 create-cluster-examples。

Amazon CLI

以下大多数示例均假定您指定了 Amazon EMR 服务角色和 Amazon EC2 实例配置文件。如果您尚未执行此操作，则必须指定每个必需的 IAM 角色或在创建集群时使用 --use-default-roles 参数。有关指定 IAM 角色的更多信息，请参阅《Amazon EMR 管理指南》中的为 Amazon 服务的 Amazon EMR 权限配置 IAM 角色。

示例 1：创建集群

以下 create-cluster 示例将创建一个简单的 EMR 集群。

```
aws emr create-cluster \
    --release-label emr-5.14.0 \
    --instance-type m4.large \
    --instance-count 2
```

此命令不生成任何输出。

示例 2：创建具有默认 ServiceRole 和 InstanceProfile 角色的 Amazon EMR 集群

以下 create-cluster 示例创建使用 --instance-groups 配置的 Amazon EMR 集群。

```
aws emr create-cluster \
    --release-label emr-5.14.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

示例 3：创建使用实例集的 Amazon EMR 集群

以下 create-cluster 示例创建一个使用 --instance-fleets 配置的 Amazon EMR 集群，为每个实例集指定两个实例类型和两个 EC2 子网。

```
aws emr create-cluster \
    --release-label emr-5.14.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole,SubnetIds=['subnet-
ab12345c','subnet-de67890f'] \
    --instance-fleets
 InstanceFleetType=MASTER,TargetOnDemandCapacity=1,InstanceTypeConfigs=['{InstanceType=m4.la
 InstanceFleetType=CORE,TargetSpotCapacity=11,InstanceTypeConfigs=['{InstanceType=m4.large,B
```

示例 4：创建具有默认角色的集群

以下 create-cluster 示例使用 --use-default-roles 参数指定默认服务角色和实例配置文件。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --use-default-roles \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 5：创建集群并指定要安装的应用程序

以下 create-cluster 示例使用 --applications 参数指定 Amazon EMR 安装的应用程序。此示例安装了 Hadoop、Hive 和 Pig。

```
aws emr create-cluster \
    --applications Name=Hadoop Name=Hive Name=Pig \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 6：创建包含 Spark 的集群

以下示例安装 Spark。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --applications Name=Spark \
    --ec2-attributes KeyName=myKey \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 7：指定用于集群实例的自定义 AMI

以下 `create-cluster` 示例基于 ID 为 `ami-a518e6df` 的 Amazon Linux AMI 创建集群实例。

```
aws emr create-cluster \
    --name "Cluster with My Custom AMI" \
    --custom-ami-id ami-a518e6df \
    --ebs-root-volume-size 20 \
    --release-label emr-5.9.0 \
    --use-default-roles \
    --instance-count 2 \
    --instance-type m4.large
```

示例 8：自定义应用程序配置

以下示例使用 `--configurations` 参数指定包含 Hadoop 应用程序自定义项的 JSON 配置文件。有关更多信息，请参阅《Amazon EMR 发行版指南》中的配置应用程序。

的内容`configurations.json`：

```
[
```

```
    {
       "Classification": "mapred-site",
       "Properties": {
           "mapred.tasktracker.map.tasks.maximum": 2
       }
    },
    {
        "Classification": "hadoop-env",
        "Properties": {},
        "Configurations": [
            {
                "Classification": "export",
                "Properties": {
                    "HADOOP_DATANODE_HEAPSIZE": 2048,
                    "HADOOP_NAMENODE_OPTS": "-XX:GCTimeRatio=19"
                }
            }
        ]
    }
]
```

以下示例将 configurations.json 作为本地文件引用。

```
aws emr create-cluster \
    --configurations file://configurations.json \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

以下示例将 configurations.json 作为 Amazon S3 中的文件引用。

```
aws emr create-cluster \
    --configurations https://s3.amazonaws.com/amzn-s3-demo-bucket/
configurations.json \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 9：创建包含主实例组、核心实例组和任务实例组的集群

以下 create-cluster 示例使用 --instance-groups 指定要用于主实例组、核心实例组和任务实例组的 EC2 实例的类型和数量。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --instance-
groups Name=Master,InstanceGroupType=MASTER,InstanceType=m4.large,InstanceCount=1 Name=Core,
```

示例 10：指定集群应在完成所有步骤后终止

以下 create-cluster 示例使用 --auto-terminate 指定集群应在完成所有步骤后自动关闭。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
    --auto-terminate
```

示例 11：指定集群配置详细信息，例如 Amazon EC2 密钥对、网络配置和安全组

以下 create-cluster 示例使用名为 myKey 的 Amazon EC2 密钥对和名为 myProfile 的自定义实例配置文件创建一个集群。密钥对用于授权与集群节点（通常是主节点）的 SSH 连接。有关更多信息，请参阅《Amazon EMR 管理指南》中的对 SSH 凭证使用 Amazon EC2 密钥对。

```
aws emr create-cluster \
    --ec2-attributes KeyName=myKey,InstanceProfile=myProfile \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

以下示例在 Amazon VPC 子网中创建一个集群。

```
aws emr create-cluster \
    --ec2-attributes SubnetId=subnet-xxxxx \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
```

```
    --auto-terminate
```

以下示例在 us-east-1b 可用区中创建一个集群。

```
aws emr create-cluster \
    --ec2-attributes AvailabilityZone=us-east-1b \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下示例创建一个集群并仅指定 Amazon EMR 托管安全组。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role myServiceRole \
    --ec2-attributes InstanceProfile=myRole,EmrManagedMasterSecurityGroup=sg-
master1,EmrManagedSlaveSecurityGroup=sg-slave1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下示例创建一个集群并仅指定其他 Amazon EC2 安全组。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role myServiceRole \
    --ec2-attributes InstanceProfile=myRole,AdditionalMasterSecurityGroups=[sg-
addMaster1,sg-addMaster2,sg-addMaster3,sg-
addMaster4],AdditionalSlaveSecurityGroups=[sg-addSlave1,sg-addSlave2,sg-
addSlave3,sg-addSlave4] \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下示例创建一个集群并指定 EMR 托管安全组以及其他安全组。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role myServiceRole \
    --ec2-attributes InstanceProfile=myRole,EmrManagedMasterSecurityGroup=sg-
master1,EmrManagedSlaveSecurityGroup=sg-slave1,AdditionalMasterSecurityGroups=[sg-
addMaster1,sg-addMaster2,sg-addMaster3,sg-
addMaster4],AdditionalSlaveSecurityGroups=[sg-addSlave1,sg-addSlave2,sg-
addSlave3,sg-addSlave4] \
```

```
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下示例在 VPC 私有子网中创建一个集群，并使用特定的 Amazon EC2 安全组来启用 Amazon EMR 服务访问权限，这是私有子网中的集群所必需的。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role myServiceRole \
    --ec2-attributes InstanceProfile=myRole,ServiceAccessSecurityGroup=sg-service-
access,EmrManagedMasterSecurityGroup=sg-master,EmrManagedSlaveSecurityGroup=sg-slave
 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下示例使用存储在本地的名为 ec2_attributes.json 的 JSON 文件指定安全组配置参数。注意：JSON 参数必须将选项和值作为其各自的项目包含在列表中。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role myServiceRole \
    --ec2-attributes file://ec2_attributes.json  \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

的内容ec2_attributes.json：

```
[
    {
        "SubnetId": "subnet-xxxxx",
        "KeyName": "myKey",
        "InstanceProfile":"myRole",
        "EmrManagedMasterSecurityGroup": "sg-master1",
        "EmrManagedSlaveSecurityGroup": "sg-slave1",
        "ServiceAccessSecurityGroup": "sg-service-access",
        "AdditionalMasterSecurityGroups": ["sg-addMaster1","sg-addMaster2","sg-
addMaster3","sg-addMaster4"],
        "AdditionalSlaveSecurityGroups": ["sg-addSlave1","sg-addSlave2","sg-
addSlave3","sg-addSlave4"]
    }
]
```

示例 12：启用调试并指定日志 URI

以下 create-cluster 示例使用 --enable-debugging 参数，该参数允许您使用 Amazon EMR 控制台中的调试工具更轻松地查看日志文件。--enable-debugging 需要 --log-uri 参数。

```
aws emr create-cluster \
    --enable-debugging \
    --log-uri s3://amzn-s3-demo-bucket/myLog \
    --release-label emr-5.9.0 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 13：在创建集群时添加标签

标签是可以帮助您识别和管理集群的键值对。以下 create-cluster 示例使用 --tags 参数为集群创建三个标签，一个标签的键名称为 name，值为 Shirley Rodriguez，第二个标签的键名称为 age，值为 29，第三个标签的键名称为 department，值为 Analytics。

```
aws emr create-cluster \
    --tags name="Shirley Rodriguez" age=29 department="Analytics" \
    --release-label emr-5.32.0 \
    --instance-type m5.xlarge \
    --instance-count 3 \
    --use-default-roles
```

以下示例列出了应用于集群的标签。

```
aws emr describe-cluster \
    --cluster-id j-XXXXXXYY \
    --query Cluster.Tags
```

示例 14：使用启用加密和其他安全功能的安全配置

以下 create-cluster 示例使用 --security-configuration 参数指定 EMR 集群的安全配置。您可以在 Amazon EMR 版本 4.8.0 或更高版本中使用安全配置。

```
aws emr create-cluster \
    --instance-type m4.large \
    --release-label emr-5.9.0 \
```

```
    --security-configuration mySecurityConfiguration
```

示例 15：创建具有为实例组配置的额外 EBS 存储卷的集群

指定其他 EBS 卷时，需要以下参数：VolumeType、SizeInGB（如果指定了 EbsBlockDeviceConfigs）。

以下 create-cluster 示例创建了一个集群，其中多个 EBS 卷挂载到核心实例组中的 EC2 实例。

```
aws emr create-cluster \
    --release-label emr-5.9.0  \
    --use-default-roles \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=d2.xlarge
 'InstanceGroupType=CORE,InstanceCount=2,InstanceType=d2.xlarge,EbsConfiguration={EbsOptimiz
{VolumeSpecification={VolumeType=io1,SizeInGB=100,Iops=100},VolumesPerInstance=4}]}'
 \
    --auto-terminate
```

以下示例创建了一个集群，其中多个 EBS 卷挂载到主实例组中的 EC2 实例。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --use-default-roles \
    --instance-groups 'InstanceGroupType=MASTER, InstanceCount=1,
 InstanceType=d2.xlarge, EbsConfiguration={EbsOptimized=true,
 EbsBlockDeviceConfigs=[{VolumeSpecification={VolumeType=io1, SizeInGB=100,
 Iops=100}},
{VolumeSpecification={VolumeType=standard,SizeInGB=50},VolumesPerInstance=3}]}' InstanceGrou
 \
    --auto-terminate
```

示例 16：创建包含自动扩缩策略的集群

您可以使用 Amazon EMR 版本 4.0 和更高版本将自动扩缩策略附加到核心实例组和任务实例组。自动扩缩策略会动态添加和删除 EC2 实例，以响应 Amazon CloudWatch 指标。有关更多信息，请参阅《Amazon EMR Management Guide》中的“在 Amazon EMR 中使用自动扩缩”<https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-automatic-scaling.html>。

附加自动扩缩策略时，您还必须使用 --auto-scaling-role EMR_AutoScaling_DefaultRole 为自动扩缩指定默认角色。

以下 create-cluster 示例使用具有嵌入式 JSON 结构的 AutoScalingPolicy 参数指定 CORE 实例组的自动扩缩策略，该参数指定了扩缩策略配置。具有嵌入式 JSON 结构的实例组必须将整个参数集合用单引号引起来。对于没有嵌入式 JSON 结构的实例组，可选择使用单引号。

```
aws emr create-cluster
    --release-label emr-5.9.0 \
    --use-default-roles --auto-scaling-role EMR_AutoScaling_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceType=d2.xlarge,InstanceCount=1
 'InstanceGroupType=CORE,InstanceType=d2.xlarge,InstanceCount=2,AutoScalingPolicy={Constrain
```

以下示例使用 JSON 文件 instancegroupconfig.json 来指定集群中所有实例组的配置。该 JSON 文件指定核心实例组的自动扩缩策略配置。

```
aws emr create-cluster \
    --release-label emr-5.9.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-groups file://myfolder/instancegroupconfig.json \
    --auto-scaling-role EMR_AutoScaling_DefaultRole
```

的内容instancegroupconfig.json：

```
[
    {
        "InstanceCount": 1,
        "Name": "MyMasterIG",
        "InstanceGroupType": "MASTER",
        "InstanceType": "m4.large"
    },
    {
        "InstanceCount": 2,
        "Name": "MyCoreIG",
        "InstanceGroupType": "CORE",
        "InstanceType": "m4.large",
        "AutoScalingPolicy": {
            "Constraints": {
                "MinCapacity": 2,
                "MaxCapacity": 10
            },
            "Rules": [
                {
```

```
                    "Name": "Default-scale-out",
                    "Description": "Replicates the default scale-out rule in the
 console for YARN memory.",
                    "Action": {
                        "SimpleScalingPolicyConfiguration": {
                            "AdjustmentType": "CHANGE_IN_CAPACITY",
                            "ScalingAdjustment": 1,
                            "CoolDown": 300
                        }
                    },
                    "Trigger": {
                        "CloudWatchAlarmDefinition": {
                            "ComparisonOperator": "LESS_THAN",
                            "EvaluationPeriods": 1,
                            "MetricName": "YARNMemoryAvailablePercentage",
                            "Namespace": "AWS/ElasticMapReduce",
                            "Period": 300,
                            "Threshold": 15,
                            "Statistic": "AVERAGE",
                            "Unit": "PERCENT",
                            "Dimensions": [
                                {
                                    "Key": "JobFlowId",
                                    "Value": "${emr.clusterId}"
                                }
                            ]
                        }
                    }
                }
            ]
        }
    }
]
```

示例 17：在创建集群时添加自定义 JAR 步骤

以下 `create-cluster` 示例通过指定存储在 Amazon S3 中的 JAR 文件来添加步骤。步骤向集群提交工作。JAR 文件中定义的主函数将在预置 EC2 实例、执行任何引导操作并安装应用程序之后执行。这些步骤是使用 `Type=CUSTOM_JAR` 指定的。

自定义 JAR 步骤需要 `Jar=` 参数，该参数可指定 JAR 的路径和文件名称。可选参数为 `Type`、`Name`、`ActionOnFailure`、`Args` 和 `MainClass`。如果未指定主类，则 JAR 文件应在其清单文件中指定 `Main-Class`。

```
aws emr create-cluster \
    --steps Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://amzn-
s3-demo-bucket/
mytest.jar,Args=arg1,arg2,arg3 Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s
amzn-s3-demo-bucket/mytest.jar,MainClass=mymainclass,Args=arg1,arg2,arg3  \
    --release-label emr-5.3.1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 18：在创建集群时添加流式处理步骤

以下 create-cluster 示例将流式处理步骤添加到集群，该集群在所有步骤运行后终止。流式处理步骤需要参数 Type 和 Args。流式处理步骤可选参数为 Name 和 ActionOnFailure。

以下示例指定内联步骤。

```
aws emr create-cluster \
    --steps Type=STREAMING,Name='Streaming Program',ActionOnFailure=CONTINUE,Args=[-
files,s3://elasticmapreduce/samples/wordcount/wordSplitter.py,-
mapper,wordSplitter.py,-reducer,aggregate,-input,s3://elasticmapreduce/samples/
wordcount/input,-output,s3://amzn-s3-demo-bucket/wordcount/output] \
    --release-label emr-5.3.1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

以下示例使用名为 multiplefiles.json 的本地存储 JSON 配置文件。JSON 配置指定多个文件。要在一个步骤中指定多个文件，必须使用 JSON 配置文件来指定该步骤。JSON 参数必须将选项和值作为其各自的项目包含在列表中。

```
aws emr create-cluster \
    --steps file://./multiplefiles.json \
    --release-label emr-5.9.0  \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

的内容multiplefiles.json：

```
[
    {
        "Name": "JSON Streaming Step",
        "Args": [
            "-files",
            "s3://elasticmapreduce/samples/wordcount/wordSplitter.py",
            "-mapper",
            "wordSplitter.py",
            "-reducer",
            "aggregate",
            "-input",
            "s3://elasticmapreduce/samples/wordcount/input",
            "-output",
            "s3://amzn-s3-demo-bucket/wordcount/output"
        ],
        "ActionOnFailure": "CONTINUE",
        "Type": "STREAMING"
    }
]
```

示例 19：在创建集群时添加 Hive 步骤

以下示例在创建集群时添加 Hive 步骤。Hive 步骤需要参数 Type 和 Args。Hive 步骤可选参数为 Name 和 ActionOnFailure。

```
aws emr create-cluster \
    --steps Type=HIVE,Name='Hive
 program',ActionOnFailure=CONTINUE,ActionOnFailure=TERMINATE_CLUSTER,Args=[-
f,s3://elasticmapreduce/samples/hive-ads/libs/model-build.q,-d,INPUT=s3://
elasticmapreduce/samples/hive-ads/tables,-d,OUTPUT=s3://amzn-s3-demo-bucket/hive-
ads/output/2014-04-18/11-07-32,-d,LIBS=s3://elasticmapreduce/samples/hive-ads/libs]
 \
    --applications Name=Hive \
    --release-label emr-5.3.1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

示例 20：在创建集群时添加 Pig 步骤

以下示例在创建集群时添加 Pig 步骤。Pig 步骤必需的参数为 Type 和 Args。Pig 步骤可选参数为 Name 和 ActionOnFailure。

```
aws emr create-cluster \
    --steps Type=PIG,Name='Pig program',ActionOnFailure=CONTINUE,Args=[-f,s3://
elasticmapreduce/samples/pig-apache/do-reports2.pig,-p,INPUT=s3://elasticmapreduce/
samples/pig-apache/input,-p,OUTPUT=s3://amzn-s3-demo-bucket/pig-apache/output] \
    --applications Name=Pig \
    --release-label emr-5.3.1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

示例 21：添加引导操作

以下 create-cluster 示例运行两个引导操作，这些操作定义为存储在 Amazon S3 中的脚本。

```
aws emr create-cluster \
    --bootstrap-actions Path=s3://amzn-s3-demo-bucket/
myscript1,Name=BootstrapAction1,Args=[arg1,arg2] Path=s3://amzn-s3-demo-bucket/
myscript2,Name=BootstrapAction2,Args=[arg1,arg2] \
    --release-label emr-5.3.1 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-terminate
```

示例 22：启用 EMRFS 一致视图并自定义 RetryCount 和 RetryPeriod 设置

以下 create-cluster 示例指定 EMRFS 一致视图的重试计数和重试期。Consistent=true 是必需参数。

```
aws emr create-cluster \
    --instance-type m4.large \
    --release-label emr-5.9.0 \
    --emrfs Consistent=true,RetryCount=6,RetryPeriod=30
```

以下示例使用名为 emrfsconfig.json 的本地存储 JSON 配置文件指定与上一个示例相同的 EMRFS 配置。

```
aws emr create-cluster \
    --instance-type m4.large \
    --release-label emr-5.9.0 \
    --emrfs file://emrfsconfig.json
```

的内容emrfsconfig.json：

```
{
    "Consistent": true,
    "RetryCount": 6,
    "RetryPeriod": 30
}
```

示例 23：创建配置了 Kerberos 的集群

以下 create-cluster 示例使用启用了 Kerberos 的安全配置创建集群，并使用 --kerberos-attributes 为该集群建立 Kerberos 参数。

以下命令指定集群内联的 Kerberos 属性。

```
aws emr create-cluster \
    --instance-type m3.xlarge \
    --release-label emr-5.10.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --security-configuration mySecurityConfiguration \
    --kerberos-
attributes Realm=EC2.INTERNAL,KdcAdminPassword=123,CrossRealmTrustPrincipalPassword=123
```

以下命令指定相同的属性，但引用名为 kerberos_attributes.json 的本地存储 JSON 文件。在此示例中，此文件保存在您运行该命令的同一目录中。您也可以引用保存在 Amazon S3 中的配置文件。

```
aws emr create-cluster \
    --instance-type m3.xlarge \
    --release-label emr-5.10.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --security-configuration mySecurityConfiguration \
    --kerberos-attributes file://kerberos_attributes.json
```

的内容kerberos_attributes.json：

```
{
    "Realm": "EC2.INTERNAL",
    "KdcAdminPassword": "123",
    "CrossRealmTrustPrincipalPassword": "123",
```

```
}
```

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用 --instance-groups 配置并具有托管扩缩策略。

```
aws emr create-cluster \
    --release-label emr-5.30.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
    --managed-scaling-policy
 ComputeLimits='{MinimumCapacityUnits=2,MaximumCapacityUnits=4,UnitType=Instances}'
```

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用“--log-encryption-kms-key-id”来定义用于日志加密的 KMS 密钥 ID。

```
aws emr create-cluster \
    --release-label emr-5.30.0 \
    --log-uri s3://amzn-s3-demo-bucket/myLog \
    --log-encryption-kms-key-id arn:aws:kms:us-east-1:110302272565:key/
dd559181-283e-45d7-99d1-66da348c4d33 \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用“--placement-group-configs”配置，利用 SPREAD 置放策略将主节点置于 EC2 置放组内的高可用性（HA）集群中。

```
aws emr create-cluster \
    --release-label emr-5.30.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=3,InstanceType=m4.largeInstanceGroupType=CORE,
 \
    --placement-group-configs InstanceRole=MASTER
```

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用“--auto-termination-policy”配置为该集群设定自动空闲终止阈值。

```
aws emr create-cluster \
```

```
    --release-label emr-5.34.0 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
 \
    --auto-termination-policy IdleTimeout=100
```

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用“--os-release-label”来定义用于集群启动的 Amazon Linux 发行版。

```
aws emr create-cluster \
    --release-label emr-6.6.0 \
    --os-release-label 2.0.20220406.1 \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
    --instance-
groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE
```

示例 24：指定 EBS 根卷属性：使用 EMR 发行版 6.15.0 及更高版本创建的集群实例的大小、IOPS 和吞吐量

以下 create-cluster 示例创建一个 Amazon EMR 集群，该集群使用根卷属性为 EC2 实例配置根卷规范。

```
aws emr create-cluster \
    --name "Cluster with My Custom AMI" \
    --custom-ami-id ami-a518e6df \
    --ebs-root-volume-size 20 \
    --ebs-root-volume-iops 3000 \
    --ebs-root-volume-throughput 125 \
    --release-label emr-6.15.0 \
    --use-default-roles \
    --instance-count 2 \
    --instance-type m4.large
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClusterExamples。

create-default-roles

以下代码示例演示了如何使用 create-default-roles。

Amazon CLI

1. 为 EC2 创建默认 IAM 角色

命令:

```
aws emr create-default-roles
```

输出：

```
If the role already exists then the command returns nothing.

If the role does not exist then the output will be:

[
    {
        "RolePolicy": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "cloudwatch:*",
                        "dynamodb:*",
                        "ec2:Describe*",
                        "elasticmapreduce:Describe*",
                        "elasticmapreduce:ListBootstrapActions",
                        "elasticmapreduce:ListClusters",
                        "elasticmapreduce:ListInstanceGroups",
                        "elasticmapreduce:ListInstances",
                        "elasticmapreduce:ListSteps",
                        "kinesis:CreateStream",
                        "kinesis:DeleteStream",
                        "kinesis:DescribeStream",
                        "kinesis:GetRecords",
                        "kinesis:GetShardIterator",
                        "kinesis:MergeShards",
                        "kinesis:PutRecord",
                        "kinesis:SplitShard",
                        "rds:Describe*",
                        "s3:*",
                        "sdb:*",
                        "sns:*",
                        "sqs:*"
```

```
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "Role": {
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Sid": "",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ec2.amazonaws.com"
                    }
                }
            ]
        },
        "RoleId": "AROAIQ5SIQUGL5KMYBJX6",
        "CreateDate": "2015-06-09T17:09:04.602Z",
        "RoleName": "EMR_EC2_DefaultRole",
        "Path": "/",
        "Arn": "arn:aws:iam::176430881729:role/EMR_EC2_DefaultRole"
    }
},
{
    "RolePolicy": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": [
                    "ec2:AuthorizeSecurityGroupIngress",
                    "ec2:CancelSpotInstanceRequests",
                    "ec2:CreateSecurityGroup",
                    "ec2:CreateTags",
                    "ec2:DeleteTags",
                    "ec2:DescribeAvailabilityZones",
                    "ec2:DescribeAccountAttributes",
                    "ec2:DescribeInstances",
                    "ec2:DescribeInstanceStatus",
                    "ec2:DescribeKeyPairs",
                    "ec2:DescribePrefixLists",
```

```
                    "ec2:DescribeRouteTables",
                    "ec2:DescribeSecurityGroups",
                    "ec2:DescribeSpotInstanceRequests",
                    "ec2:DescribeSpotPriceHistory",
                    "ec2:DescribeSubnets",
                    "ec2:DescribeVpcAttribute",
                    "ec2:DescribeVpcEndpoints",
                    "ec2:DescribeVpcEndpointServices",
                    "ec2:DescribeVpcs",
                    "ec2:ModifyImageAttribute",
                    "ec2:ModifyInstanceAttribute",
                    "ec2:RequestSpotInstances",
                    "ec2:RunInstances",
                    "ec2:TerminateInstances",
                    "iam:GetRole",
                    "iam:GetRolePolicy",
                    "iam:ListInstanceProfiles",
                    "iam:ListRolePolicies",
                    "iam:PassRole",
                    "s3:CreateBucket",
                    "s3:Get*",
                    "s3:List*",
                    "sdb:BatchPutAttributes",
                    "sdb:Select",
                    "sqs:CreateQueue",
                    "sqs:Delete*",
                    "sqs:GetQueue*",
                    "sqs:ReceiveMessage"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "Role": {
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Sid": "",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "elasticmapreduce.amazonaws.com"
```

```
                        }
                    }
                ]
            },
            "RoleId": "AROAI3SRVPPVSRDLARBPY",
            "CreateDate": "2015-06-09T17:09:10.401Z",
            "RoleName": "EMR_DefaultRole",
            "Path": "/",
            "Arn": "arn:aws:iam::176430881729:role/EMR_DefaultRole"
        }
    }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDefaultRoles。

`create-security-configuration`

以下代码示例演示了如何使用 `create-security-configuration`。

Amazon CLI

1. 创建安全配置，其中针对证书提供商，使用 PEM 启用传输中加密，并针对 S3 加密和本地磁盘密钥提供商，使用 SSE-S3 和 Amazon-KMS 启用静态加密

命令:

```
 aws emr create-security-configuration --name MySecurityConfig --security-
configuration '{
        "EncryptionConfiguration": {
                "EnableInTransitEncryption" : true,
                "EnableAtRestEncryption" : true,
                "InTransitEncryptionConfiguration" : {
                        "TLSCertificateConfiguration" : {
                                "CertificateProviderType" : "PEM",
                                "S3Object" : "s3://mycertstore/artifacts/
MyCerts.zip"
                        }
                },
                "AtRestEncryptionConfiguration" : {
                        "S3EncryptionConfiguration" : {
                                "EncryptionMode" : "SSE-S3"
                        },
```

```
                        "LocalDiskEncryptionConfiguration" : {
                                "EncryptionKeyProviderType" : "AwsKms",
                                "AwsKmsKey" : "arn:aws:kms:us-
east-1:123456789012:key/12345678-1234-1234-1234-123456789012"
                        }
                }
        }
}'
```

输出：

```
{
"CreationDateTime": 1474070889.129,
"Name": "MySecurityConfig"
}
```

JSON 等效值（security_configuration.json 的内容）：

```
{
    "EncryptionConfiguration": {
        "EnableInTransitEncryption": true,
        "EnableAtRestEncryption": true,
        "InTransitEncryptionConfiguration": {
            "TLSCertificateConfiguration": {
                "CertificateProviderType": "PEM",
                "S3Object": "s3://mycertstore/artifacts/MyCerts.zip"
            }
        },
        "AtRestEncryptionConfiguration": {
            "S3EncryptionConfiguration": {
                "EncryptionMode": "SSE-S3"
            },
            "LocalDiskEncryptionConfiguration": {
                "EncryptionKeyProviderType": "AwsKms",
                "AwsKmsKey": "arn:aws:kms:us-
east-1:123456789012:key/12345678-1234-1234-1234-123456789012"
            }
        }
    }
}
```

命令（使用 security_configuration.json）：

```
aws emr create-security-configuration --name "MySecurityConfig" --security-
configuration file://./security_configuration.json
```

输出：

```
{
"CreationDateTime": 1474070889.129,
"Name": "MySecurityConfig"
}
```

2. 使用集群专用 KDC 和跨领域信任创建启用了 Kerberos 的安全配置

命令:

```
 aws emr create-security-configuration --name MySecurityConfig --security-
configuration '{
     "AuthenticationConfiguration": {
         "KerberosConfiguration": {
             "Provider": "ClusterDedicatedKdc",
             "ClusterDedicatedKdcConfiguration": {
                 "TicketLifetimeInHours": 24,
                 "CrossRealmTrustConfiguration": {
                   "Realm": "AD.DOMAIN.COM",
                   "Domain": "ad.domain.com",
                   "AdminServer": "ad.domain.com",
                   "KdcServer": "ad.domain.com"
                 }
             }
         }
     }
}'
```

输出：

```
{
"CreationDateTime": 1490225558.982,
"Name": "MySecurityConfig"
}
```

JSON 等效值（security_configuration.json 的内容）：

```
{
```

```
    "AuthenticationConfiguration": {
        "KerberosConfiguration": {
            "Provider": "ClusterDedicatedKdc",
            "ClusterDedicatedKdcConfiguration": {
                "TicketLifetimeInHours": 24,
                "CrossRealmTrustConfiguration": {
                    "Realm": "AD.DOMAIN.COM",
                    "Domain": "ad.domain.com",
                    "AdminServer": "ad.domain.com",
                    "KdcServer": "ad.domain.com"
                }
            }
        }
    }
}
```

命令（使用 security_configuration.json）：

```
aws emr create-security-configuration --name "MySecurityConfig" --security-
configuration file://./security_configuration.json
```

输出：

```
{
"CreationDateTime": 1490225558.982,
"Name": "MySecurityConfig"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSecurityConfiguration。

`delete-security-configuration`

以下代码示例演示了如何使用 `delete-security-configuration`。

Amazon CLI

删除当前区域中的安全配置

命令:

```
aws emr delete-security-configuration --name MySecurityConfig
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSecurityConfiguration。

describe-cluster

以下代码示例演示了如何使用 describe-cluster。

Amazon CLI

命令:

```
aws emr describe-cluster --cluster-id j-XXXXXXXX
```

输出：

```
For release-label based uniform instance groups cluster:

        {
            "Cluster": {
                "Status": {
                    "Timeline": {
                        "ReadyDateTime": 1436475075.199,
                        "CreationDateTime": 1436474656.563,
                    },
                    "State": "WAITING",
                    "StateChangeReason": {
                        "Message": "Waiting for steps to run"
                    }
                },
                "Ec2InstanceAttributes": {
                    "ServiceAccessSecurityGroup": "sg-xxxxxxxx",
                    "EmrManagedMasterSecurityGroup": "sg-xxxxxxxx",
                    "IamInstanceProfile": "EMR_EC2_DefaultRole",
                    "Ec2KeyName": "myKey",
                    "Ec2AvailabilityZone": "us-east-1c",
                    "EmrManagedSlaveSecurityGroup": "sg-yyyyyyyyy"
                },
                "Name": "My Cluster",
                "ServiceRole": "EMR_DefaultRole",
```

```
                "Tags": [],
                "TerminationProtected": true,
                "UnhealthyNodeReplacement": true,
                "ReleaseLabel": "emr-4.0.0",
                "NormalizedInstanceHours": 96,
                "InstanceGroups": [
                    {
                        "RequestedInstanceCount": 2,
                        "Status": {
                            "Timeline": {
                                "ReadyDateTime": 1436475074.245,
                                "CreationDateTime": 1436474656.564,
                                "EndDateTime": 1436638158.387
                            },
                            "State": "RUNNING",
                            "StateChangeReason": {
                                "Message": "",
                            }
                        },
                        "Name": "CORE",
                        "InstanceGroupType": "CORE",
                        "Id": "ig-YYYYYYY",
                        "Configurations": [],
                        "InstanceType": "m3.large",
                        "Market": "ON_DEMAND",
                        "RunningInstanceCount": 2
                    },
                    {
                        "RequestedInstanceCount": 1,
                        "Status": {
                            "Timeline": {
                                "ReadyDateTime": 1436475074.245,
                                "CreationDateTime": 1436474656.564,
                                "EndDateTime": 1436638158.387
                            },
                            "State": "RUNNING",
                            "StateChangeReason": {
                                "Message": "",
                            }
                        },
                        "Name": "MASTER",
                        "InstanceGroupType": "MASTER",
                        "Id": "ig-XXXXXXXXX",
                        "Configurations": [],
```

```
                        "InstanceType": "m3.large",
                        "Market": "ON_DEMAND",
                        "RunningInstanceCount": 1
                    }
                ],
                "Applications": [
                    {
                        "Name": "Hadoop"
                    }
                ],
                "VisibleToAllUsers": true,
                "BootstrapActions": [],
                "MasterPublicDnsName": "ec2-54-147-144-78.compute-1.amazonaws.com",
                "AutoTerminate": false,
                "Id": "j-XXXXXXXX",
                "Configurations": [
                    {
                        "Properties": {
                            "fs.s3.consistent.retryPeriodSeconds": "20",
                            "fs.s3.enableServerSideEncryption": "true",
                            "fs.s3.consistent": "false",
                            "fs.s3.consistent.retryCount": "2"
                        },
                        "Classification": "emrfs-site"
                    }
                ]
            }
        }


For release-label based instance fleet cluster:
{
    "Cluster": {
        "Status": {
            "Timeline": {
                "ReadyDateTime": 1487897289.705,
                "CreationDateTime": 1487896933.942
            },
            "State": "WAITING",
            "StateChangeReason": {
                "Message": "Waiting for steps to run"
            }
        },
        "Ec2InstanceAttributes": {
```

```
            "EmrManagedMasterSecurityGroup": "sg-xxxxx",
            "RequestedEc2AvailabilityZones": [],
            "RequestedEc2SubnetIds": [],
            "IamInstanceProfile": "EMR_EC2_DefaultRole",
            "Ec2AvailabilityZone": "us-east-1a",
            "EmrManagedSlaveSecurityGroup": "sg-xxxxx"
        },
        "Name": "My Cluster",
        "ServiceRole": "EMR_DefaultRole",
        "Tags": [],
        "TerminationProtected": false,
        "UnhealthyNodeReplacement": false,
        "ReleaseLabel": "emr-5.2.0",
        "NormalizedInstanceHours": 472,
        "InstanceCollectionType": "INSTANCE_FLEET",
        "InstanceFleets": [
            {
                "Status": {
                    "Timeline": {
                        "ReadyDateTime": 1487897212.74,
                        "CreationDateTime": 1487896933.948
                    },
                    "State": "RUNNING",
                    "StateChangeReason": {
                        "Message": ""
                    }
                },
                "ProvisionedSpotCapacity": 1,
                "Name": "MASTER",
                "InstanceFleetType": "MASTER",
                "LaunchSpecifications": {
                    "SpotSpecification": {
                        "TimeoutDurationMinutes": 60,
                        "TimeoutAction": "TERMINATE_CLUSTER"
                    }
                },
                "TargetSpotCapacity": 1,
                "ProvisionedOnDemandCapacity": 0,
                "InstanceTypeSpecifications": [
                    {
                        "BidPrice": "0.5",
                        "InstanceType": "m3.xlarge",
                        "WeightedCapacity": 1
                    }
```

```
                ],
                "Id": "if-xxxxxxx",
                "TargetOnDemandCapacity": 0
            }
        ],
        "Applications": [
            {
                "Version": "2.7.3",
                "Name": "Hadoop"
            }
        ],
        "ScaleDownBehavior": "TERMINATE_AT_INSTANCE_HOUR",
        "VisibleToAllUsers": true,
        "BootstrapActions": [],
        "MasterPublicDnsName": "ec2-xxx-xx-xxx-xx.compute-1.amazonaws.com",
        "AutoTerminate": false,
        "Id": "j-xxxxx",
        "Configurations": []
    }
}

For ami based uniform instance group cluster:

    {
        "Cluster": {
            "Status": {
                "Timeline": {
                    "ReadyDateTime": 1399400564.432,
                    "CreationDateTime": 1399400268.62
                },
                "State": "WAITING",
                "StateChangeReason": {
                    "Message": "Waiting for steps to run"
                }
            },
            "Ec2InstanceAttributes": {
                "IamInstanceProfile": "EMR_EC2_DefaultRole",
                "Ec2AvailabilityZone": "us-east-1c"
            },
            "Name": "My Cluster",
            "Tags": [],
            "TerminationProtected": true,
            "UnhealthyNodeReplacement": true,
            "RunningAmiVersion": "2.5.4",
```

```
            "InstanceGroups": [
                {
                    "RequestedInstanceCount": 1,
                    "Status": {
                        "Timeline": {
                            "ReadyDateTime": 1399400558.848,
                            "CreationDateTime": 1399400268.621
                        },
                        "State": "RUNNING",
                        "StateChangeReason": {
                            "Message": ""
                        }
                    },
                    "Name": "Master instance group",
                    "InstanceGroupType": "MASTER",
                    "InstanceType": "m1.small",
                    "Id": "ig-ABCD",
                    "Market": "ON_DEMAND",
                    "RunningInstanceCount": 1
                },
                {
                    "RequestedInstanceCount": 2,
                    "Status": {
                        "Timeline": {
                            "ReadyDateTime": 1399400564.439,
                            "CreationDateTime": 1399400268.621
                        },
                        "State": "RUNNING",
                        "StateChangeReason": {
                            "Message": ""
                        }
                    },
                    "Name": "Core instance group",
                    "InstanceGroupType": "CORE",
                    "InstanceType": "m1.small",
                    "Id": "ig-DEF",
                    "Market": "ON_DEMAND",
                    "RunningInstanceCount": 2
                }
            ],
            "Applications": [
                {
                    "Version": "1.0.3",
                    "Name": "hadoop"
```

```
                }
            ],
            "BootstrapActions": [],
            "VisibleToAllUsers": false,
            "RequestedAmiVersion": "2.4.2",
            "LogUri": "s3://myLogUri/",
            "AutoTerminate": false,
            "Id": "j-XXXXXXXX"
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCluster。

describe-step

以下代码示例演示了如何使用 describe-step。

Amazon CLI

以下命令描述集群中步骤 ID 为 s-3LZC0QUT43AM 和集群 ID 为 j-3SD91U2E1L2QX 的步骤：

```
aws emr describe-step --cluster-id j-3SD91U2E1L2QX --step-id s-3LZC0QUT43AM
```

输出：

```
{
    "Step": {
        "Status": {
            "Timeline": {
                "EndDateTime": 1433200470.481,
                "CreationDateTime": 1433199926.597,
                "StartDateTime": 1433200404.959
            },
            "State": "COMPLETED",
            "StateChangeReason": {}
        },
        "Config": {
            "Args": [
                "s3://us-west-2.elasticmapreduce/libs/hive/hive-script",
                "--base-path",
                "s3://us-west-2.elasticmapreduce/libs/hive/",
                "--install-hive",
                "--hive-versions",
```

```
                "0.13.1"
            ],
            "Jar": "s3://us-west-2.elasticmapreduce/libs/script-runner/script-
runner.jar",
            "Properties": {}
        },
        "Id": "s-3LZC0QUT43AM",
        "ActionOnFailure": "TERMINATE_CLUSTER",
        "Name": "Setup hive"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStep。

get

以下代码示例演示了如何使用 get。

Amazon CLI

以下内容会从集群 ID 为 j-3SD91U2E1L2QX 的集群中的主实例中下载 hadoop-examples.jar 归档：

```
aws emr get --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem --src /
home/hadoop-examples.jar --dest ~
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Get。

list-clusters

以下代码示例演示了如何使用 list-clusters。

Amazon CLI

以下命令将列出当前区域中所有活动 EMR 集群：

```
aws emr list-clusters --active
```

输出：

```
{
```

```
    "Clusters": [
        {
            "Status": {
                "Timeline": {
                    "ReadyDateTime": 1433200405.353,
                    "CreationDateTime": 1433199926.596
                },
                "State": "WAITING",
                "StateChangeReason": {
                    "Message": "Waiting after step completed"
                }
            },
            "NormalizedInstanceHours": 6,
            "Id": "j-3SD91U2E1L2QX",
            "Name": "my-cluster"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListClusters。

`list-instance-fleets`

以下代码示例演示了如何使用 `list-instance-fleets`。

Amazon CLI

获取集群中的实例集的配置详细信息

该示例列出指定集群中的实例集的详细信息。

命令:

```
list-instance-fleets --cluster-id 'j-12ABCDEFGHI34JK'
```

输出：

```
{
  "InstanceFleets": [
      {
          "Status": {
              "Timeline": {
                  "ReadyDateTime": 1488759094.637,
```

```
                "CreationDateTime": 1488758719.817
            },
            "State": "RUNNING",
            "StateChangeReason": {
                "Message": ""
            }
        },
        "ProvisionedSpotCapacity": 6,
        "Name": "CORE",
        "InstanceFleetType": "CORE",
        "LaunchSpecifications": {
            "SpotSpecification": {
                "TimeoutDurationMinutes": 60,
                "TimeoutAction": "TERMINATE_CLUSTER"
            }
        },
        "ProvisionedOnDemandCapacity": 2,
        "InstanceTypeSpecifications": [
            {
                "BidPrice": "0.5",
                "InstanceType": "m3.xlarge",
                "WeightedCapacity": 2
            }
        ],
        "Id": "if-1ABC2DEFGHIJ3"
    },
    {
        "Status": {
            "Timeline": {
                "ReadyDateTime": 1488759058.598,
                "CreationDateTime": 1488758719.811
            },
            "State": "RUNNING",
            "StateChangeReason": {
                "Message": ""
            }
        },
        "ProvisionedSpotCapacity": 0,
        "Name": "MASTER",
        "InstanceFleetType": "MASTER",
        "ProvisionedOnDemandCapacity": 1,
        "InstanceTypeSpecifications": [
            {
                "BidPriceAsPercentageOfOnDemandPrice": 100.0,
```

```
                "InstanceType": "m3.xlarge",
                "WeightedCapacity": 1
            }
         ],
         "Id": "if-2ABC4DEFGHIJ4"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstanceFleets。

list-instances

以下代码示例演示了如何使用 list-instances。

Amazon CLI

以下命令列出了集群 ID 为 j-3C6XNQ39VR9WL 的集群中的所有实例：

```
aws emr list-instances --cluster-id j-3C6XNQ39VR9WL
```

输出：

```
For a uniform instance group based cluster
  {
    "Instances": [
         {
            "Status": {
                "Timeline": {
                    "ReadyDateTime": 1433200400.03,
                    "CreationDateTime": 1433199960.152
                },
                "State": "RUNNING",
                "StateChangeReason": {}
            },
            "Ec2InstanceId": "i-f19ecfee",
            "PublicDnsName": "ec2-52-52-41-150.us-west-2.compute.amazonaws.com",
            "PrivateDnsName": "ip-172-21-11-216.us-west-2.compute.internal",
            "PublicIpAddress": "52.52.41.150",
            "Id": "ci-3NNHQUQ2TWB6Y",
            "PrivateIpAddress": "172.21.11.216"
        },
        {
```

```
            "Status": {
                "Timeline": {
                    "ReadyDateTime": 1433200400.031,
                    "CreationDateTime": 1433199949.102
                },
                "State": "RUNNING",
                "StateChangeReason": {}
            },
            "Ec2InstanceId": "i-1feee4c2",
            "PublicDnsName": "ec2-52-63-246-32.us-west-2.compute.amazonaws.com",
            "PrivateDnsName": "ip-172-31-24-130.us-west-2.compute.internal",
            "PublicIpAddress": "52.63.246.32",
            "Id": "ci-GAOCMKNKDCV7",
            "PrivateIpAddress": "172.21.11.215"
        },
        {
            "Status": {
                "Timeline": {
                    "ReadyDateTime": 1433200400.031,
                    "CreationDateTime": 1433199949.102
                },
                "State": "RUNNING",
                "StateChangeReason": {}
            },
            "Ec2InstanceId": "i-15cfeee3",
            "PublicDnsName": "ec2-52-25-246-63.us-west-2.compute.amazonaws.com",
            "PrivateDnsName": "ip-172-31-24-129.us-west-2.compute.internal",
            "PublicIpAddress": "52.25.246.63",
            "Id": "ci-2W3TDFFB47UAD",
            "PrivateIpAddress": "172.21.11.214"
        }
    ]
  }


For a fleet based cluster:
   {
      "Instances": [
          {
              "Status": {
                  "Timeline": {
                      "ReadyDateTime": 1487810810.878,
                      "CreationDateTime": 1487810588.367,
                      "EndDateTime": 1488022990.924
```

```
                },
                "State": "TERMINATED",
                "StateChangeReason": {
                    "Message": "Instance was terminated."
                }
            },
            "Ec2InstanceId": "i-xxxxx",
            "InstanceFleetId": "if-xxxxx",
            "EbsVolumes": [],
            "PublicDnsName": "ec2-xx-xxx-xxx-xxx.compute-1.amazonaws.com",
            "InstanceType": "m3.xlarge",
            "PrivateDnsName": "ip-xx-xx-xxx-xx.ec2.internal",
            "Market": "SPOT",
            "PublicIpAddress": "xx.xx.xxx.xxx",
            "Id": "ci-xxxxx",
            "PrivateIpAddress": "10.47.191.80"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstances。

list-security-configurations

以下代码示例演示了如何使用 list-security-configurations。

Amazon CLI

列出当前区域中的安全配置

命令:

```
aws emr list-security-configurations
```

输出：

```
{
    "SecurityConfigurations": [
        {
            "CreationDateTime": 1473889697.417,
            "Name": "MySecurityConfig-1"
        },
```

```
        {
            "CreationDateTime": 1473889697.417,
            "Name": "MySecurityConfig-2"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecurityConfigurations。

`list-steps`

以下代码示例演示了如何使用 `list-steps`。

Amazon CLI

以下命令列出了集群 ID 为 `j-3SD91U2E1L2QX` 的集群的所有步骤：

```
aws emr list-steps --cluster-id j-3SD91U2E1L2QX
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSteps。

`modify-cluster-attributes`

以下代码示例演示了如何使用 `modify-cluster-attributes`。

Amazon CLI

以下命令将 ID 为 `j-301CDNY0J5XM4` 的 EMR 集群的可见性设置为面向所有用户：

```
aws emr modify-cluster-attributes --cluster-id j-301CDNY0J5XM4 --visible-to-all-
users
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterAttributes。

`modify-instance-fleet`

以下代码示例演示了如何使用 `modify-instance-fleet`。

Amazon CLI

更改实例集的目标容量

此示例将指定实例集的按需目标容量和竞价型目标容量更改为 1。

命令:

```
aws emr modify-instance-fleet --cluster-id 'j-12ABCDEFGHI34JK' --instance-fleet
 InstanceFleetId='if-2ABC4DEFGHIJ4',TargetOnDemandCapacity=1,TargetSpotCapacity=1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyInstanceFleet。

put

以下代码示例演示了如何使用 put。

Amazon CLI

以下命令将名为 healthcheck.sh 的文件上传到集群 ID 为 j-3SD91U2E1L2QX 的集群中的主实例：

```
aws emr put --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem --src ~/
scripts/healthcheck.sh --dest /home/hadoop/bin/healthcheck.sh
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Put。

remove-tags

以下代码示例演示了如何使用 remove-tags。

Amazon CLI

以下命令从集群 ID 为 j-3SD91U2E1L2QX 的集群中删除键为 prod 的标签：

```
aws emr remove-tags --resource-id j-3SD91U2E1L2QX --tag-keys prod
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTags。

schedule-hbase-backup

以下代码示例演示了如何使用 schedule-hbase-backup。

Amazon CLI

注意：此命令只能与 AMI 版本 2.x 和 3.x 上的 HBase 结合使用

1. 计划完整 HBase 备份 >>>>>>> 06ab6d6e13564b5733d75abaf3b599f93cf39a23

命令:

```
aws emr schedule-hbase-backup --cluster-id j-XXXXXXYY --type full --dir
s3://amzn-s3-demo-bucket/backup --interval 10 --unit hours --start-time
2014-04-21T05:26:10Z --consistent
```

输出：

```
None
```

2. 计划增量 HBase 备份

命令:

```
aws emr schedule-hbase-backup --cluster-id j-XXXXXXYY --type incremental
 --dir s3://amzn-s3-demo-bucket/backup --interval 30 --unit minutes --start-time
2014-04-21T05:26:10Z --consistent
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ScheduleHbaseBackup。

socks

以下代码示例演示了如何使用 socks。

Amazon CLI

以下命令打开与集群 ID 为 j-3SD91U2E1L2QX 的集群中主实例的 Socks 连接：

```
aws emr socks --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem
```

密钥对文件选项采用私有密钥文件的本地路径。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Socks。

ssh

以下代码示例演示了如何使用 ssh。

Amazon CLI

以下命令打开与集群 ID 为 j-3SD91U2E1L2QX 的集群中主实例的 Ssh 连接：

```
aws emr ssh --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem
```

密钥对文件选项采用私有密钥文件的本地路径。

输出：

```
ssh -o StrictHostKeyChecking=no -o ServerAliveInterval=10 -i /home/local/user/.ssh/
mykey.pem hadoop@ec2-52-52-41-150.us-west-2.compute.amazonaws.com
Warning: Permanently added 'ec2-52-52-41-150.us-
west-2.compute.amazonaws.com,52.52.41.150' (ECDSA) to the list of known hosts.
Last login: Mon Jun  1 23:15:38 2015

      __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2015.03-release-notes/
26 package(s) needed for security, out of 39 available
Run "sudo yum update" to apply all updates.

--------------------------------------------------------------------------------

Welcome to Amazon Elastic MapReduce running Hadoop and Amazon Linux.

Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

  ResourceManager    lynx http://ip-172-21-11-216:9026/
```

```
  NameNode            lynx http://ip-172-21-11-216:9101/

----------------------------------------------------------------------------

[hadoop@ip-172-31-16-216 ~]$
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Ssh。

使用 Amazon CLI 的 Amazon EMR on EKS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon EMR on EKS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`update-role-trust-policy`

以下代码示例演示了如何使用 `update-role-trust-policy`。

Amazon CLI

更新要与 Amazon EMR on EKS 结合使用的 IAM 角色的信任策略

此示例命令更新名为 example_iam_role 的角色的信任策略，以便它可以与名为 example_cluster 的 EKS 集群中具有 example_namespace 命名空间的 Amazon EMR on EKS 结合使用。

命令:

```
aws emr-containers update-role-trust-policy \
    --cluster example_cluster \
```

```
    --namespace example_namespace \
    --role-name example_iam_role
```

输出：

```
If the trust policy has already been updated, then the output will be:
Trust policy statement already exists for role example_iam_role. No
changes were made!

If the trust policy has not been updated yet, then the output will be:
Successfully updated trust policy of role example_iam_role.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoleTrustPolicy。

使用 Amazon CLI 的 EventBridge 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 EventBridge 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-rule`

以下代码示例演示了如何使用 `delete-rule`。

Amazon CLI

删除 CloudWatch Events 规则

以下示例删除了名为 EC2InstanceStateChanges 的规则：

```
aws events delete-rule --name "EC2InstanceStateChanges"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRule。

describe-rule

以下代码示例演示了如何使用 describe-rule。

Amazon CLI

显示有关 CloudWatch Events 规则的信息

以下示例显示了有关名为 DailyLambdaFunction 的规则的信息：

```
aws events describe-rule --name "DailyLambdaFunction"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRule。

disable-rule

以下代码示例演示了如何使用 disable-rule。

Amazon CLI

禁用 CloudWatch Events 规则

以下示例禁用了名为 DailyLambdaFunction 的规则。该规则未删除：

```
aws events disable-rule --name "DailyLambdaFunction"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableRule。

enable-rule

以下代码示例演示了如何使用 enable-rule。

Amazon CLI

启用 CloudWatch Events 规则

以下示例启用了名为 DailyLambdaFunction 的规则，该规则之前已被禁用：

```
aws events enable-rule --name "DailyLambdaFunction"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableRule。

list-rule-names-by-target

以下代码示例演示了如何使用 list-rule-names-by-target。

Amazon CLI

显示具有指定目标的所有规则

以下示例显示了所有将名为“MyFunctionName”的 Lambda 函数作为目标的规则：

```
aws events list-rule-names-by-target --target-arn "arn:aws:lambda:us-
east-1:123456789012:function:MyFunctionName"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRuleNamesByTarget。

list-rules

以下代码示例演示了如何使用 list-rules。

Amazon CLI

显示所有 CloudWatch Events 规则的列表

以下示例显示了该区域中的所有 CloudWatch Events 规则：

```
aws events list-rules
```

显示以特定字符串开头的 CloudWatch Events 规则列表。

以下示例显示了该区域中名称以“Daily”开头的所有 CloudWatch Events 规则：

```
aws events list-rules --name-prefix "Daily"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRules。

list-targets-by-rule

以下代码示例演示了如何使用 list-targets-by-rule。

Amazon CLI

显示 CloudWatch Events 规则的所有目标

以下示例显示了名为 DailyLambdaFunction 的规则的所有目标：

```
aws events list-targets-by-rule --rule  "DailyLambdaFunction"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetsByRule。

put-events

以下代码示例演示了如何使用 put-events。

Amazon CLI

向 CloudWatch Events 发送自定义事件

以下示例向 CloudWatch Events 发送了自定义事件。该事件包含在 putevents.json 文件中：

```
aws events put-events --entries file://putevents.json
```

以下是 putevents.json 文件的内容：

```
[
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
  },
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value3\", \"key2\": \"value4\" }",
    "Resources": [
```

```
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
   }
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutEvents。

put-rule

以下代码示例演示了如何使用 put-rule。

Amazon CLI

创建 CloudWatch Events 规则

该示例创建一条可在协调世界时每天上午 9:00 触发的规则。如果您使用 put-targets 将 Lambda 函数添加为该规则的目标，则可以在每天的指定时间运行该 Lambda 函数：

```
aws events put-rule --name "DailyLambdaFunction" --schedule-expression "cron(0 9 *
 * ? *)"
```

以下示例创建一条规则，将在区域中的任何 EC2 实例更改状态时触发该规则：

```
aws events put-rule --name "EC2InstanceStateChanges" --event-pattern "{\"source\":
[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change Notification\"]}"  --
role-arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

以下示例创建一条规则，将在区域中的任何 EC2 实例停止或终止时触发该规则：

```
aws events put-rule --name "EC2InstanceStateChangeStopOrTerminate" --event-
pattern "{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change
 Notification\"],\"detail\":{\"state\":[\"stopped\",\"terminated\"]}}" --role-
arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRule。

put-targets

以下代码示例演示了如何使用 put-targets。

Amazon CLI

为 CloudWatch Events 规则添加目标

以下示例添加了一个 Lambda 函数作为规则的目标：

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="1","Arn"="arn:aws:lambda:us-east-1:123456789012:function:MyFunctionName"
```

以下示例将 Amazon Kinesis 流设置为目标，以便将按此规则捕获的事件中继到该流：

```
aws events put-targets --rule EC2InstanceStateChanges --targets
 "Id"="1","Arn"="arn:aws:kinesis:us-east-1:123456789012:stream/
MyStream","RoleArn"="arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

以下示例将两个 Amazon Kinesis 流设置为一条规则的目标：

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="Target1","Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream1","RoleArn"="arn:aws:iam::379642911888:role/ MyRoleToAccessLambda"
  "Id"="Target2"," Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream2","RoleArn"="arn:aws:iam::379642911888:role/MyRoleToAccessLambda"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutTargets。

remove-targets

以下代码示例演示了如何使用 `remove-targets`。

Amazon CLI

移除事件的目标

以下示例将名为 MyStream1 的 Amazon Kinesis 流从规则 DailyLambdaFunction 的目标中移除。创建 DailyLambdaFunction 后，这个流曾被设置为 ID 为 Target1 的目标：

```
aws events remove-targets --rule "DailyLambdaFunction" --ids "Target1"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTargets。

`test-event-pattern`

以下代码示例演示了如何使用 `test-event-pattern`。

Amazon CLI

检查事件模式是否与指定事件匹配

此示例测试模式“source:com.mycompany.myapp”是否与指定事件匹配。在此示例中，输出将为“true”：

```
aws events test-event-pattern --event-pattern "{\"source\":[\"com.mycompany.myapp\"]}" --event "{\"id\":\"1\",\"source\":\"com.mycompany.myapp\",\"detail-type\":\"myDetailType\",\"account\":\"123456789012\",\"region\":\"us-east-1\",\"time\":\"2017-04-11T20:11:04Z\"}"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestEventPattern。

使用 Amazon CLI 的 EventBridge Pipes 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 EventBridge Pipes 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-pipe`

以下代码示例演示了如何使用 `create-pipe`。

Amazon CLI

创建管道

以下 create-pipe 示例创建一个名为 Demo_Pipe 的管道，其中将 SQS 作为该管道的源，并将 CloudWatch 日志组作为该管道的目标。

```
aws pipes create-pipe \
    --name Demo_Pipe \
    --desired-state RUNNING \
    --role-arn arn:aws:iam::123456789012:role/service-role/
Amazon_EventBridge_Pipe_Demo_Pipe_28b3aa4f \
    --source arn:aws:sqs:us-east-1:123456789012:Demo_Queue \
    --target arn:aws:logs:us-east-1:123456789012:log-group:/aws/pipes/Demo_LogGroup
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
    "DesiredState": "RUNNING",
    "CurrentState": "CREATING",
    "CreationTime": "2024-10-08T12:33:59-05:00",
    "LastModifiedTime": "2024-10-08T12:33:59.684839-05:00"
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreatePipe。

delete-pipe

以下代码示例演示了如何使用 delete-pipe。

Amazon CLI

删除现有管道

以下 delete-pipe 示例从指定账户中删除名为 Demo_Pipe 的管道。

```
aws pipes delete-pipe \
    --name Demo_Pipe
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
    "DesiredState": "STOPPED",
    "CurrentState": "DELETING",
    "CreationTime": "2024-10-08T09:29:10-05:00",
    "LastModifiedTime": "2024-10-08T11:57:22-05:00"
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeletePipe。

describe-pipe

以下代码示例演示了如何使用 describe-pipe。

Amazon CLI

检索有关管道的信息

以下 describe-pipe 示例显示有关指定账户中的管道 Demo_Pipe 的信息。

```
aws pipes describe-pipe \
    --name Demo_Pipe
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
    "DesiredState": "RUNNING",
    "CurrentState": "RUNNING",
    "StateReason": "User initiated",
    "Source": "arn:aws:sqs:us-east-1:123456789012:Demo_Queue",
    "SourceParameters": {
        "SqsQueueParameters": {
            "BatchSize": 1
        }
    },
    "EnrichmentParameters": {},
```

```
    "Target": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/pipes/
Demo_LogGroup",
    "TargetParameters": {},
    "RoleArn": "arn:aws:iam::123456789012:role/service-role/
Amazon_EventBridge_Pipe_Demo_Pipe_28b3aa4f",
    "Tags": {},
    "CreationTime": "2024-10-08T09:29:10-05:00",
    "LastModifiedTime": "2024-10-08T10:23:47-05:00",
    "LogConfiguration": {
        "CloudwatchLogsLogDestination": {
            "LogGroupArn": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/
vendedlogs/pipes/Demo_Pipe"
        },
        "Level": "ERROR"
    }
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribePipe。

`list-pipes`

以下代码示例演示了如何使用 `list-pipes`。

Amazon CLI

检索管道的列表

以下 `list-pipes` 示例显示指定账户中的所有管道。

```
aws pipes list-pipes
```

输出：

```
{
    "Pipes": [
        {
            "Name": "Demo_Pipe",
            "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
            "DesiredState": "RUNNING",
```

```
            "CurrentState": "RUNNING",
            "StateReason": "User initiated",
            "CreationTime": "2024-10-08T09:29:10-05:00",
            "LastModifiedTime": "2024-10-08T10:23:47-05:00",
            "Source": "arn:aws:sqs:us-east-1:123456789012:Demo_Queue",
            "Target": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/pipes/
Demo_LogGroup"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListPipes。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出与现有管道关联的标签

以下 `list-tags-for-resource` 示例列出与指定账户中名为 `Demo_Pipe` 的管道关联的所有标签。

```
aws pipes list-tags-for-resource \
    --resource-arn arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe
```

输出：

```
{
    "tags": {
        "stack": "Production",
        "team": "DevOps"
    }
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-pipe

以下代码示例演示了如何使用 start-pipe。

Amazon CLI

启动现有管道

以下 start-pipe 示例启动指定账户中名为 Demo_Pipe 的管道。

```
aws pipes start-pipe \
    --name Demo_Pipe
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
    "DesiredState": "RUNNING",
    "CurrentState": "STARTING",
    "CreationTime": "2024-10-08T09:29:10-05:00",
    "LastModifiedTime": "2024-10-08T10:17:24-05:00"
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Starting or stopping an Amazon EventBridge pipe。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartPipe。

stop-pipe

以下代码示例演示了如何使用 stop-pipe。

Amazon CLI

停止现有管道

以下 stop-pipe 示例停止指定账户中名为 Demo_Pipe 的管道。

```
aws pipes stop-pipe \
    --name Demo_Pipe
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
    "DesiredState": "STOPPED",
    "CurrentState": "STOPPING",
    "CreationTime": "2024-10-08T09:29:10-05:00",
    "LastModifiedTime": "2024-10-08T09:29:49-05:00"
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Starting or stopping an Amazon EventBridge pipe。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopPipe。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记现有管道

以下 tag-resource 示例标记名为 Demo_Pipe 的管道。如果命令成功，则不返回任何输出。

```
aws pipes tag-resource \
    --resource-arn arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe \
    --tags stack=Production
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从现有管道中移除标签

以下 untag-resource 示例从名为 Demo_Pipe 的管道中移除键为 stack 的标签。如果命令成功，则不返回任何输出。

```
aws pipes untag-resource \
    --resource-arn arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe \
    --tags stack
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-pipe

以下代码示例演示了如何使用 update-pipe。

Amazon CLI

更新现有管道

以下 update-pipe 示例通过添加 CloudWatch Log 配置参数来更新名为 Demo_Pipe 的管道，确保更新该管道的执行角色，使其具有 Log 目标的正确权限。

```
aws pipes update-pipe \
    --name Demo_Pipe \
    --desired-state RUNNING \
    --log-configuration CloudwatchLogsLogDestination={LogGroupArn=arn:aws:logs:us-
east-1:123456789012:log-group:/aws/vendedlogs/pipes/Demo_Pipe},Level=TRACE \
    --role-arn arn:aws:iam::123456789012:role/service-role/
Amazon_EventBridge_Pipe_Demo_Pipe_28b3aa4f
```

输出：

```
{
    "Arn": "arn:aws:pipes:us-east-1:123456789012:pipe/Demo_Pipe",
    "Name": "Demo_Pipe",
```

```
    "DesiredState": "RUNNING",
    "CurrentState": "UPDATING",
    "CreationTime": "2024-10-08T09:29:10-05:00",
    "LastModifiedTime": "2024-10-08T11:35:48-05:00"
}
```

有关更多信息，请参阅《Amazon EventBridge User Guide》中的 Amazon EventBridge Pipes concepts。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdatePipe。

使用 Amazon CLI 的 Firewall Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Firewall Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-admin-account`

以下代码示例演示了如何使用 `associate-admin-account`。

Amazon CLI

设置 Firewall Manager 管理员账户

以下 `associate-admin-account` 示例设置 Firewall Manager 的管理员账户。

```
aws fms associate-admin-account \
    --admin-account 123456789012
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的设置 Amazon Firewall Manager 管理员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAdminAccount。

delete-notification-channel

以下代码示例演示了如何使用 delete-notification-channel。

Amazon CLI

删除 Firewall Manager 日志的 SNS 主题信息

以下 delete-notification-channel 示例删除 SNS 主题信息。

```
aws fms delete-notification-channel
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的配置 Amazon SNS 通知和 Amazon CloudWatch 警报。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNotificationChannel。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除 Firewall Manager 策略

以下 delete-policy 示例删除具有指定 ID 的策略及其所有资源。

```
aws fms delete-policy \
    --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --delete-all-policy-resources
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的使用 Amazon Firewall Manager 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

`disassociate-admin-account`

以下代码示例演示了如何使用 `disassociate-admin-account`。

Amazon CLI

删除 Firewall Manager 管理员账户

以下 `disassociate-admin-account` 示例从 Firewall Manager 中删除当前的管理员账户关联。

```
aws fms disassociate-admin-account
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的设置 Amazon Firewall Manager 管理员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateAdminAccount。

`get-admin-account`

以下代码示例演示了如何使用 `get-admin-account`。

Amazon CLI

检索 Firewall Manager 管理员账户

以下 `get-admin-account` 示例检索管理员账户。

```
aws fms get-admin-account
```

输出：

```
{
    "AdminAccount": "123456789012",
    "RoleStatus": "READY"
```

```
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon Firewall Manager 先决条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAdminAccount。

get-compliance-detail

以下代码示例演示了如何使用 get-compliance-detail。

Amazon CLI

检索账户的合规信息

以下 get-compliance-detail 示例检索指定策略和成员账户的合规信息。

```
aws fms get-compliance-detail \
    --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --member-account 123456789012
```

输出：

```
{
    "PolicyComplianceDetail": {
    "EvaluationLimitExceeded": false,
    "IssueInfoMap": {},
    "MemberAccount": "123456789012",
    "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "PolicyOwner": "123456789012",
    "Violators": []
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的查看资源的策略合规性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComplianceDetail。

get-notification-channel

以下代码示例演示了如何使用 get-notification-channel。

Amazon CLI

检索 Firewall Manager 日志的 SNS 主题信息

以下 get-notification-channel 示例检索 SNS 主题信息。

```
aws fms get-notification-channel
```

输出：

```
{
    "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:us-west-2-fms",
    "SnsRoleName": "arn:aws:iam::123456789012:role/aws-service-role/
fms.amazonaws.com/AWSServiceRoleForFMS"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的配置 Amazon SNS 通知和 Amazon CloudWatch 警报。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetNotificationChannel。

get-policy

以下代码示例演示了如何使用 get-policy。

Amazon CLI

检索 Firewall Manager 策略

以下 get-policy 示例检索具有指定 ID 的策略。

```
aws fms get-policy \
    --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Policy": {
        "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "PolicyName": "test",
        "PolicyUpdateToken": "1:p+2RpKR4wPFx7mcrL1UOQQ==",
        "SecurityServicePolicyData": {
```

```
            "Type": "SECURITY_GROUPS_COMMON",
            "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_COMMON\",
\"revertManualSecurityGroupChanges\":true,\"exclusiveResourceSecurityGroupManagement
\":false,\"securityGroups\":[{\"id\":\"sg-045c43ccc9724e63e\"}]}"
        },
        "ResourceType": "AWS::EC2::Instance",
        "ResourceTags": [],
        "ExcludeResourceTags": false,
        "RemediationEnabled": false
    },
    "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/d1ac59b8-938e-42b3-
b2e0-7c620422ddc2"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的使用 Amazon Firewall Manager 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicy。

list-compliance-status

以下代码示例演示了如何使用 list-compliance-status。

Amazon CLI

检索成员账户的策略合规信息

以下 list-compliance-status 示例检索指定策略的成员账户合规信息。

```
aws fms list-compliance-status \
    --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "PolicyComplianceStatusList": [
        {
            "PolicyOwner": "123456789012",
            "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "PolicyName": "test",
            "MemberAccount": "123456789012",
            "EvaluationResults": [
```

```
                {
                    "ComplianceStatus": "COMPLIANT",
                    "ViolatorCount": 0,
                    "EvaluationLimitExceeded": false
                },
                {
                    "ComplianceStatus": "NON_COMPLIANT",
                    "ViolatorCount": 2,
                    "EvaluationLimitExceeded": false
                }
            ],
            "LastUpdated": 1576283774.0,
            "IssueInfoMap": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的查看资源的策略合规性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListComplianceStatus。

`list-member-accounts`

以下代码示例演示了如何使用 `list-member-accounts`。

Amazon CLI

检索组织中的成员账户

以下 `list-member-accounts` 示例列出 Firewall Manager 管理员的组织中的所有成员账户。

```
aws fms list-member-accounts
```

输出：

```
{
    "MemberAccounts": [
        "222222222222",
        "333333333333",
        "444444444444"
    ]
```

```
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon Firewall Manager。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMemberAccounts。

`list-policies`

以下代码示例演示了如何使用 `list-policies`。

Amazon CLI

检索所有 Firewall Manager 策略

以下 `list-policies` 示例检索账户的策略列表。在此示例中，每个请求的输出限制为两个结果。每次调用都会返回一个 `NextToken`，它可用作下一次 `list-policies` 调用中 `--starting-token` 参数的值，以获取列表的下一组结果。

```
aws fms list-policies \
    --max-items 2
```

输出：

```
{
    "PolicyList": [
        {
            "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "PolicyName": "test",
            "ResourceType": "AWS::EC2::Instance",
            "SecurityServiceType": "SECURITY_GROUPS_COMMON",
            "RemediationEnabled": false
        },
        {
            "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "PolicyId": "457c9b21-fc94-406c-ae63-21217395ba72",
            "PolicyName": "test",
            "ResourceType": "AWS::EC2::Instance",
            "SecurityServiceType": "SECURITY_GROUPS_COMMON",
```

```
            "RemediationEnabled": false
        }
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的使用 Amazon Firewall Manager 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicies。

put-notification-channel

以下代码示例演示了如何使用 put-notification-channel。

Amazon CLI

设置 Firewall Manager 日志的 SNS 主题信息

以下 put-notification-channel 示例设置 SNS 主题信息。

```
aws fms put-notification-channel \
    --sns-topic-arn arn:aws:sns:us-west-2:123456789012:us-west-2-fms \
    --sns-role-name arn:aws:iam::123456789012:role/aws-service-role/
fms.amazonaws.com/AWSServiceRoleForFMS
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的配置 Amazon SNS 通知和 Amazon CloudWatch 警报。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutNotificationChannel。

put-policy

以下代码示例演示了如何使用 put-policy。

Amazon CLI

创建 Firewall Manager 策略

以下 put-policy 示例创建一个 Firewall Manager 安全组策略。

```
aws fms put-policy \
    --cli-input-json file://policy.json
```

的内容policy.json：

```
{
    "Policy": {
        "PolicyName": "test",
        "SecurityServicePolicyData": {
            "Type": "SECURITY_GROUPS_USAGE_AUDIT",
            "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_USAGE_AUDIT\",
\"deleteUnusedSecurityGroups\":false,\"coalesceRedundantSecurityGroups\":true}"
        },
        "ResourceType": "AWS::EC2::SecurityGroup",
        "ResourceTags": [],
        "ExcludeResourceTags": false,
        "RemediationEnabled": false
    },
    "TagList": [
        {
            "Key": "foo",
            "Value": "foo"
        }
    ]
}
```

输出：

```
{
    "Policy": {
        "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "PolicyName": "test",
        "PolicyUpdateToken": "1:X9QGexP7HASDlsFp+G31Iw==",
        "SecurityServicePolicyData": {
            "Type": "SECURITY_GROUPS_USAGE_AUDIT",
            "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_USAGE_AUDIT\",
\"deleteUnusedSecurityGroups\":false,\"coalesceRedundantSecurityGroups\":true,
\"optionalDelayForUnusedInMinutes\":null}"
        },
        "ResourceType": "AWS::EC2::SecurityGroup",
        "ResourceTags": [],
        "ExcludeResourceTags": false,
```

```
        "RemediationEnabled": false
    },
    "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的使用 Amazon Firewall Manager 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPolicy。

使用 Amazon CLI 的 Amazon FIS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon FIS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-experiment-template`

以下代码示例演示了如何使用 `create-experiment-template`。

Amazon CLI

创建试验模板

以下 `create-experiment-template` 示例在您的 Amazon FIS 账户中创建一个试验模板。

```
aws fis create-experiment-template \
    --cli-input-json file://myfile.json
```

的内容`myfile.json`：

```
{
    "description": "experimentTemplate",
    "stopConditions": [
        {
            "source": "aws:cloudwatch:alarm",
            "value": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:alarmName"
        }
    ],
    "targets": {
        "Instances-Target-1": {
            "resourceType": "aws:ec2:instance",
            "resourceArns": [
                "arn:aws:ec2:us-west-2:123456789012:instance/i-12a3b4c56d78e9012"
            ],
            "selectionMode": "ALL"
        }
    },
    "actions": {
        "reboot": {
            "actionId": "aws:ec2:reboot-instances",
            "description": "reboot",
            "parameters": {},
            "targets": {
                "Instances": "Instances-Target-1"
            }
        }
    },
    "roleArn": "arn:aws:iam::123456789012:role/myRole"
}
```

输出：

```
{
    "experimentTemplate": {
        "id": "ABCDE1fgHIJkLmNop",
        "description": "experimentTemplate",
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
```

```
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "reboot": {
                "actionId": "aws:ec2:reboot-instances",
                "description": "reboot",
                "parameters": {},
                "targets": {
                    "Instances": "Instances-Target-1"
                }
            }
        },
        "stopConditions": [
            {
                "source": "aws:cloudwatch:alarm",
                "value": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:alarmName"
            }
        ],
        "creationTime": 1616434850.659,
        "lastUpdateTime": 1616434850.659,
        "roleArn": "arn:aws:iam::123456789012:role/myRole",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的创建试验模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateExperimentTemplate。

delete-experiment-template

以下代码示例演示了如何使用 delete-experiment-template。

Amazon CLI

删除试验模板

以下 delete-experiment-template 示例删除指定的试验模板。

```
aws fis delete-experiment-template \
    --id ABCDE1fgHIJkLmNop
```

输出：

```
{
    "experimentTemplate": {
        "id": "ABCDE1fgHIJkLmNop",
        "description": "myExperimentTemplate",
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "testaction": {
                "actionId": "aws:ec2:stop-instances",
                "parameters": {},
                "targets": {
                    "Instances": "Instances-Target-1"
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616017191.124,
        "lastUpdateTime": 1616017859.607,
        "roleArn": "arn:aws:iam::123456789012:role/FISRole"
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的删除试验模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteExperimentTemplate。

get-action

以下代码示例演示了如何使用 get-action。

Amazon CLI

获取操作详细信息

以下 get-action 示例获取指定操作的详细信息。

```
aws fis get-action \
    --id aws:ec2:stop-instances
```

输出：

```
{
    "action": {
        "id": "aws:ec2:stop-instances",
        "description": "Stop the specified EC2 instances.",
        "parameters": {
            "startInstancesAfterDuration": {
                "description": "The time to wait before restarting the instances
 (ISO 8601 duration).",
                "required": false
            }
        },
        "targets": {
            "Instances": {
                "resourceType": "aws:ec2:instance"
            }
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAction。

get-experiment-template

以下代码示例演示了如何使用 get-experiment-template。

Amazon CLI

获取试验模板详细信息

以下 get-experiment-template 示例获取指定试验模板的详细信息。

```
aws fis get-experiment-template \
    --id ABCDE1fgHIJkLmNop
```

输出：

```
{
    "experimentTemplate": {
        "id": "ABCDE1fgHIJkLmNop",
        "description": "myExperimentTemplate",
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "testaction": {
                "actionId": "aws:ec2:stop-instances",
                "parameters": {},
                "targets": {
                    "Instances": "Instances-Target-1"
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616017191.124,
        "lastUpdateTime": 1616017331.51,
        "roleArn": "arn:aws:iam::123456789012:role/FISRole",
        "tags": {
        "key: "value"
        }
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的试验模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExperimentTemplate。

get-experiment

以下代码示例演示了如何使用 get-experiment。

Amazon CLI

获取试验详细信息

以下 get-experiment 示例获取指定试验的详细信息。

```
aws fis get-experiment \
    --id ABC12DeFGhI3jKLMNOP
```

输出：

```
{
    "experiment": {
        "id": "ABC12DeFGhI3jKLMNOP",
        "experimentTemplateId": "ABCDE1fgHIJkLmNop",
        "roleArn": "arn:aws:iam::123456789012:role/myRole",
        "state": {
            "status": "completed",
            "reason": "Experiment completed."
        },
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "reboot": {
                "actionId": "aws:ec2:reboot-instances",
                "parameters": {},
                "targets": {
```

```
                    "Instances": "Instances-Target-1"
                },
                "state": {
                    "status": "completed",
                    "reason": "Action was completed."
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616432509.662,
        "startTime": 1616432509.962,
        "endTime": 1616432522.307,
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的 Amazon FIS 试验。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExperiment。

`list-actions`

以下代码示例演示了如何使用 `list-actions`。

Amazon CLI

列出操作

以下 `list-actions` 示例列出可用操作。

```
aws fis list-actions
```

输出：

```
{
    "actions": [
        {
            "id": "aws:ec2:reboot-instances",
            "description": "Reboot the specified EC2 instances.",
```

```
            "targets": {
                "Instances": {
                    "resourceType": "aws:ec2:instance"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:ec2:stop-instances",
            "description": "Stop the specified EC2 instances.",
            "targets": {
                "Instances": {
                    "resourceType": "aws:ec2:instance"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:ec2:terminate-instances",
            "description": "Terminate the specified EC2 instances.",
            "targets": {
                "Instances": {
                    "resourceType": "aws:ec2:instance"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:ecs:drain-container-instances",
            "description": "Drain percentage of underlying EC2 instances on an ECS
cluster.",
            "targets": {
                "Clusters": {
                    "resourceType": "aws:ecs:cluster"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:eks:terminate-nodegroup-instances",
            "description": "Terminates a percentage of the underlying EC2 instances
in an EKS cluster.",
            "targets": {
                "Nodegroups": {
```

```
                    "resourceType": "aws:eks:nodegroup"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:fis:inject-api-internal-error",
            "description": "Cause an AWS service to return internal error responses
for specific callers and operations.",
            "targets": {
                "Roles": {
                    "resourceType": "aws:iam:role"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:fis:inject-api-throttle-error",
            "description": "Cause an AWS service to return throttled responses for
specific callers and operations.",
            "targets": {
                "Roles": {
                    "resourceType": "aws:iam:role"
                }
            },
            "tags": {}
        },
        {
        "id": "aws:fis:inject-api-unavailable-error",
            "description": "Cause an AWS service to return unavailable error
responses for specific callers and operations.",
            "targets": {
                "Roles": {
                    "resourceType": "aws:iam:role"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:fis:wait",
            "description": "Wait for the specified duration. Stop condition
monitoring will continue during this time.",
            "tags": {}
        },
```

```
        {
            "id": "aws:rds:failover-db-cluster",
            "description": "Failover a DB Cluster to one of the replicas.",
            "targets": {
                "Clusters": {
                    "resourceType": "aws:rds:cluster"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:rds:reboot-db-instances",
            "description": "Reboot the specified DB instances.",
            "targets": {
                "DBInstances": {
                    "resourceType": "aws:rds:db"
                }
            },
            "tags": {}
        },
        {
            "id": "aws:ssm:send-command",
            "description": "Run the specified SSM document.",
            "targets": {
                "Instances": {
                    "resourceType": "aws:ec2:instance"
                }
            },
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListActions。

list-experiment-templates

以下代码示例演示了如何使用 list-experiment-templates。

Amazon CLI

列出试验模板

以下 list-experiment-templates 示例列出您的 Amazon 账户中的试验模板。

```
aws fis list-experiment-templates
```

输出：

```
{
    "experimentTemplates": [
        {
            "id": "ABCDE1fgHIJkLmNop",
            "description": "myExperimentTemplate",
            "creationTime": 1616017191.124,
            "lastUpdateTime": 1616017191.124,
            "tags": {
                "key": "value"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的试验模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListExperimentTemplates。

list-experiments

以下代码示例演示了如何使用 list-experiments。

Amazon CLI

列出试验

以下 list-experiments 示例列出您的 Amazon 账户中的试验。

```
aws fis list-experiments
```

输出：

```
{
    "experiments": [
        {
            "id": "ABCdeF1GHiJkLM23NO",
```

```
            "experimentTemplateId": "ABCDE1fgHIJkLmNop",
            "state": {
                "status": "running",
                "reason": "Experiment is running."
            },
            "creationTime": 1616017341.197,
            "tags": {
            "key": "value"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的试验。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListExperiments。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出资源标签

以下 `list-tags-for-resource` 示例列出指定资源的标签。

```
aws fis list-tags-for-resource \
    --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
```

输出：

```
{
    "tags": {
        "key1": "value1",
        "key2": "value2"
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的标记 Amazon FIS 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-experiment

以下代码示例演示了如何使用 start-experiment。

Amazon CLI

开始试验

以下 start-experiment 示例开始指定的试验。

```
aws fis start-experiment \
    --experiment-template-id ABCDE1fgHIJkLmNop
```

输出：

```
{
    "experiment": {
        "id": "ABC12DeFGhI3jKLMNOP",
        "experimentTemplateId": "ABCDE1fgHIJkLmNop",
        "roleArn": "arn:aws:iam::123456789012:role/myRole",
        "state": {
            "status": "initiating",
            "reason": "Experiment is initiating."
        },
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "reboot": {
                "actionId": "aws:ec2:reboot-instances",
                "parameters": {},
                "targets": {
                    "Instances": "Instances-Target-1"
                },
                "state": {
                    "status": "pending",
                    "reason": "Initial state"
```

```
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616432464.025,
        "startTime": 1616432464.374,
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的 Amazon FIS 试验。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartExperiment。

stop-experiment

以下代码示例演示了如何使用 stop-experiment。

Amazon CLI

停止试验

以下 stop-experiment 示例停止运行指定的试验。

```
aws fis stop-experiment \
    --id ABC12DeFGhI3jKLMNOP
```

输出：

```
{
    "experiment": {
        "id": "ABC12DeFGhI3jKLMNOP",
        "experimentTemplateId": "ABCDE1fgHIJkLmNop",
        "roleArn": "arn:aws:iam::123456789012:role/myRole",
        "state": {
            "status": "stopping",
            "reason": "Stopping Experiment."
        },
        "targets": {
```

```
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "reboot": {
                "actionId": "aws:ec2:reboot-instances",
                "parameters": {},
                "targets": {
                    "Instances": "Instances-Target-1"
                },
                "startAfter": [
                    "wait"
                ],
                "state": {
                    "status": "pending",
                    "reason": "Initial state."
                }
            },
            "wait": {
                "actionId": "aws:fis:wait",
                "parameters": {
                    "duration": "PT5M"
                },
                "state": {
                    "status": "running",
                    "reason": ""
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616432680.927,
        "startTime": 1616432681.177,
        "tags": {}
    }
```

```
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的 Amazon FIS 试验。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopExperiment。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例标记指定的资源。

```
aws fis tag-resource \
    --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
 \
    --tags key1=value1,key2=value2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的标记 Amazon FIS 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

取消标记资源

以下 untag-resource 示例从指定资源中删除标签。

```
aws fis untag-resource \
    --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的标记 Amazon FIS 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-experiment-template

以下代码示例演示了如何使用 update-experiment-template。

Amazon CLI

更新试验模板

以下 update-experiment-template 示例更新指定试验模板的描述。

```
aws fis update-experiment-template \
    --id ABCDE1fgHIJkLmNop \
    ---description myExperimentTemplate
```

输出：

```
{
    "experimentTemplate": {
        "id": "ABCDE1fgHIJkLmNop",
        "description": "myExperimentTemplate",
        "targets": {
            "Instances-Target-1": {
                "resourceType": "aws:ec2:instance",
                "resourceArns": [
                    "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
                ],
                "selectionMode": "ALL"
            }
        },
        "actions": {
            "testaction": {
                "actionId": "aws:ec2:stop-instances",
                "parameters": {},
                "targets": {
```

```
                    "Instances": "Instances-Target-1"
                }
            }
        },
        "stopConditions": [
            {
                "source": "none"
            }
        ],
        "creationTime": 1616017191.124,
        "lastUpdateTime": 1616017859.607,
        "roleArn": "arn:aws:iam::123456789012:role/FISRole",
        "tags": {
            "key": "value"
        }
    }
}
```

有关更多信息，请参阅《Amazon Fault Injection Simulator 用户指南》中的更新试验模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateExperimentTemplate。

使用 Amazon CLI 的 Amazon GameLift Servers 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon GameLift Servers 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-build`

以下代码示例演示了如何使用 `create-build`。

Amazon CLI

示例 1：使用 S3 存储桶中的文件创建游戏生成包

以下 create-build 示例创建自定义游戏生成包资源。它使用存储在您控制的 Amazon 账户的 S3 位置中的压缩文件。此示例假定您已经创建了一个 IAM 角色，该角色授予 Amazon GameLift 访问 S3 位置的权限。由于请求未指定操作系统，因此新的生成包资源默认为 WINDOWS_2012。

```
aws gamelift create-build \
    --storage-location file://storage-loc.json \
    --name MegaFrogRaceServer.NA \
    --build-version 12345.678
```

的内容storage-loc.json：

```
{
    "Bucket":"MegaFrogRaceServer_NA_build_files"
    "Key":"MegaFrogRaceServer_build_123.zip"
    "RoleArn":"arn:aws:iam::123456789012:role/gamelift"
}
```

输出：

```
{
    "Build": {
        "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreationTime": 1496708916.18,
        "Name": "MegaFrogRaceServer.NA",
        "OperatingSystem": "WINDOWS_2012",
        "SizeOnDisk": 479303,
        "Status": "INITIALIZED",
        "Version": "12345.678"
    },
    "StorageLocation": {
        "Bucket": "MegaFrogRaceServer_NA_build_files",
        "Key": "MegaFrogRaceServer_build_123.zip"
    }
}
```

示例 2：创建用于手动将文件上传到 GameLift 的游戏生成包资源

以下 create-build 示例创建一个新的生成包资源。它还会获取存储位置和临时凭证，允许您将游戏生成包手动上传到 Amazon S3 中的 GameLift 位置。成功上传生成包后，GameLift 服务会验证该生成包并对新生成包的状态进行更新。

```
aws gamelift create-build \
    --name MegaFrogRaceServer.NA \
    --build-version 12345.678 \
    --operating-system AMAZON_LINUX
```

输出：

```
{
    "Build": {
        "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreationTime": 1496708916.18,
        "Name": "MegaFrogRaceServer.NA",
        "OperatingSystem": "AMAZON_LINUX",
        "SizeOnDisk": 0,
        "Status": "INITIALIZED",
        "Version": "12345.678"
    },
    "StorageLocation": {
        "Bucket": "gamelift-builds-us-west-2",
        "Key": "123456789012/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "UploadCredentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AgoGb3JpZ2luENz...EXAMPLETOKEN=="
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的将自定义服务器生成包上传到 GameLift。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBuild。

create-fleet

以下代码示例演示了如何使用 create-fleet。

Amazon CLI

示例 1：创建基本 Linux 实例集

以下 create-fleet 示例创建一个最低配置的按需型 Linux 实例集来托管自定义服务器生成包。您可以通过使用 update-fleet 来完成配置。

```
aws gamelift create-fleet \
    --name MegaFrogRaceServer.NA.v2 \
    --description 'Hosts for v2 North America' \
    --build-id build-1111aaaa-22bb-33cc-44dd-5555eeee66ff \
    --certificate-configuration 'CertificateType=GENERATED' \
    --ec2-instance-type c4.large \
    --fleet-type ON_DEMAND \
    --runtime-configuration 'ServerProcesses=[{LaunchPath=/local/game/release-na/
MegaFrogRace_Server.exe,ConcurrentExecutions=1}]'
```

输出：

```
{
    "FleetAttributes": {
        "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
        "CertificateConfiguration": {
            "CertificateType": "GENERATED"
        },
        "CreationTime": 1496365885.44,
        "Description": "Hosts for v2 North America",
        "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetType": "ON_DEMAND",
        "InstanceType": "c4.large",
        "MetricGroups": ["default"],
        "Name": "MegaFrogRace.NA.v2",
        "NewGameSessionProtectionPolicy": "NoProtection",
        "OperatingSystem": "AMAZON_LINUX",
        "ServerLaunchPath": "/local/game/release-na/MegaFrogRace_Server.exe",
        "Status": "NEW"
    }
}
```

示例 2：创建基本 Windows 实例集

以下 create-fleet 示例创建一个最低配置的竞价型 Windows 实例集来托管自定义服务器生成包。您可以通过使用 update-fleet 来完成配置。

```
aws gamelift create-fleet \
    --name MegaFrogRace.NA.v2 \
    --description 'Hosts for v2 North America' \
    --build-id build-2222aaaa-33bb-44cc-55dd-6666eeee77ff  \
    --certificate-configuration 'CertificateType=GENERATED' \
    --ec2-instance-type c4.large \
    --fleet-type SPOT \
    --runtime-configuration 'ServerProcesses=[{LaunchPath=C:\game
\Bin64.Release.Dedicated\MegaFrogRace_Server.exe,ConcurrentExecutions=1}]'
```

输出：

```
{
    "FleetAttributes": {
        "BuildId": "build-2222aaaa-33bb-44cc-55dd-6666eeee77ff",
        "CertificateConfiguration": {
            "CertificateType": "GENERATED"
        },
        "CreationTime": 1496365885.44,
        "Description": "Hosts for v2 North America",
        "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetType": "SPOT",
        "InstanceType": "c4.large",
        "MetricGroups": ["default"],
        "Name": "MegaFrogRace.NA.v2",
        "NewGameSessionProtectionPolicy": "NoProtection",
        "OperatingSystem": "WINDOWS_2012",
        "ServerLaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
        "Status": "NEW"
    }
}
```

示例 3：创建完全配置的实例集

以下 create-fleet 示例为自定义服务器生成包创建了一个竞价型 Windows 实例集，并提供了最常用的配置设置。

```
aws gamelift create-fleet \
    --name MegaFrogRace.NA.v2 \
    --description 'Hosts for v2 North America' \
    --build-id build-2222aaaa-33bb-44cc-55dd-6666eeee77ff \
    --certificate-configuration 'CertificateType=GENERATED' \
    --ec2-instance-type c4.large \
    --ec2-inbound-permissions
 'FromPort=33435,ToPort=33435,IpRange=10.24.34.0/23,Protocol=UDP' \
    --fleet-type SPOT \
    --new-game-session-protection-policy FullProtection \
    --runtime-configuration file://runtime-config.json \
    --metric-groups default \
    --instance-role-arn 'arn:aws:iam::444455556666:role/GameLiftS3Access'
```

的内容runtime-config.json：

```
GameSessionActivationTimeoutSeconds=300,
 MaxConcurrentGameSessionActivations=2,
 ServerProcesses=[
   {LaunchPath=C:\game\Bin64.Release.Dedicated\MegaFrogRace_Server.exe,Parameters=-
debug,ConcurrentExecutions=1},
   {LaunchPath=C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe,ConcurrentExecutions=1}]
```

输出：

```
{
    "FleetAttributes": {
        "InstanceRoleArn": "arn:aws:iam::444455556666:role/GameLiftS3Access",
        "Status": "NEW",
        "InstanceType": "c4.large",
        "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "Description": "Hosts for v2 North America",
        "FleetType": "SPOT",
        "OperatingSystem": "WINDOWS_2012",
        "Name": "MegaFrogRace.NA.v2",
        "CreationTime": 1569309011.11,
        "MetricGroups": [
            "default"
        ],
```

```
        "BuildId": "build-2222aaaa-33bb-44cc-55dd-6666eeee77ff",
        "ServerLaunchParameters": "abc",
        "ServerLaunchPath": "C:\\game\\Bin64.Release.Dedicated\
\MegaFrogRace_Server.exe",
        "NewGameSessionProtectionPolicy": "FullProtection",
        "CertificateConfiguration": {
            "CertificateType": "GENERATED"
        }
    }
}
```

示例 4：创建实时服务器队组

以下 `create-fleet` 示例使用已上传到 Amazon GameLift 的实时配置脚本创建竞价型实例集。所有实时服务器均部署在 Linux 计算机上。出于本示例的目的，假定上传的实时脚本包含多个脚本文件，其中 `Init()` 函数位于名为 `MainScript.js` 的脚本文件中。如下所示，此文件在运行时配置中被标识为启动脚本。

```
aws gamelift create-fleet \
    --name MegaFrogRace.NA.realtime \
    --description 'Mega Frog Race Realtime fleet' \
    --script-id script-1111aaaa-22bb-33cc-44dd-5555eeee66ff \
    --ec2-instance-type c4.large \
    --fleet-type SPOT \
    --certificate-configuration 'CertificateType=GENERATED' --runtime-configuration
 'ServerProcesses=[{LaunchPath=/local/game/MainScript.js,Parameters=+map
 Winter444,ConcurrentExecutions=5}]'
```

输出：

```
{
    "FleetAttributes": {
        "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "Status": "NEW",
        "CreationTime": 1569310745.212,
        "InstanceType": "c4.large",
        "NewGameSessionProtectionPolicy": "NoProtection",
        "CertificateConfiguration": {
            "CertificateType": "GENERATED"
        },
        "Name": "MegaFrogRace.NA.realtime",
        "ScriptId": "script-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
```

```
        "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
        "FleetType": "SPOT",
        "MetricGroups": [
            "default"
        ],
        "Description": "Mega Frog Race Realtime fleet",
        "OperatingSystem": "AMAZON_LINUX"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFleet。

create-game-session-queue

以下代码示例演示了如何使用 create-game-session-queue。

Amazon CLI

示例 1：设置有序游戏会话队列

以下 create-game-session-queue 示例创建一个新的游戏会话队列，其目标位于两个区域。它还会配置队列，以便游戏会话在等待放置 10 分钟后请求超时。由于未定义延迟策略，因此 GameLift 会尝试将所有游戏会话放置在列出的第一个目标上。

```
aws gamelift create-game-session-queue \
    --name MegaFrogRaceServer-NA \
    --destinations file://destinations.json \
    --timeout-in-seconds 600
```

的内容destinations.json：

```
{
    "Destinations": [
        {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" },
        {"DestinationArn": "arn:aws:gamelift:us-west-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222" }
    ]
}
```

输出：

```
{
    "GameSessionQueues": [
        {
            "Name": "MegaFrogRaceServer-NA",
            "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:123456789012:gamesessionqueue/MegaFrogRaceServer-NA",
            "TimeoutInSeconds": 600,
            "Destinations": [
                {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"},
                {"DestinationArn": "arn:aws:gamelift:us-west-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"}
            ]
        }
    ]
}
```

示例 2：使用玩家延迟策略设置游戏会话队列

以下 create-game-session-queue 示例使用两个玩家延迟策略创建一个新的游戏会话队列。第一个策略设置 100 毫秒的延迟上限，该上限在尝试放置游戏会话的第一分钟内强制执行。第二个策略将延迟上限提高到 200 毫秒，直到放置请求在 3 分钟后超时。

```
aws gamelift create-game-session-queue \
    --name MegaFrogRaceServer-NA \
    --destinations file://destinations.json \
    --player-latency-policies file://latency-policies.json \
    --timeout-in-seconds 180
```

的内容destinations.json：

```
{
    "Destinations": [
        { "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" },
        { "DestinationArn": "arn:aws:gamelift:us-east-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222" }
    ]
}
```

的内容latency-policies.json：

```
{
    "PlayerLatencyPolicies": [
        {"MaximumIndividualPlayerLatencyMilliseconds": 200},
        {"MaximumIndividualPlayerLatencyMilliseconds": 100, "PolicyDurationSeconds":
 60}
    ]
}
```

输出：

```
{
    "GameSessionQueue": {
        "Name": "MegaFrogRaceServer-NA",
        "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:111122223333:gamesessionqueue/MegaFrogRaceServer-NA",
        "TimeoutInSeconds": 600,
        "PlayerLatencyPolicies": [
            {
                "MaximumIndividualPlayerLatencyMilliseconds": 100,
                "PolicyDurationSeconds": 60
            },
            {
                "MaximumIndividualPlayerLatencyMilliseconds": 200
            }
        ]
        "Destinations": [
            {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"},
            {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"}
        ],
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的创建队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGameSessionQueue。

delete-build

以下代码示例演示了如何使用 delete-build。

Amazon CLI

删除自定义游戏生成包

以下 delete-build 示例从您的 Amazon GameLift 账户中删除生成包。删除该生成包后，您将无法使用它来创建新实例集。此操作无法撤消。

```
aws gamelift delete-build \
   --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBuild。

delete-fleet

以下代码示例演示了如何使用 delete-fleet。

Amazon CLI

删除不再使用的实例集

以下 delete-fleet 示例删除已缩减为零个实例的实例集。如果该实例集容量大于零，则请求失败并返回 HTTP 400 错误。

```
aws gamelift delete-fleet \
   --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的管理 GameLift 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFleet。

delete-game-session-queue

以下代码示例演示了如何使用 delete-game-session-queue。

Amazon CLI

删除游戏会话队列

以下 delete-game-session-queue 示例删除指定的游戏会话队列。

```
aws gamelift delete-game-session-queue \
    --name MegaFrogRace-NA
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGameSessionQueue。

describe-build

以下代码示例演示了如何使用 describe-build。

Amazon CLI

获取有关自定义游戏生成包的信息

以下 describe-build 示例检索游戏服务器生成包资源的属性。

```
aws gamelift describe-build \
    --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Build": {
        "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreationTime": 1496708916.18,
        "Name": "My_Game_Server_Build_One",
        "OperatingSystem": "AMAZON_LINUX",
        "SizeOnDisk": 1304924,
        "Status": "READY",
        "Version": "12345.678"
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的将自定义服务器生成包上传到 GameLift。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBuild。

describe-ec2-instance-limits

以下代码示例演示了如何使用 describe-ec2-instance-limits。

Amazon CLI

检索 EC2 实例类型的服务限制

以下 describe-ec2-instance-limits 示例显示当前区域中指定 EC2 实例类型允许的最大实例数和当前使用的实例数。结果表明，在允许的二十个实例中，只使用了五个。

```
aws gamelift describe-ec2-instance-limits \
    --ec2-instance-type m5.large
```

输出：

```
{
    "EC2InstanceLimits": [
        {
            "EC2InstanceType": ""m5.large",
            "CurrentInstances": 5,
            "InstanceLimit": 20
        }
    ]
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的选择计算资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEc2InstanceLimits。

describe-fleet-attributes

以下代码示例演示了如何使用 describe-fleet-attributes。

Amazon CLI

示例 1：查看实例集列表的属性

以下 describe-fleet-attributes 示例检索两个指定实例集的实例集属性。如图所示，所请求的实例集采用相同的生成包部署，一个用于按需型实例，一个用于竞价型实例，但拥有一些细微的配置差异。

```
aws gamelift describe-fleet-attributes \
    --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

输出：

```
{
    "FleetAttributes": [
        {
            "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "FleetType": "ON_DEMAND",
            "InstanceType": "c4.large",
            "Description": "On-demand hosts for v2 North America",
            "Name": "MegaFrogRaceServer.NA.v2-od",
            "CreationTime": 1568836191.995,
            "Status": "ACTIVE",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "ServerLaunchPath": "C:\\game\\MegaFrogRace_Server.exe",
            "ServerLaunchParameters": "+gamelift_start_server",
            "NewGameSessionProtectionPolicy": "NoProtection",
            "OperatingSystem": "WINDOWS_2012",
            "MetricGroups": [
                "default"
            ],
            "CertificateConfiguration": {
                "CertificateType": "DISABLED"
            }
        },
        {
            "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "FleetType": "SPOT",
            "InstanceType": "c4.large",
            "Description": "On-demand hosts for v2 North America",
            "Name": "MegaFrogRaceServer.NA.v2-spot",
            "CreationTime": 1568838275.379,
            "Status": "ACTIVATING",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
```

```
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE33333",
            "ServerLaunchPath": "C:\\game\\MegaFrogRace_Server.exe",
            "NewGameSessionProtectionPolicy": "NoProtection",
            "OperatingSystem": "WINDOWS_2012",
                "MetricGroups": [
                "default"
            ],
            "CertificateConfiguration": {
                "CertificateType": "GENERATED"
            }
        }
    ]
}
```

示例 2：请求所有实例集的属性

以下 describe-fleet-attributes 返回具有任何状态的所有实例集的实例集属性。此示例演示如何使用分页参数一次返回一个实例集。

```
aws gamelift describe-fleet-attributes \
    --limit 1
```

输出：

```
{
    "FleetAttributes": [
        {
            "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222",
            "FleetType": "SPOT",
            "InstanceType": "c4.large",
            "Description": "On-demand hosts for v2 North America",
            "Name": "MegaFrogRaceServer.NA.v2-spot",
            "CreationTime": 1568838275.379,
            "Status": "ACTIVATING",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE33333",
            "ServerLaunchPath": "C:\\game\\MegaFrogRace_Server.exe",
            "NewGameSessionProtectionPolicy": "NoProtection",
            "OperatingSystem": "WINDOWS_2012",
```

```
            "MetricGroups": [
                "default"
            ],
            "CertificateConfiguration": {
                "CertificateType": "GENERATED"
            }
        }
    ],
    "NextToken":
 "eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS
}
```

该输出包含一个 NextToken 值，您可以在第二次调用该命令时使用该值。将该值传递给 --next-token 参数，以指定从何处获取输出。以下命令在输出中返回第二个结果。

```
aws gamelift describe-fleet-attributes \
    --limit 1 \
    --next-
token eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZT
```

重复操作，直到响应不包含 NextToken 值。

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的设置 GameLift 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetAttributes。

describe-fleet-capacity

以下代码示例演示了如何使用 describe-fleet-capacity。

Amazon CLI

查看实例集列表的容量状态

以下 describe-fleet-capacity 示例检索两个指定实例集的当前容量。

```
aws gamelift describe-fleet-capacity \
    --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111 fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

输出：

```
{
    "FleetCapacity": [
        {
            "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "InstanceType": "c5.large",
            "InstanceCounts": {
                "DESIRED": 10,
                "MINIMUM": 1,
                "MAXIMUM": 20,
                "PENDING": 0,
                "ACTIVE": 10,
                "IDLE": 3,
                "TERMINATING": 0
            }
        },
        {
            "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "InstanceType": "c5.large",
            "InstanceCounts": {
                "DESIRED": 13,
                "MINIMUM": 1,
                "MAXIMUM": 20,
                "PENDING": 0,
                "ACTIVE": 15,
                "IDLE": 2,
                "TERMINATING": 2
            }
        }

    ]
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的 GameLift 实例集指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetCapacity。

`describe-fleet-events`

以下代码示例演示了如何使用 `describe-fleet-events`。

Amazon CLI

请求指定时间范围的事件

以下 describe-fleet-events 示例显示在指定时间范围内发生的所有实例集相关事件的详细信息。

```
aws gamelift describe-fleet-events \
    --fleet-id arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --start-time 1579647600 \
    --end-time 1579649400 \
    --limit 5
```

输出：

```
{
    "Events": [
        {
            "EventId": "a37b6892-5d07-4d3b-8b47-80244ecf66b9",
            "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "EventCode": "FLEET_STATE_ACTIVE",
            "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
 state to ACTIVE",
            "EventTime": 1579649342.191
        },
        {
            "EventId": "67da4ec9-92a3-4d95-886a-5d6772c24063",
            "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "EventCode": "FLEET_STATE_ACTIVATING",
            "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
 state to ACTIVATING",
            "EventTime": 1579649321.427
        },
        {
            "EventId": "23813a46-a9e6-4a53-8847-f12e6a8381ac",
            "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "EventCode": "FLEET_STATE_BUILDING",
            "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
 state to BUILDING",
            "EventTime": 1579649321.243
        },
        {
            "EventId": "3bf217d0-1d44-42f9-9202-433ed475d2e8",
            "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "EventCode": "FLEET_STATE_VALIDATING",
```

```
            "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
 state to VALIDATING",
            "EventTime": 1579649197.449
        },
        {
            "EventId": "2ecd0130-5986-44eb-99a7-62df27741084",
            "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "EventCode": "FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND",
            "Message": "Failed to find a valid path",
            "EventTime": 1569319075.839,
            "PreSignedLogUrl": "https://gamelift-event-logs-prod-
us-west-2.s3.us-west-2.amazonaws.com/logs/fleet-83422059-8329-42a2-
a4d6-c4444386a6f8/events/2ecd0130-5986-44eb-99a7-62df27741084/
FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND.txt?X-Amz-Security-
Token=IQoJb3JpZ2luX2VjEB8aCXVzLXdlc3QtMiJHMEUCIHV5K%2FLPx8h310D
%2FAvx0%2FZxsDy5XA3cJOwPdu3T0eBa%2FAiEA1yovokcZYy%2FV4CWW6l26aFyiSHO
%2Bxz%2FBMAhEHYHMQNcqkQMImP%2F%2F%2F%2F%2F%2F%2F%2F%2F
%2FARAAGgw3NDEwNjE1OTIxNzEiDI8rsZtzLzlwEDQhXSrlAtl5Ae
%2Fgo6FCIzqXPbXfBOnSvFYqeDlriZarEpKqKrUt8mXQv9iqHResqCph9AKo49lwgSYTT2QoSxnrD7%2FUgv
%2BZm2pVuczvuKtUA0fcx6s0GxpjIAzdIE%2F5P%2FB7B9M%2BVZ
%2F9KF82hbJi0HTE6Y7BjKsEgFCvk4UXILhfjtan9iQl8%2F21ZTurAcJbm7Y5tuLF9SWSK3%2BEa7VXOcCK4D4O1sMj
%2FIaXoHkNvg0RVTa0hIqdvpaDQlsSBNdqTXbjHTu6fETE9Y9Ky%2BiJK5KiUG
%2F59GjCpDcvS1FqKeLUEmKT7wysGmvjMc2n%2Fr
%2F9VxQfte7w9srXwlLAQuwhiXAAyI5ICMZ5JvzjzQwTqD4CHTVKUUDwL
%2BRZzbuuqkJObZml02CkRGp%2B74RTAzLbWptVqZTIfzctiCTmWxb
%2FmKyELRYsVLrwNJ%2BGJ7%2BCrN0RC%2FjlgfLYIZyeAqjPgAu5HjgX
%2BM7jCo9M7wBTrnAXKOFQuf9dvA84SuwXOJFp17LYGjrHMKv0qC3GfbTMrZ6kzeNV9awKCpXB2Gnx9z2KvIlJdqirWV
%2F9C6%2B4jIZPME3jXmZcEHqqw5uvAVF7aeIavtUZU8pxpDIWT0YE4p3Kriy2AA7ziCRKtVfjV839InyLk8LUjsioWK
%2BYUq8%2FDTlLxqj1S%2FiO4TI0Wo7ilAo%2FKKWWF4guuNDexj8EOOynSp1yImB
%2BZf2Fua3O44W4eEXAMPLE33333&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170621T231808Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20170621%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-
Signature=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
        }
    ],
    "NextToken":
 "eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的调试 GameLift 实例集问题。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetEvents。

describe-fleet-port-settings

以下代码示例演示了如何使用 describe-fleet-port-settings。

Amazon CLI

查看实例集的入站连接权限

以下 describe-fleet-port-settings 示例检索指定实例集的连接设置。

```
aws gamelift describe-fleet-port-settings \
    --fleet-id arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "InboundPermissions": [
        {
            "FromPort": 33400,
            "ToPort": 33500,
            "IpRange": "0.0.0.0/0",
            "Protocol": "UDP"
        },
        {
            "FromPort": 1900,
            "ToPort": 2000,
            "IpRange": "0.0.0.0/0",
            "Protocol": "TCP"
        }
    ]
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的设置 GameLift 实例集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetPortSettings。

describe-fleet-utilization

以下代码示例演示了如何使用 describe-fleet-utilization。

Amazon CLI

示例 1：查看实例集列表的使用情况数据

以下 describe-fleet-utilization 示例检索一个指定实例集的当前使用情况信息。

```
aws gamelift describe-fleet-utilization \
    --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "FleetUtilization": [
        {
        "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ActiveServerProcessCount": 100,
        "ActiveGameSessionCount": 62,
        "CurrentPlayerSessionCount": 329,
        "MaximumPlayerSessionCount": 1000
        }
    ]
}
```

示例 2：请求所有实例集的使用情况数据

以下 describe-fleet-utilization 返回处于任何状态的所有实例集的实例集使用情况数据。此示例使用分页参数一次返回两个实例集的数据。

```
aws gamelift describe-fleet-utilization \
    --limit 2
```

输出：

```
{
    "FleetUtilization": [
        {
            "FleetId": "fleet-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
            "ActiveServerProcessCount": 100,
            "ActiveGameSessionCount": 13,
```

```
            "CurrentPlayerSessionCount": 98,
            "MaximumPlayerSessionCount": 1000
        },
        {
            "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
            "ActiveServerProcessCount": 100,
            "ActiveGameSessionCount": 62,
            "CurrentPlayerSessionCount": 329,
            "MaximumPlayerSessionCount": 1000
        }
    ],
    "NextToken":
 "eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS
}
```

再次调用该命令，将 NextToken 值作为参数传递给 --next-token 参数，以查看接下来的两个结果。

```
aws gamelift describe-fleet-utilization \
    --limit 2 \
    --next-
token eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZT
```

重复操作，直到响应在输出中不再包含 NextToken 值。

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的 GameLift 实例集指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFleetUtilization。

describe-game-session-queues

以下代码示例演示了如何使用 describe-game-session-queues。

Amazon CLI

查看游戏会话队列

以下 describe-game-session-queues 示例检索两个指定队列的属性。

```
aws gamelift describe-game-session-queues \
    --names MegaFrogRace-NA MegaFrogRace-EU
```

输出：

```
{
    "GameSessionQueues": [{
            "Destinations": [{
                    "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                },
                {
                    "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
                }
            ],
            "Name": "MegaFrogRace-NA",
            "TimeoutInSeconds": 600,
            "GameSessionQueueArn": "arn:aws:gamelift:us-west-2::gamesessionqueue/
MegaFrogRace-NA",
            "PlayerLatencyPolicies": [{
                    "MaximumIndividualPlayerLatencyMilliseconds": 200
                },
                {
                    "MaximumIndividualPlayerLatencyMilliseconds": 100,
                    "PolicyDurationSeconds": 60
                }
            ],
            "FilterConfiguration": {
                "AllowedLocations": ["us-west-2", "ap-south-1", "us-east-1"]
            },
            "PriorityConfiguration": {
                "PriorityOrder": ["LOCATION", "FLEET_TYPE", "DESTINATION"],
                "LocationOrder": ["us-west-2", "ap-south-1", "us-east-1"]
            }
        },
        {
            "Destinations": [{
                "DestinationArn": "arn:aws:gamelift:eu-west-3::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
            }],
            "Name": "MegaFrogRace-EU",
            "TimeoutInSeconds": 600,
            "GameSessionQueueArn": "arn:aws:gamelift:us-west-2::gamesessionqueue/
MegaFrogRace-EU"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的使用多区域队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGameSessionQueues。

describe-runtime-configuration

以下代码示例演示了如何使用 describe-runtime-configuration。

Amazon CLI

请求实例集的运行时配置

以下 describe-runtime-configuration 示例检索有关指定实例集的当前运行时配置的详细信息。

```
aws gamelift describe-runtime-configuration \
    --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RuntimeConfiguration": {
        "ServerProcesses": [
            {
                "LaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
                "Parameters": "+gamelift_start_server",
                "ConcurrentExecutions": 3
            },
            {
                "LaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
                "Parameters": "+gamelift_start_server +debug",
                "ConcurrentExecutions": 1
            }
        ],
        "MaxConcurrentGameSessionActivations": 2147483647,
        "GameSessionActivationTimeoutSeconds": 300
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的在实例集上运行多个进程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRuntimeConfiguration。

list-builds

以下代码示例演示了如何使用 list-builds。

Amazon CLI

示例 1：获取自定义游戏生成包列表

以下 list-builds 示例检索当前区域中所有游戏服务器生成包的属性。示例请求说明了如何使用分页参数 Limit 和 NextToken 来检索顺序集中的结果。第一个命令检索前两个生成包。由于有两个以上结果可用，因此响应包括一个 NextToken，以指示有更多结果可用。

```
aws gamelift list-builds \
    --limit 2
```

输出：

```
{
    "Builds": [
        {
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "CreationTime": 1495664528.723,
            "Name": "My_Game_Server_Build_One",
            "OperatingSystem": "WINDOWS_2012",
            "SizeOnDisk": 8567781,
            "Status": "READY",
            "Version": "12345.678"
        },
        {
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "CreationTime": 1495528748.555,
            "Name": "My_Game_Server_Build_Two",
            "OperatingSystem": "AMAZON_LINUX_2",
            "SizeOnDisk": 8567781,
            "Status": "FAILED",
```

```
            "Version": "23456.789"
        }
    ],
    "NextToken":
 "eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS
}
```

然后，您可以使用 `--next-token` 参数再次调用该命令，如下所示，以查看接下来的两个生成包。

```
aws gamelift list-builds \
    --limit 2
    --next-
token eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZT
```

重复操作，直到响应不包含 NextToken 值。

示例 2：获取处于失败状态的自定义游戏生成包的列表

以下 `list-builds` 示例检索当前区域中当前状态为 FAILED 的所有游戏服务器生成包的属性。

```
aws gamelift list-builds \
    --status FAILED
```

输出：

```
{
    "Builds": [
        {
            "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222",
            "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "CreationTime": 1495528748.555,
            "Name": "My_Game_Server_Build_Two",
            "OperatingSystem": "AMAZON_LINUX_2",
            "SizeOnDisk": 8567781,
            "Status": "FAILED",
            "Version": "23456.789"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuilds。

list-fleets

以下代码示例演示了如何使用 list-fleets。

Amazon CLI

示例 1：获取一个区域中所有实例集的列表

以下 list-fleets 示例显示当前区域中所有实例集的实例集 ID。此示例使用分页参数一次检索两个实例集 ID。响应包含一个 next-token 属性，表示还有更多结果要检索。

```
aws gamelift list-fleets \
    --limit 2
```

输出：

```
{
    "FleetIds": [
        "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
    ],
    "NextToken":
 "eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS
}
```

您可以在下一个命令中传递上一个响应的 NextToken 值，如此处所示，以获取接下来的两个结果。

```
aws gamelift list-fleets \
    --limit 2 \
    --next-
token eyJhd3NBY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC00NDRlZj
```

示例 2：获取具有特定生成包或脚本的区域内所有实例集的列表

以下 list-builds 示例检索使用指定游戏生成包部署的实例集的 ID。如果您正在使用实时服务器，则可以提供脚本 ID 来代替生成包 ID。由于此示例未指定限制参数，因此结果最多可以包含 16 个实例集 ID。

```
aws gamelift list-fleets \
    --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "FleetIds": [
        "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE44444"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFleets。

request-upload-credentials

以下代码示例演示了如何使用 request-upload-credentials。

Amazon CLI

刷新用于上传生成包的访问凭证

以下 create-build 示例获取用于将 GameLift 生成包文件上传到 Amazon S3 位置的新的有效访问凭证。凭证具有有限的生命周期。您可以从对原始 CreateBuild 请求的响应中获取生成包 ID。

```
aws gamelift request-upload-credentials \
    --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "StorageLocation": {
        "Bucket": "gamelift-builds-us-west-2",
        "Key": "123456789012/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "UploadCredentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AgoGb3JpZ2luENz...EXAMPLETOKEN=="
    }
```

```
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的将自定义服务器生成包上传到 GameLift。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestUploadCredentials。

start-fleet-actions

以下代码示例演示了如何使用 start-fleet-actions。

Amazon CLI

重启实例集自动扩缩活动

以下 start-fleet-actions 示例恢复使用为指定实例集定义但已通过调用“stop-fleet-actions”停止的所有扩缩策略。启动后，扩缩策略会立即开始跟踪其各自的指标。

```
aws gamelift start-fleet-actions \
    --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --actions AUTO_SCALING
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartFleetActions。

stop-fleet-actions

以下代码示例演示了如何使用 stop-fleet-actions。

Amazon CLI

停止实例集的自动扩缩活动

以下 stop-fleet-actions 示例停止使用为指定实例集定义的所有扩缩策略。策略暂停后，除非您手动调整实例集容量，否则实例集容量将保持不变。

```
aws gamelift start-fleet-actions \
    --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --actions AUTO_SCALING
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopFleetActions。

update-build

以下代码示例演示了如何使用 update-build。

Amazon CLI

更新自定义游戏生成包

以下 update-build 示例更改与指定生成包资源关联的名称和版本信息。返回的生成包对象验证更改是否已成功完成。

```
aws gamelift update-build \
    --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --name MegaFrogRaceServer.NA.east \
    --build-version 12345.east
```

输出：

```
{
    "Build": {
        "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
        "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "CreationTime": 1496708916.18,
        "Name": "MegaFrogRaceServer.NA.east",
        "OperatingSystem": "AMAZON_LINUX_2",
        "SizeOnDisk": 1304924,
        "Status": "READY",
        "Version": "12345.east"
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的更新您的生成包文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBuild。

update-game-session-queue

以下代码示例演示了如何使用 update-game-session-queue。

Amazon CLI

更新游戏会话队列配置

以下 update-game-session-queue 示例添加一个新的目标并更新现有游戏会话队列的玩家延迟策略。

```
aws gamelift update-game-session-queue \
    --name MegaFrogRace-NA \
    --destinations file://destinations.json \
    --player-latency-policies file://latency-policies.json
```

的内容destinations.json：

```
{
    "Destinations": [
        {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d"},
        {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/
fleet-5c6d3c4d-5e6f-7a8b-9c0d-1e2f3a4b5a2b"},
        {"DestinationArn": "arn:aws:gamelift:us-east-1::alias/
alias-11aa22bb-3c4d-5e6f-000a-1111aaaa22bb"}
    ]
}
```

的内容latency-policies.json：

```
{
    "PlayerLatencyPolicies": [
        {"MaximumIndividualPlayerLatencyMilliseconds": 200},
        {"MaximumIndividualPlayerLatencyMilliseconds": 150, "PolicyDurationSeconds":
 120},
        {"MaximumIndividualPlayerLatencyMilliseconds": 100, "PolicyDurationSeconds":
 120}
    ]
}
```

输出：

```
{
    "GameSessionQueue": {
```

```
        "Destinations": [
            {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d"},
            {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/
fleet-5c6d3c4d-5e6f-7a8b-9c0d-1e2f3a4b5a2b"},
            {"DestinationArn": "arn:aws:gamelift:us-east-1::alias/
alias-11aa22bb-3c4d-5e6f-000a-1111aaaa22bb"}
        ],
        "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:111122223333:gamesessionqueue/MegaFrogRace-NA",
        "Name": "MegaFrogRace-NA",
        "TimeoutInSeconds": 600,
        "PlayerLatencyPolicies": [
            {"MaximumIndividualPlayerLatencyMilliseconds": 200},
            {"MaximumIndividualPlayerLatencyMilliseconds": 150,
 "PolicyDurationSeconds": 120},
            {"MaximumIndividualPlayerLatencyMilliseconds": 100,
 "PolicyDurationSeconds": 120}
        ]
    }
}
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的使用多区域队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGameSessionQueue。

upload-build

以下代码示例演示了如何使用 upload-build。

Amazon CLI

示例 1：上传 Linux 游戏服务器生成包

以下 upload-build 示例将 Linux 游戏服务器生成包文件从文件目录上传到 GameLift 服务并创建生成包资源。

```
aws gamelift upload-build \
    --name MegaFrogRaceServer.NA \
    --build-version 2.0.1 \
    --build-root ~/MegaFrogRace_Server/release-na \
    --operating-system AMAZON_LINUX_2
    --server-sdk-version 4.0.2
```

输出：

```
Uploading ~/MegaFrogRace_Server/release-na:  16.0 KiB / 74.6 KiB (21.45%)
Uploading ~/MegaFrogRace_Server/release-na:  32.0 KiB / 74.6 KiB (42.89%)
Uploading ~/MegaFrogRace_Server/release-na:  48.0 KiB / 74.6 KiB (64.34%)
Uploading ~/MegaFrogRace_Server/release-na:  64.0 KiB / 74.6 KiB (85.79%)
Uploading ~/MegaFrogRace_Server/release-na:  74.6 KiB / 74.6 KiB (100.00%)
Successfully uploaded ~/MegaFrogRace_Server/release-na to AWS GameLift
Build ID: build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

示例 2：上传 Windows 游戏服务器生成包

以下 upload-build 示例将 Windows 游戏服务器生成包文件从目录上传到 GameLift 服务并创建生成包记录。

```
aws gamelift upload-build \
    --name MegaFrogRaceServer.NA \
    --build-version 2.0.1 \
    --build-root C:\MegaFrogRace_Server\release-na \
    --operating-system WINDOWS_2012
    --server-sdk-version 4.0.2
```

输出：

```
Uploading C:\MegaFrogRace_Server\release-na:  16.0 KiB / 74.6 KiB (21.45%)
Uploading C:\MegaFrogRace_Server\release-na:  32.0 KiB / 74.6 KiB (42.89%)
Uploading C:\MegaFrogRace_Server\release-na:  48.0 KiB / 74.6 KiB (64.34%)
Uploading C:\MegaFrogRace_Server\release-na:  64.0 KiB / 74.6 KiB (85.79%)
Uploading C:\MegaFrogRace_Server\release-na:  74.6 KiB / 74.6 KiB (100.00%)
Successfully uploaded C:\MegaFrogRace_Server\release-na to AWS GameLift
Build ID: build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

有关更多信息，请参阅《Amazon GameLift 开发人员指南》中的将自定义服务器生成包上传到 GameLift。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadBuild。

使用Amazon CLI 的 Amazon Glacier 示例

以下代码示例演示了如何通过将 Amazon Command Line Interface 与 Amazon Glacier 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

abort-multipart-upload

以下代码示例演示了如何使用 abort-multipart-upload。

Amazon CLI

以下命令删除到名为 my-vault 的文件库的正在进行的分段上传：

```
aws glacier abort-multipart-upload --account-id - --vault-name my-vault
 --upload-id 19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ
```

此命令不生成任何输出。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。上传 ID 由 aws glacier initiate-multipart-upload 命令返回，也可以使用 aws glacier list-multipart-uploads 获取它。

有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortMultipartUpload。

abort-vault-lock

以下代码示例演示了如何使用 abort-vault-lock。

Amazon CLI

中止正在进行的文件库锁定过程

以下 abort-vault-lock 示例从指定的文件库中删除文件库锁定策略，并将文件库锁定的锁定状态重置为已解锁。

```
aws glacier abort-vault-lock \
    --account-id - \
    --vault-name MyVaultName
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Glacier API 开发人员指南》中的中止文件库锁定（DELETE 锁定策略）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortVaultLock。

add-tags-to-vault

以下代码示例演示了如何使用 add-tags-to-vault。

Amazon CLI

以下命令向名为 my-vault 的文件库中添加两个标签：

```
aws glacier add-tags-to-vault --account-id - --vault-name my-vault --
tags id=1234,date=july2015
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToVault。

complete-multipart-upload

以下代码示例演示了如何使用 complete-multipart-upload。

Amazon CLI

以下命令完成 3 MiB 归档的分段上传：

```
aws glacier complete-multipart-upload --archive-size 3145728 --
checksum 9628195fcdbcbbe76cdde456d4646fa7de5f219fb39823836d81f0cc0e18aa67
 --upload-id 19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ --account-id - --vault-name my-vault
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

上传 ID 由 `aws glacier initiate-multipart-upload` 命令返回，也可以使用 `aws glacier list-multipart-uploads` 获取它。校验和参数采用十六进制归档的 SHA-256 树形哈希。

有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，包括有关计算树形哈希的说明，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteMultipartUpload。

complete-vault-lock

以下代码示例演示了如何使用 `complete-vault-lock`。

Amazon CLI

完成正在进行的文件库锁定过程

以下 `complete-vault-lock` 示例完成指定文件库的正在进行的锁定进程，并将文件库锁定的锁定状态设置为 `Locked`。运行 `initiate-lock-process` 时，您将获得 `lock-id` 参数的值。

```
aws glacier complete-vault-lock \
    --account-id - \
    --vault-name MyVaultName \
    --lock-id 9QZgEXAMPLEPhvL6xEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Glacier API 开发人员指南》中的完成文件库锁定（POST lockId）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteVaultLock。

create-vault

以下代码示例演示了如何使用 `create-vault`。

Amazon CLI

以下命令创建名为 `my-vault` 的新文件库：

```
aws glacier create-vault --vault-name my-vault --account-id -
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVault。

delete-archive

以下代码示例演示了如何使用 delete-archive。

Amazon CLI

从文件库中删除存档

以下 delete-archive 示例从 example_vault 中删除指定存档。

```
aws glacier delete-archive \
    --account-id 111122223333  \
    --vault-name example_vault \
    --archive-id Sc0u9ZP8yaWkmh-XGlIvAVprtLhaLCGnNwNl5I5x9HqPIkX5mjc0DrId3Ln-
Gi_k2HzmlIDZUz117KSdVMdMXLuFWi9PJUitxWO73edQ43eTlMWkH0pd9zVSAuV_XXZBVhKhyGhJ7w
```

此命令不生成任何输出。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteArchive。

delete-vault-access-policy

以下代码示例演示了如何使用 delete-vault-access-policy。

Amazon CLI

删除文件库的访问策略

以下 delete-vault-access-policy 示例删除指定文件库的访问策略。

```
aws glacier delete-vault-access-policy \
    --account-id 111122223333 \
    --vault-name example_vault
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVaultAccessPolicy。

delete-vault-notifications

以下代码示例演示了如何使用 delete-vault-notifications。

Amazon CLI

删除文件库的 SNS 通知

以下 delete-vault-notifications 示例演示了如何删除 Amazon Simple Notification Service（Amazon SNS）针对指定文件库发送的通知。

```
aws glacier delete-vault-notifications \
    --account-id 111122223333 \
    --vault-name example_vault
```

此命令不生成任何输出。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVaultNotifications。

delete-vault

以下代码示例演示了如何使用 delete-vault。

Amazon CLI

以下命令删除名为 my-vault 的文件库：

```
aws glacier delete-vault --vault-name my-vault --account-id -
```

此命令不生成任何输出。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVault。

describe-job

以下代码示例演示了如何使用 describe-job。

Amazon CLI

以下命令检索名为 my-vault 的文件库中有关库存检索作业的信息：

```
aws glacier describe-job --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW
```

输出：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
    "Completed": false,
    "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "Action": "InventoryRetrieval",
    "CreationDate": "2015-07-17T20:23:41.616Z",
    "StatusCode": "InProgress"
}
```

作业 ID 可以在 aws glacier initiate-job 和 aws glacier list-jobs 的输出中找到。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJob。

describe-vault

以下代码示例演示了如何使用 describe-vault。

Amazon CLI

以下命令检索名为 my-vault 的文件库的相关数据：

```
aws glacier describe-vault --vault-name my-vault --account-id -
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeVault。

get-data-retrieval-policy

以下代码示例演示了如何使用 get-data-retrieval-policy。

Amazon CLI

以下命令获取正在使用中的账户的数据检索策略：

```
aws glacier get-data-retrieval-policy --account-id -
```

输出：

```
{
    "Policy": {
        "Rules": [
            {
                "BytesPerHour": 10737418240,
                "Strategy": "BytesPerHour"
            }
        ]
    }
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataRetrievalPolicy。

get-job-output

以下代码示例演示了如何使用 get-job-output。

Amazon CLI

以下命令将文件库清单作业的输出保存到名为 output.json 的当前目录中的某个文件中：

```
aws glacier get-job-output --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW output.json
```

可在 aws glacier list-jobs 输出中找到 job-id。请注意，输出文件名是一个位置参数，不以选项名称作为前缀。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

输出：

```
{
    "status": 200,
    "acceptRanges": "bytes",
    "contentType": "application/json"
}
```

output.json:

```
{"VaultARN":"arn:aws:glacier:us-west-2:0123456789012:vaults/
my-vault","InventoryDate":"2015-04-07T00:26:18Z","ArchiveList":
[{"ArchiveId":"kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw","ArchiveDescription":"multipart
 upload
 test","CreationDate":"2015-04-06T22:24:34Z","Size":3145728,"SHA256TreeHash":"9628195fcdbcbb
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobOutput。

get-vault-access-policy

以下代码示例演示了如何使用 get-vault-access-policy。

Amazon CLI

检索文件库的访问策略

以下 get-vault-access-policy 示例检索指定文件库的访问策略。

```
aws glacier get-vault-access-policy \
    --account-id 111122223333 \
    --vault-name example_vault
```

输出：

```
{
```

```
    "policy": {
        "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::444455556666:root
\"},\"Action\":\"glacier:ListJobs\",\"Resource\":\"arn:aws:glacier:us-
east-1:111122223333:vaults/example_vault\"},{\"Effect\":\"Allow\",\"Principal\":
{\"AWS\":\"arn:aws:iam::444455556666:root\"},\"Action\":\"glacier:UploadArchive\",
\"Resource\":\"arn:aws:glacier:us-east-1:111122223333:vaults/example_vault\"}]}"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVaultAccessPolicy。

get-vault-lock

以下代码示例演示了如何使用 get-vault-lock。

Amazon CLI

获取文件库锁定的详细信息

以下 get-vault-lock 示例检索了有关指定文件库的锁定的详细信息。

```
aws glacier get-vault-lock \
    --account-id - \
    --vault-name MyVaultName
```

输出：

```
{
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-
lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam::999999999999:root
\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-
west-2:99999999999:vaults/MyVaultName\",\"Condition\":{\"NumericLessThanEquals\":
{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}",
    "State": "Locked",
    "CreationDate": "2019-07-29T22:25:28.640Z"
}
```

有关更多信息，请参阅《Amazon Glacier API 开发人员指南》中的获取文件库锁定（GET 锁定策略）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVaultLock。

get-vault-notifications

以下代码示例演示了如何使用 get-vault-notifications。

Amazon CLI

以下命令获取名为 my-vault 的文件库的通知配置的描述：

```
aws glacier get-vault-notifications --account-id - --vault-name my-vault
```

输出：

```
{
    "vaultNotificationConfig": {
        "Events": [
            "InventoryRetrievalCompleted",
            "ArchiveRetrievalCompleted"
        ],
        "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault"
    }
}
```

如果没有为该文件库配置任何通知，则会返回错误。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 GetVaultNotifications。

initiate-job

以下代码示例演示了如何使用 initiate-job。

Amazon CLI

以下命令启动作业以获取文件库 my-vault 的清单：

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
 '{"Type": "inventory-retrieval"}'
```

输出：

```
{
```

```
    "location": "/0123456789012/vaults/my-vault/jobs/
zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "jobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW"
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

以下命令启动一个作业以从文件库 `my-vault` 检索归档：

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-
parameters file://job-archive-retrieval.json
```

`job-archive-retrieval.json` 是本地文件夹中的一个 JSON 文件，用于指定作业类型、归档 ID 和一些可选参数：

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
  "Description": "Retrieve archive on 2015-07-17",
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-topic"
}
```

归档 ID 可在 `aws glacier upload-archive` 和 `aws glacier get-job-output` 的输出中找到。

输出：

```
{
    "location": "/011685312445/vaults/mwunderl/jobs/l7IL5-
EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
    "jobId": "l7IL5-EkXy2O5uLYaFdAYOiEY9Ws95fClzIbk-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav"
}
```

有关作业参数格式的详细信息，请参阅《Amazon Glacier API Reference》中的“启动作业”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InitiateJob。

initiate-multipart-upload

以下代码示例演示了如何使用 initiate-multipart-upload。

Amazon CLI

以下命令将启动到名为 my-vault 的文件库的分段上传，其中每个文件的分段大小为 1 MiB（1024 x 1024 字节）：

```
aws glacier initiate-multipart-upload --account-id - --part-size 1048576 --vault-name my-vault --archive-description "multipart upload test"
```

归档描述参数是可选的。Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

成功时，此命令会输出一个上传 ID。使用 aws glacier upload-multipart-part 上传归档的每个分段时，请使用此上传 ID。有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InitiateMultipartUpload。

initiate-vault-lock

以下代码示例演示了如何使用 initiate-vault-lock。

Amazon CLI

启动文件库锁定过程

以下 initiate-vault-lock 示例在指定的文件库上安装文件库锁定策略，并将文件库锁定的锁定状态设置为 InProgress。您必须通过在 24 小时内调用 complete-vault-lock 以将文件库锁定的状态设置为 Locked 来完成该过程。

```
aws glacier initiate-vault-lock \
    --account-id - \
    --vault-name MyVaultName \
    --policy file://vault_lock_policy.json
```

的内容vault_lock_policy.json：

```
{"Policy":"{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam::999999999999:root\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-west-2:999999999999:vaults/examplevault\",\"Condition\":{\"NumericLessThanEquals\":{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}"}
```

该输出是可用于完成文件库锁定过程的文件库锁定 ID。

```
{
    "lockId": "9QZgEXAMPLEPhvL6xEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Glacier API 开发人员指南》中的启动文件库锁定（POST 锁定策略）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InitiateVaultLock。

list-jobs

以下代码示例演示了如何使用 list-jobs。

Amazon CLI

以下命令列出了名为 my-vault 的文件库的正在进行和最近完成的作业：

```
aws glacier list-jobs --account-id - --vault-name my-vault
```

输出：

```
{
    "JobList": [
        {
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
            "RetrievalByteRange": "0-3145727",
            "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
            "Completed": false,
            "SHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
```

```
            "JobId": "l7IL5-EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
            "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
            "JobDescription": "Retrieve archive on 2015-07-17",
            "ArchiveSizeInBytes": 3145728,
            "Action": "ArchiveRetrieval",
            "ArchiveSHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
            "CreationDate": "2015-07-17T21:16:13.840Z",
            "StatusCode": "InProgress"
        },
        {
            "InventoryRetrievalParameters": {
                "Format": "JSON"
            },
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
            "Completed": false,
            "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
            "Action": "InventoryRetrieval",
            "CreationDate": "2015-07-17T20:23:41.616Z",
            "StatusCode": ""InProgress""
        }
    ]
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

list-multipart-uploads

以下代码示例演示了如何使用 list-multipart-uploads。

Amazon CLI

以下命令显示名为 my-vault 的文件库的所有正在进行的分段上传：

```
aws glacier list-multipart-uploads --account-id - --vault-name my-vault
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMultipartUploads。

`list-parts`

以下代码示例演示了如何使用 `list-parts`。

Amazon CLI

以下命令列出了到名为 `my-vault` 的文件库的分段上传的上传分段：

```
aws glacier list-parts --account-id - --vault-name my-vault --upload-id "SYZi7qnL-YGqGwAm8Kn3BLP2ElNCvnB-5961R09CSaPmPwkYGHOqeN_nX3-Vhnd2yF0KfB5FkmbnBU9GubbdrCs8ut-D"
```

输出：

```
{
    "MultipartUploadId": "SYZi7qnL-
YGqGwAm8Kn3BLP2ElNCvnB-5961R09CSaPmPwkYGHOqeN_nX3-Vhnd2yF0KfB5FkmbnBU9GubbdrCs8ut-
D",
    "Parts": [
        {
            "RangeInBytes": "0-1048575",
            "SHA256TreeHash":
 "e1f2a7cd6e047350f69b9f8cfa60fa606fe2f02802097a9a026360a7edc1f553"
        },
        {
            "RangeInBytes": "1048576-2097151",
            "SHA256TreeHash":
 "43cf3061fb95796aed99a11a6aa3cd8f839eed15e655ab0a597126210636aee6"
        }
    ],
    "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
    "CreationDate": "2015-07-18T00:05:23.830Z",
    "PartSizeInBytes": 1048576
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListParts。

`list-provisioned-capacity`

以下代码示例演示了如何使用 `list-provisioned-capacity`。

Amazon CLI

检索预置容量单位

以下 `list-provisioned-capacity` 示例检索指定账户的任何预置容量单位的详细信息。

```
aws glacier list-provisioned-capacity \
    --account-id 111122223333
```

输出：

```
{
    "ProvisionedCapacityList": [
        {
            "CapacityId": "HpASAuvfRFiVDbOjMfEIcr8K",
            "ExpirationDate": "2020-03-18T19:59:24.000Z",
            "StartDate": "2020-02-18T19:59:24.912Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProvisionedCapacity。

`list-tags-for-vault`

以下代码示例演示了如何使用 `list-tags-for-vault`。

Amazon CLI

以下命令列出应用于名为 `my-vault` 的文件库的标签：

```
aws glacier list-tags-for-vault --account-id - --vault-name my-vault
```

输出：

```
{
    "Tags": {
        "date": "july2015",
        "id": "1234"
    }
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForVault。

`list-vaults`

以下代码示例演示了如何使用 `list-vaults`。

Amazon CLI

以下命令列出默认账户和区域中的文件库：

```
aws glacier list-vaults --account-id -
```

输出：

```
{
    "VaultList": [
        {
            "SizeInBytes": 3178496,
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
            "LastInventoryDate": "2015-04-07T00:26:19.028Z",
            "VaultName": "my-vault",
            "NumberOfArchives": 1,
            "CreationDate": "2015-04-06T21:23:45.708Z"
        }
    ]
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListVaults。

purchase-provisioned-capacity

以下代码示例演示了如何使用 purchase-provisioned-capacity。

Amazon CLI

购买预置容量单位

以下 purchase-provisioned-capacity 示例购买预置容量单位。

```
aws glacier purchase-provisioned-capacity \
    --account-id 111122223333
```

输出：

```
{
    "capacityId": "HpASAuvfRFiVDbOjMfEIcr8K"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseProvisionedCapacity。

remove-tags-from-vault

以下代码示例演示了如何使用 remove-tags-from-vault。

Amazon CLI

以下命令从名为 my-vault 的文件库中删除键为 date 的标签：

```
aws glacier remove-tags-from-vault --account-id - --vault-name my-vault --tag-
keys date
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromVault。

set-data-retrieval-policy

以下代码示例演示了如何使用 set-data-retrieval-policy。

Amazon CLI

以下命令为正在使用的账户配置数据检索策略：

```
aws glacier set-data-retrieval-policy --account-id - --policy file://data-retrieval-policy.json
```

data-retrieval-policy.json 是当前文件夹中指定数据检索策略的 JSON 文件：

```
{
  "Rules":[
     {
         "Strategy":"BytesPerHour",
         "BytesPerHour":10737418240
      }
   ]
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

以下命令使用内联 JSON 将数据检索策略设置为 FreeTier：

```
aws glacier set-data-retrieval-policy --account-id - --policy '{"Rules":[{"Strategy":"FreeTier"}]}'
```

有关策略格式的详细信息，请参阅《Amazon Glacier API 参考》中的“设置数据检索策略”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetDataRetrievalPolicy。

set-vault-access-policy

以下代码示例演示了如何使用 set-vault-access-policy。

Amazon CLI

设置文件库的访问策略

以下 set-vault-access-policy 示例将权限策略附加到指定的文件库。

```
aws glacier set-vault-access-policy \
    --account-id 111122223333 \
    --vault-name example_vault
    --policy '{"Policy": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::444455556666:root
\"},\"Action\":\"glacier:ListJobs\",\"Resource\":\"arn:aws:glacier:us-
east-1:111122223333:vaults/example_vault\"},{\"Effect\":\"Allow\",\"Principal\":
{\"AWS\":\"arn:aws:iam::444455556666:root\"},\"Action\":\"glacier:UploadArchive\",
\"Resource\":\"arn:aws:glacier:us-east-1:111122223333:vaults/example_vault\"}]}"}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetVaultAccessPolicy。

set-vault-notifications

以下代码示例演示了如何使用 set-vault-notifications。

Amazon CLI

以下命令为名为 my-vault 的文件库配置 SNS 通知：

```
aws glacier set-vault-notifications --account-id - --vault-name my-vault --vault-
notification-config file://notificationconfig.json
```

notificationconfig.json 是当前文件夹中的一个 JSON 文件，用于指定 SNS 主题和要发布的事件：

```
{
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
  "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 SetVaultNotifications。

upload-archive

以下代码示例演示了如何使用 upload-archive。

Amazon CLI

以下命令将名为 archive.zip 的当前文件夹中的存档上传到名为 my-vault 的文件库：

```
aws glacier upload-archive --account-id - --vault-name my-vault --body archive.zip
```

输出：

```
{
    "archiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
    "checksum": "969fb39823836d81f0cc028195fcdbcbbe76cdde932d4646fa7de5f21e18aa67",
    "location": "/0123456789012/vaults/my-vault/archives/
kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw"
}
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

要检索上传的存档，可使用 aws glacier initiate-job 命令启动检索作业。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 UploadArchive。

upload-multipart-part

以下代码示例演示了如何使用 upload-multipart-part。

Amazon CLI

以下命令上传存档的前 1 MiB（1024 x 1024 字节）部分：

```
aws glacier upload-multipart-part --body part1 --range 'bytes
 0-1048575/*' --account-id - --vault-name my-vault --upload-
```

```
id 19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ
```

Amazon Glacier 在执行操作时需要一个账户 ID 参数，但您可以使用连字符来指定正在使用的账户。

正文参数采用本地文件系统上分段文件的路径。范围参数采用 HTTP 内容范围，指示该分段在已完成的存档中占用的字节。上传 ID 由 `aws glacier initiate-multipart-upload` 命令返回，也可以使用 `aws glacier list-multipart-uploads` 获取它。

有关使用 Amazon CLI 分段上传到 Amazon Glacier 的更多信息，请参阅《Amazon CLI 用户指南》中的“使用 Amazon Glacier”。

- 有关 API 详细信息，请参阅《Amazon CLI API Reference》中的 UploadMultipartPart。

使用 Amazon CLI 的 Global Accelerator 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Global Accelerator 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-custom-routing-endpoints`

以下代码示例演示了如何使用 `add-custom-routing-endpoints`。

Amazon CLI

将 VPC 子网端点添加到自定义路由加速器的端点组

以下 `add-custom-routing-endpoints` 示例将 VPC 子网端点添加到自定义路由加速器的端点组。

```
aws globalaccelerator add-custom-routing-endpoints \
    --endpoint-group-
arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd \
    --endpoint-configurations "EndpointId=subnet-1234567890abcdef0"
```

输出：

```
{
    "EndpointDescriptions": [
        {
            "EndpointId": "subnet-1234567890abcdef0"
        }
    ],

 "EndpointGroupArn":"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd"
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器的 VPC 子网端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddCustomRoutingEndpoints。

advertise-byoip-cidr

以下代码示例演示了如何使用 advertise-byoip-cidr。

Amazon CLI

公告地址范围

以下 advertise-byoip-cidr 示例请求 Amazon 公告您已预置用于 Amazon 资源的地址范围。

```
aws globalaccelerator advertise-byoip-cidr \
    --cidr 198.51.100.0/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "198.51.100.0/24",
```

```
        "State": "PENDING_ADVERTISING"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AdvertiseByoipCidr。

`allow-custom-routing-traffic`

以下代码示例演示了如何使用 `allow-custom-routing-traffic`。

Amazon CLI

允许流量传入自定义路由加速器的 VPC 子网中的特定 Amazon EC2 实例目标

以下 `allow-custom-routing-traffic` 示例指定允许流量传入特定的 Amazon EC2 实例（目标）IP 地址，自定义路由加速器中的 VPC 子网端点可以接收流量。

```
aws globalaccelerator allow-custom-routing-traffic \
    --endpoint-group-
arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab88888example \
    --endpoint-id subnet-abcd123example \
    --destination-addresses "172.31.200.6" "172.31.200.7" \
    --destination-ports 80 81
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器的 VPC 子网端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllowCustomRoutingTraffic。

`create-accelerator`

以下代码示例演示了如何使用 `create-accelerator`。

Amazon CLI

创建加速器

以下 create-accelerator 示例创建一个带有两个标签和两个 BYOIP 静态 IP 地址的加速器。必须指定用于创建或更新加速器的 US-West-2（Oregon）区域。

```
aws globalaccelerator create-accelerator \
    --name ExampleAccelerator \
    --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project" \
    --ip-addresses 192.0.2.250 198.51.100.52
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleAccelerator",
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1542394847.0,
        "LastModifiedTime": 1542394847.0
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccelerator。

create-custom-routing-accelerator

以下代码示例演示了如何使用 create-custom-routing-accelerator。

Amazon CLI

创建自定义路由加速器

以下 create-custom-routing-accelerator 示例使用标签 Name 和 Project 创建自定义路由加速器。

```
aws globalaccelerator create-custom-routing-accelerator \
    --name ExampleCustomRoutingAccelerator \
    --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project" \
    --ip-addresses 192.0.2.250 198.51.100.52
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleCustomRoutingAccelerator",
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1542394847.0,
        "LastModifiedTime": 1542394847.0
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomRoutingAccelerator。

create-custom-routing-endpoint-group

以下代码示例演示了如何使用 create-custom-routing-endpoint-group。

Amazon CLI

为自定义路由加速器创建端点组

以下 create-custom-routing-endpoint-group 示例为自定义路由加速器创建端点组。

```
aws globalaccelerator create-custom-routing-endpoint-group \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
    --endpoint-group-region us-east-2 \
    --destination-configurations "FromPort=80,ToPort=81,Protocols=TCP,UDP"
```

输出：

```
{
    "EndpointGroup": {
        "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd",
        "EndpointGroupRegion": "us-east-2",
        "DestinationDescriptions": [
            {
                "FromPort": 80,
                "ToPort": 81,
                "Protocols": [
                    "TCP",
                    "UDP"
                ]
            }
        ],
        "EndpointDescriptions": []
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomRoutingEndpointGroup。

create-custom-routing-listener

以下代码示例演示了如何使用 create-custom-routing-listener。

Amazon CLI

为自定义路由加速器创建侦听器

以下 create-custom-routing-listener 示例为自定义路由加速器创建端口范围为 5000 到 10000 的侦听器。

```
aws globalaccelerator create-custom-routing-listener \
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --port-ranges FromPort=5000,ToPort=10000
```

输出：

```
{
    "Listener": {
        "PortRange": [
            "FromPort": 5000,
            "ToPort": 10000
        ],
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中自定义路由加速器的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomRoutingListener。

create-endpoint-group

以下代码示例演示了如何使用 create-endpoint-group。

Amazon CLI

创建端点组

以下 create-endpoint-group 示例创建一个具有一个端点的端点组。

```
aws globalaccelerator create-endpoint-group \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
    --endpoint-group-region us-east-1 \
    --endpoint-configurations EndpointId=i-1234567890abcdef0,Weight=128
```

输出：

```
{
    "EndpointGroup": {
        "TrafficDialPercentage": 100.0,
        "EndpointDescriptions": [
            {
                "Weight": 128,
                "EndpointId": "i-1234567890abcdef0"
            }
        ],
        "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
        "EndpointGroupRegion": "us-east-1"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEndpointGroup。

create-listener

以下代码示例演示了如何使用 create-listener。

Amazon CLI

创建侦听器

以下 create-listener 示例创建一个具有两个端口的侦听器。

```
aws globalaccelerator create-listener \
```

```
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh \
    --port-ranges FromPort=80,ToPort=80 FromPort=81,ToPort=81 \
    --protocol TCP
```

输出：

```
{
    "Listener": {
        "PortRanges": [
            {
                "ToPort": 80,
                "FromPort": 80
            },
            {
                "ToPort": 81,
                "FromPort": 81
            }
        ],
        "ClientAffinity": "NONE",
        "Protocol": "TCP",
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateListener。

deny-custom-routing-traffic

以下代码示例演示了如何使用 deny-custom-routing-traffic。

Amazon CLI

指定无法在自定义路由加速器中接收流量的目标地址

以下 deny-custom-routing-traffic 示例指定无法接收自定义路由加速器流量的子网端点中的一个或多个目标地址。要指定多个目标地址，请使用空格分隔地址。没有成功的 deny-custom-routing-traffic 调用的响应。

```
aws globalaccelerator deny-custom-routing-traffic \
    --endpoint-group-
arn "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab88888example" \
    --endpoint-id "subnet-abcd123example" \
    --destination-addresses "198.51.100.52"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器的 VPC 子网端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DenyCustomRoutingTraffic。

deprovision-byoip-cidr

以下代码示例演示了如何使用 deprovision-byoip-cidr。

Amazon CLI

取消预置地址范围

以下 deprovision-byoip-cidr 示例释放您预置用于 Amazon 资源的指定地址范围。

```
aws globalaccelerator deprovision-byoip-cidr \
    --cidr "198.51.100.0/24"
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "198.51.100.0/24",
        "State": "PENDING_DEPROVISIONING"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprovisionByoipCidr。

describe-accelerator-attributes

以下代码示例演示了如何使用 describe-accelerator-attributes。

Amazon CLI

描述加速器的属性

以下 describe-accelerator-attributes 示例检索加速器的属性详细信息。

```
aws globalaccelerator describe-accelerator-attributes \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "AcceleratorAttributes": {
        "FlowLogsEnabled": true
        "FlowLogsS3Bucket": flowlogs-abc
        "FlowLogsS3Prefix": bucketprefix-abc
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAcceleratorAttributes。

describe-accelerator

以下代码示例演示了如何使用 describe-accelerator。

Amazon CLI

描述加速器

以下 describe-accelerator 示例检索有关指定加速器的详细信息。

```
aws globalaccelerator describe-accelerator \
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleAccelerator",
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1542394847,
        "LastModifiedTime": 1542395013
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccelerator。

describe-custom-routing-accelerator-attributes

以下代码示例演示了如何使用 describe-custom-routing-accelerator-attributes。

Amazon CLI

描述自定义路由加速器的属性

以下 describe-custom-routing-accelerator-attributes 示例描述自定义路由加速器的属性。

```
aws globalaccelerator describe-custom-routing-accelerator-attributes \
```

```
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "AcceleratorAttributes": {
    "FlowLogsEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomRoutingAcceleratorAttributes。

describe-custom-routing-accelerator

以下代码示例演示了如何使用 describe-custom-routing-accelerator。

Amazon CLI

描述自定义路由加速器

以下 describe-custom-routing-accelerator 示例检索有关指定自定义路由加速器的详细信息。

```
aws globalaccelerator describe-custom-routing-accelerator \
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleCustomRoutingAccelerator",
```

```
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1542394847,
        "LastModifiedTime": 1542395013
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomRoutingAccelerator。

describe-custom-routing-endpoint-group

以下代码示例演示了如何使用 describe-custom-routing-endpoint-group。

Amazon CLI

描述自定义路由加速器的端点组

以下 describe-custom-routing-endpoint-group 示例描述自定义路由加速器的端点组。

```
aws globalaccelerator describe-custom-routing-endpoint-group \
    --endpoint-group-
arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz/endpoint-group/ab88888example
```

输出：

```
{
    "EndpointGroup": {
```

```
        "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz/endpoint-group/ab88888example",
        "EndpointGroupRegion": "us-east-2",
        "DestinationDescriptions": [
            {
                "FromPort": 5000,
                "ToPort": 10000,
                "Protocols": [
                    "UDP"
                ]
            }
        ],
        "EndpointDescriptions": [
            {
                "EndpointId": "subnet-1234567890abcdef0"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomRoutingEndpointGroup。

describe-custom-routing-listener

以下代码示例演示了如何使用 describe-custom-routing-listener。

Amazon CLI

描述自定义路由加速器的侦听器

以下 describe-custom-routing-listener 示例描述自定义路由加速器的侦听器。

```
aws globalaccelerator describe-custom-routing-listener \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

输出：

```
{
    "Listener": {
        "PortRanges": [
            "FromPort": 5000,
            "ToPort": 10000
        ],
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中自定义路由加速器的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomRoutingListener。

describe-endpoint-group

以下代码示例演示了如何使用 describe-endpoint-group。

Amazon CLI

描述端点组

以下 describe-endpoint-group 示例检索有关具有以下端点的端点组的详细信息：Amazon EC2 实例、ALB 和 NLB。

```
aws globalaccelerator describe-endpoint-group \
    --endpoint-group-
arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-group/
ab88888example
```

输出：

```
{
    "EndpointGroup": {
        "TrafficDialPercentage": 100.0,
        "EndpointDescriptions": [
        {
```

```
            "Weight": 128,
            "EndpointId": "i-1234567890abcdef0"
        },
        {
            "Weight": 128,
            "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/app/ALBTesting/alb01234567890xyz"
        },
        {
            "Weight": 128,
            "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/net/NLBTesting/alb01234567890qrs"
        }
        ],
        "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-
group/4321abcd-abcd-4321-abcd-4321abcdefg",
        "EndpointGroupRegion": "us-east-1"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpointGroup。

describe-listener

以下代码示例演示了如何使用 describe-listener。

Amazon CLI

描述侦听器

以下 describe-listener 示例描述侦听器。

```
aws globalaccelerator describe-listener \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

输出：

```
{
    "Listener": {
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234",
        "PortRanges": [
            {
                "FromPort": 80,
                "ToPort": 80
            }
        ],
        "Protocol": "TCP",
        "ClientAffinity": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeListener。

list-accelerators

以下代码示例演示了如何使用 list-accelerators。

Amazon CLI

列出加速器

以下 list-accelerators 示例列出您的 Amazon 账户中的加速器。该账户拥有两个加速器。

```
aws globalaccelerator list-accelerators
```

输出：

```
{
    "Accelerators": [
        {
            "AcceleratorArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/5555abcd-abcd-5555-
abcd-5555EXAMPLE1",
            "Name": "TestAccelerator",
```

```
            "IpAddressType": "IPV4",
            "Enabled": true,
            "IpSets": [
                {
                    "IpFamily": "IPv4",
                    "IpAddresses": [
                        "192.0.2.250",
                        "198.51.100.52"
                    ]
                }
            ],
            "DnsName": "5a5a5a5a5a5a5a5a.awsglobalaccelerator.com",
            "Status": "DEPLOYED",
            "CreatedTime": 1552424416.0,
            "LastModifiedTime": 1569375641.0
        },
        {
            "AcceleratorArn":
 "arn:aws:globalaccelerator::888888888888:accelerator/8888abcd-abcd-8888-
abcd-8888EXAMPLE2",
            "Name": "ExampleAccelerator",
            "IpAddressType": "IPV4",
            "Enabled": true,
            "IpSets": [
                {
                    "IpFamily": "IPv4",
                    "IpAddresses": [
                        "192.0.2.100",
                        "198.51.100.10"
                    ]
                }
            ],
            "DnsName": "6a6a6a6a6a6a6a6a.awsglobalaccelerator.com",
            "Status": "DEPLOYED",
            "CreatedTime": 1575585564.0,
            "LastModifiedTime": 1579809243.0
        },
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccelerators。

`list-byoip-cidr`

以下代码示例演示了如何使用 `list-byoip-cidr`。

Amazon CLI

列出地址范围

以下 `list-byoip-cidr` 示例列出了您预置用于 Global Accelerator 的自带 IP 地址（BYOIP）地址范围。

```
aws globalaccelerator list-byoip-cidrs
```

输出：

```
{
    "ByoipCidrs": [
        {
            "Cidr": "198.51.100.0/24",
            "State": "READY"
        }
        {
            "Cidr": "203.0.113.25/24",
            "State": "READY"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListByoipCidr。

`list-custom-routing-accelerators`

以下代码示例演示了如何使用 `list-custom-routing-accelerators`。

Amazon CLI

列出自定义路由加速器

以下 `list-custom-routing-accelerators` 示例列出了 Amazon 账户中的自定义路由加速器。

```
aws globalaccelerator list-custom-routing-accelerators
```

输出：

```
{
    "Accelerators": [
        {
            "AcceleratorArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/5555abcd-abcd-5555-
abcd-5555EXAMPLE1",
            "Name": "TestCustomRoutingAccelerator",
            "IpAddressType": "IPV4",
            "Enabled": true,
            "IpSets": [
                {
                    "IpFamily": "IPv4",
                    "IpAddresses": [
                        "192.0.2.250",
                        "198.51.100.52"
                    ]
                }
            ],
            "DnsName": "5a5a5a5a5a5a5a5a.awsglobalaccelerator.com",
            "Status": "DEPLOYED",
            "CreatedTime": 1552424416.0,
            "LastModifiedTime": 1569375641.0
        },
        {
            "AcceleratorArn":
 "arn:aws:globalaccelerator::888888888888:accelerator/8888abcd-abcd-8888-
abcd-8888EXAMPLE2",
            "Name": "ExampleCustomRoutingAccelerator",
            "IpAddressType": "IPV4",
            "Enabled": true,
            "IpSets": [
                {
                    "IpFamily": "IPv4",
                    "IpAddresses": [
                        "192.0.2.100",
                        "198.51.100.10"
                    ]
                }
            ],
```

```
            "DnsName": "6a6a6a6a6a6a6a6a.awsglobalaccelerator.com",
            "Status": "DEPLOYED",
            "CreatedTime": 1575585564.0,
            "LastModifiedTime": 1579809243.0
        },
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomRoutingAccelerators。

`list-custom-routing-endpoint-groups`

以下代码示例演示了如何使用 `list-custom-routing-endpoint-groups`。

Amazon CLI

列出自定义路由加速器中侦听器的端点组

以下 `list-custom-routing-endpoint-groups` 示例列出了自定义路由加速器中侦听器的端点组。

```
aws globalaccelerator list-custom-routing-endpoint-groups \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

输出：

```
{
    "EndpointGroups": [
        {
            "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab88888example",
            "EndpointGroupRegion": "eu-central-1",
            "DestinationDescriptions": [
                {
                    "FromPort": 80,
                    "ToPort": 80,
                    "Protocols": [
```

```
                    "TCP",
                    "UDP"
                ]
            }
        ]
        "EndpointDescriptions": [
            {
                "EndpointId": "subnet-abcd123example"
            }
        ]
    }
]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomRoutingEndpointGroups。

`list-custom-routing-listeners`

以下代码示例演示了如何使用 `list-custom-routing-listeners`。

Amazon CLI

列出自定义路由加速器的侦听器

以下 `list-custom-routing-listeners` 示例列出了自定义路由加速器的侦听器。

```
aws globalaccelerator list-custom-routing-listeners \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234",
```

```
            "PortRanges": [
                {
                    "FromPort": 5000,
                    "ToPort": 10000
                }
            ],
            "Protocol": "TCP"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中自定义路由加速器的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomRoutingListeners。

`list-custom-routing-port-mappings-by-destination`

以下代码示例演示了如何使用 `list-custom-routing-port-mappings-by-destination`。

Amazon CLI

列出特定自定义路由加速器目标的端口映射

以下 `list-custom-routing-port-mappings-by-destination` 示例提供了自定义路由加速器的特定目标 EC2 服务器（位于目标地址）的端口映射。

```
aws globalaccelerator list-custom-routing-port-mappings-by-destination \
    --endpoint-id subnet-abcd123example \
    --destination-address 198.51.100.52
```

输出：

```
{
    "DestinationPortMappings": [
        {
            "AcceleratorArn":
 "arn:aws:globalaccelerator::402092451327:accelerator/24ea29b8-
d750-4489-8919-3095f3c4b0a7",
                "AcceleratorSocketAddresses": [
                    {
                        "IpAddress": "192.0.2.250",
```

```
                        "Port": 65514
                    },
                    {
                        "IpAddress": "192.10.100.99",
                        "Port": 65514
                    }
                ],
                "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab88888example",
                "EndpointId": "subnet-abcd123example",
                "EndpointGroupRegion": "us-west-2",
                "DestinationSocketAddress": {
                    "IpAddress": "198.51.100.52",
                    "Port": 80
                },
                "IpAddressType": "IPv4",
                "DestinationTrafficState": "ALLOW"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的自定义路由加速器在 Amazon Global Accelerator 中的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomRoutingPortMappingsByDestination。

list-custom-routing-port-mappings

以下代码示例演示了如何使用 list-custom-routing-port-mappings。

Amazon CLI

列出自定义路由加速器中的端口映射

以下 list-custom-routing-port-mappings 示例提供了自定义路由加速器中端口映射的部分列表。

```
aws globalaccelerator list-custom-routing-port-mappings \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "PortMappings": [
        {
            "AcceleratorPort": 40480,
            "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
            "EndpointId": "subnet-1234567890abcdef0",
            "DestinationSocketAddress": {
                "IpAddress": "192.0.2.250",
                "Port": 80
            },
            "Protocols": [
                "TCP",
                "UDP"
            ],
            "DestinationTrafficState": "ALLOW"
        }
        {
            "AcceleratorPort": 40481,
            "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
            "EndpointId": "subnet-1234567890abcdef0",
            "DestinationSocketAddress": {
               "IpAddress": "192.0.2.251",
              "Port": 80
            },
            "Protocols": [
                "TCP",
                "UDP"
            ],
            "DestinationTrafficState": "ALLOW"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的自定义路由加速器在 Amazon Global Accelerator 中的工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomRoutingPortMappings。

list-endpoint-groups

以下代码示例演示了如何使用 list-endpoint-groups。

Amazon CLI

列出端点组

以下 list-endpoint-groups 示例列出了侦听器的端点组。该侦听器拥有两个端点组。

```
aws globalaccelerator --region us-west-2 list-endpoint-groups \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

输出：

```
{
    "EndpointGroups": [
        {
            "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab88888example",
            "EndpointGroupRegion": "eu-central-1",
            "EndpointDescriptions": [],
            "TrafficDialPercentage": 100.0,
            "HealthCheckPort": 80,
            "HealthCheckProtocol": "TCP",
            "HealthCheckIntervalSeconds": 30,
            "ThresholdCount": 3
        }
        {
            "EndpointGroupArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab99999example",
            "EndpointGroupRegion": "us-east-1",
            "EndpointDescriptions": [],
            "TrafficDialPercentage": 50.0,
            "HealthCheckPort": 80,
            "HealthCheckProtocol": "TCP",
            "HealthCheckIntervalSeconds": 30,
            "ThresholdCount": 3
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEndpointGroups。

`list-listeners`

以下代码示例演示了如何使用 `list-listeners`。

Amazon CLI

列出侦听器

以下 `list-listeners` 示例列出了加速器的侦听器。

```
aws globalaccelerator list-listeners \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "Listeners": [
        {
            "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/abcdef1234",
            "PortRanges": [
                {
                    "FromPort": 80,
                    "ToPort": 80
                }
            ],
            "Protocol": "TCP",
            "ClientAffinity": "NONE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListListeners。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出加速器的标签

以下 `list-tags-for-resource` 示例列出了特定加速器的标签。

```
aws globalaccelerator list-tags-for-resource \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh
```

输出：

```
{
    "Tags": [
        {
            "Key": "Project",
            "Value": "A123456"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中进行标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`provision-byoip-cidr`

以下代码示例演示了如何使用 `provision-byoip-cidr`。

Amazon CLI

预置地址范围

以下 provision-byoip-cidr 示例预置指定的地址范围以用于您的 Amazon 资源。

```
aws globalaccelerator provision-byoip-cidr \
    --cidr 192.0.2.250/24 \
    --cidr-authorization-context Message="$text_message",Signature="$signed_message"
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "192.0.2.250/24",
        "State": "PENDING_PROVISIONING"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ProvisionByoipCidr。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记加速器

以下 tag-resource 示例向加速器添加 Name（名称）和 Project（项目）标签，以及每个标签的相应值。

```
aws globalaccelerator tag-resource \
    --resource-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中进行标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从加速器中删除标签

以下 untag-resource 示例从加速器中删除 Name（名称）和 Project（项目）标签。

```
aws globalaccelerator untag-resource \
    --resource-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --tag-keys Key="Name" Key="Project"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中进行标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-accelerator-attributes

以下代码示例演示了如何使用 update-accelerator-attributes。

Amazon CLI

更新加速器的属性

以下 update-accelerator-attributes 示例更新加速器以启用流日志。必须指定用于创建或更新加速器属性的 US-West-2（Oregon）区域。

```
aws globalaccelerator update-accelerator-attributes \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --flow-logs-enabled \
    --flow-logs-s3-bucket flowlogs-abc \
    --flow-logs-s3-prefix bucketprefix-abc
```

输出：

```
{
```

```
    "AcceleratorAttributes": {
        "FlowLogsEnabled": true
        "FlowLogsS3Bucket": flowlogs-abc
        "FlowLogsS3Prefix": bucketprefix-abc
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAcceleratorAttributes。

update-accelerator

以下代码示例演示了如何使用 update-accelerator。

Amazon CLI

更新加速器

以下 update-accelerator 示例修改加速器以将加速器名称更改为 ExampleAcceleratorNew。必须指定用于创建或更新加速器的 US-West-2 (Oregon) 区域。

```
aws globalaccelerator update-accelerator \
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --name ExampleAcceleratorNew
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleAcceleratorNew",
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
```

```
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1232394847,
        "LastModifiedTime": 1232395654
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccelerator。

update-custom-routing-accelerator-attributes

以下代码示例演示了如何使用 update-custom-routing-accelerator-attributes。

Amazon CLI

更新自定义路由加速器的属性

以下 update-custom-routing-accelerator-attributes 示例更新自定义路由加速器以启用流日志。

```
aws globalaccelerator update-custom-routing-accelerator-attributes \
    --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --flow-logs-enabled \
    --flow-logs-s3-bucket flowlogs-abc \
    --flow-logs-s3-prefix bucketprefix-abc
```

输出：

```
{
    "AcceleratorAttributes": {
        "FlowLogsEnabled": true
        "FlowLogsS3Bucket": flowlogs-abc
        "FlowLogsS3Prefix": bucketprefix-abc
```

```
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCustomRoutingAcceleratorAttributes。

update-custom-routing-accelerator

以下代码示例演示了如何使用 update-custom-routing-accelerator。

Amazon CLI

更新自定义路由加速器

以下 update-custom-routing-accelerator 示例修改自定义路由加速器以更改加速器名称。

```
aws globalaccelerator --region us-west-2 update-custom-routing-accelerator \
    --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh \
    --name ExampleCustomRoutingAcceleratorNew
```

输出：

```
{
    "Accelerator": {
        "AcceleratorArn":
 "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh",
        "IpAddressType": "IPV4",
        "Name": "ExampleCustomRoutingAcceleratorNew",
        "Enabled": true,
        "Status": "IN_PROGRESS",
        "IpSets": [
            {
                "IpAddresses": [
                    "192.0.2.250",
                    "198.51.100.52"
                ],
                "IpFamily": "IPv4"
```

```
            }
        ],
        "DnsName":"a1234567890abcdef.awsglobalaccelerator.com",
        "CreatedTime": 1232394847,
        "LastModifiedTime": 1232395654
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的自定义路由加速器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCustomRoutingAccelerator。

update-custom-routing-listener

以下代码示例演示了如何使用 update-custom-routing-listener。

Amazon CLI

更新自定义路由加速器的侦听器

以下 update-custom-routing-listener 示例更新侦听器以更改端口范围。

```
aws globalaccelerator update-custom-routing-listener \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
    --port-ranges FromPort=10000,ToPort=20000
```

输出：

```
{
    "Listener": {
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-abcd-1234abcdefgh/listener/0123vxyz
        "PortRanges": [
            {
                "FromPort": 10000,
                "ToPort": 20000
            }
        ],
        "Protocol": "TCP"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中自定义路由加速器的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCustomRoutingListener。

update-endpoint-group

以下代码示例演示了如何使用 update-endpoint-group。

Amazon CLI

更新端点组

以下 update-endpoint-group 示例向端点组添加了三个端点：弹性 IP 地址、ALB 和 NLB。

```
aws globalaccelerator update-endpoint-group \
    --endpoint-group-
arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-group/
ab88888example \
    --endpoint-configurations \
        EndpointId=eipalloc-eip01234567890abc,Weight=128 \
        EndpointId=arn:aws:elasticloadbalancing:us-east-1:000123456789:loadbalancer/
app/ALBTesting/alb01234567890xyz,Weight=128 \
        EndpointId=arn:aws:elasticloadbalancing:us-east-1:000123456789:loadbalancer/
net/NLBTesting/alb01234567890qrs,Weight=128
```

输出：

```
{
    "EndpointGroup": {
        "TrafficDialPercentage": 100,
        "EndpointDescriptions": [
            {
                "Weight": 128,
                "EndpointId": "eip01234567890abc"
            },
            {
                "Weight": 128,
                "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/app/ALBTesting/alb01234567890xyz"
```

```
            },
            {
                "Weight": 128,
                "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/net/NLBTesting/alb01234567890qrs"
            }
        ],
        "EndpointGroupArn":
 "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-
group/4321abcd-abcd-4321-abcd-4321abcdefg",
        "EndpointGroupRegion": "us-east-1"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的端点组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEndpointGroup。

update-listener

以下代码示例演示了如何使用 update-listener。

Amazon CLI

更新侦听器

以下 update-listener 示例更新侦听器以将端口更改为 100。

```
aws globalaccelerator update-listener \
    --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
    --port-ranges FromPort=100,ToPort=100
```

输出：

```
{
    "Listener": {
        "ListenerArn":
 "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz
```

```
        "PortRanges": [
            {
                "FromPort": 100,
                "ToPort": 100
            }
        ],
        "Protocol": "TCP",
        "ClientAffinity": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的 Amazon Global Accelerator 中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateListener。

`withdraw-byoip-cidr`

以下代码示例演示了如何使用 `withdraw-byoip-cidr`。

Amazon CLI

撤回地址范围

以下 `withdraw-byoip-cidr` 示例从 Amazon Global Accelerator 中撤回您之前公布用于 Amazon 资源的地址范围。

```
aws globalaccelerator withdraw-byoip-cidr \
    --cidr 192.0.2.250/24
```

输出：

```
{
    "ByoipCidr": {
        "Cidr": "192.0.2.250/24",
        "State": "PENDING_WITHDRAWING"
    }
}
```

有关更多信息，请参阅《Amazon Global Accelerator 开发人员指南》中的在 Amazon Global Accelerator 中自带 IP 地址。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 WithdrawByoipCidr。

Amazon Glue使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Glue 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-stop-job-run`

以下代码示例演示了如何使用 batch-stop-job-run。

Amazon CLI

停止作业运行

以下 batch-stop-job-run 示例停止作业运行。

```
aws glue batch-stop-job-run \
    --job-name "my-testing-job" \
    --job-run-id jr_852f1de1f29fb62e0ba4166c33970803935d87f14f96cfdee5089d5274a61d3f
```

输出：

```
{
    "SuccessfulSubmissions": [
        {
            "JobName": "my-testing-job",
            "JobRunId":
 "jr_852f1de1f29fb62e0ba4166c33970803935d87f14f96cfdee5089d5274a61d3f"
```

```
        }
    ],
    "Errors": [],
    "ResponseMetadata": {
        "RequestId": "66bd6b90-01db-44ab-95b9-6aeff0e73d88",
        "HTTPStatusCode": 200,
        "HTTPHeaders": {
            "date": "Fri, 16 Oct 2020 20:54:51 GMT",
            "content-type": "application/x-amz-json-1.1",
            "content-length": "148",
            "connection": "keep-alive",
            "x-amzn-requestid": "66bd6b90-01db-44ab-95b9-6aeff0e73d88"
        },
        "RetryAttempts": 0
    }
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的作业运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchStopJobRun。

create-connection

以下代码示例演示了如何使用 create-connection。

Amazon CLI

为 Amazon Glue 数据存储创建连接

以下 create-connection 示例在 Amazon Glue 数据目录中创建一个连接，该连接为 Kafka 数据存储提供连接信息。

```
aws glue create-connection \
    --connection-input '{ \
        "Name":"conn-kafka-custom", \
        "Description":"kafka connection with ssl to custom kafka", \
        "ConnectionType":"KAFKA",  \
        "ConnectionProperties":{  \
            "KAFKA_BOOTSTRAP_SERVERS":"<Kafka-broker-server-url>:<SSL-Port>", \
            "KAFKA_SSL_ENABLED":"true", \
            "KAFKA_CUSTOM_CERT": "s3://bucket/prefix/cert-file.pem" \
        }, \
        "PhysicalConnectionRequirements":{ \
```

```
            "SubnetId":"subnet-1234", \
            "SecurityGroupIdList":["sg-1234"], \
            "AvailabilityZone":"us-east-1a"} \
    }' \
    --region us-east-1
    --endpoint https://glue.us-east-1.amazonaws.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在 Amazon Glue 数据目录中定义连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnection。

create-database

以下代码示例演示了如何使用 create-database。

Amazon CLI

创建数据库

以下 create-database 示例在 Amazon Glue 数据目录中创建一个数据库。

```
aws glue create-database \
    --database-input "{\"Name\":\"tempdb\"}" \
    --profile my_profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在数据目录中定义数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDatabase。

create-job

以下代码示例演示了如何使用 create-job。

Amazon CLI

创建用于转换数据的作业

以下 create-job 示例创建了一个运行存储在 S3 中的脚本的流式处理作业。

```
aws glue create-job \
    --name my-testing-job \
    --role AWSGlueServiceRoleDefault \
    --command '{ \
        "Name": "gluestreaming", \
        "ScriptLocation": "s3://amzn-s3-demo-bucket/folder/" \
    }' \
    --region us-east-1 \
    --output json \
    --default-arguments '{ \
        "--job-language":"scala", \
        "--class":"GlueApp" \
    }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

的内容test_script.scala：

```
import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
    def main(sysArgs: Array[String]) {
        val spark: SparkContext = new SparkContext()
        val glueContext: GlueContext = new GlueContext(spark)
        // @params: [JOB_NAME]
        val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME").toArray)
        Job.init(args("JOB_NAME"), glueContext, args.asJava)
        // @type: DataSource
        // @args: [database = "tempdb", table_name = "s3-source", transformation_ctx
 = "datasource0"]
        // @return: datasource0
```

```
        // @inputs: []
        val datasource0 = glueContext.getCatalogSource(database = "tempdb",
 tableName = "s3-source", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
        // @type: ApplyMapping
        // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
 ("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
 "status", "string")], transformation_ctx = "applymapping1"]
        // @return: applymapping1
        // @inputs: [frame = datasource0]
        val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
 "int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
 "int"), ("status", "string", "status", "string")), caseSensitive = false,
 transformationContext = "applymapping1")
        // @type: SelectFields
        // @args: [paths = ["sensorid", "currenttemperature", "status"],
 transformation_ctx = "selectfields2"]
        // @return: selectfields2
        // @inputs: [frame = applymapping1]
        val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
 "currenttemperature", "status"), transformationContext = "selectfields2")
        // @type: ResolveChoice
        // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name = "my-
s3-sink", transformation_ctx = "resolvechoice3"]
        // @return: resolvechoice3
        // @inputs: [frame = selectfields2]
        val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
 Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
 Some("my-s3-sink"), transformationContext = "resolvechoice3")
        // @type: DataSink
        // @args: [database = "tempdb", table_name = "my-s3-sink",
 transformation_ctx = "datasink4"]
        // @return: datasink4
        // @inputs: [frame = resolvechoice3]
        val datasink4 = glueContext.getCatalogSink(database = "tempdb",
 tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
 "datasink4").writeDynamicFrame(resolvechoice3)
        Job.commit()
    }
}
```

输出：

```
{
    "Name": "my-testing-job"
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在 Amazon Glue 中编写作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJob。

create-table

以下代码示例演示了如何使用 create-table。

Amazon CLI

示例 1：为 Kinesis 数据流创建表

以下 create-table 示例在 Amazon Glue 数据目录中创建一个描述 Kinesis 数据流的表。

```
aws glue create-table \
    --database-name tempdb \
    --table-input  '{"Name":"test-kinesis-input", "StorageDescriptor":{ \
            "Columns":[ \
                {"Name":"sensorid", "Type":"int"}, \
                {"Name":"currenttemperature", "Type":"int"}, \
                {"Name":"status", "Type":"string"} \
            ], \
            "Location":"my-testing-stream", \
            "Parameters":{ \
                "typeOfData":"kinesis","streamName":"my-testing-stream", \
                "kinesisUrl":"https://kinesis.us-east-1.amazonaws.com" \
            }, \
            "SerdeInfo":{ \
                "SerializationLibrary":"org.openx.data.jsonserde.JsonSerDe"} \
        }, \
        "Parameters":{ \
            "classification":"json"} \
        }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 开发人员指南》中的在 Amazon Glue 数据目录中定义表。

示例 2：为 Kafka 数据存储创建表

以下 `create-table` 示例在 Amazon Glue 数据目录中创建一个描述 Kafka 数据存储的表。

```
aws glue create-table \
    --database-name tempdb \
    --table-input  '{"Name":"test-kafka-input", "StorageDescriptor":{ \
            "Columns":[ \
                {"Name":"sensorid", "Type":"int"}, \
                {"Name":"currenttemperature", "Type":"int"}, \
                {"Name":"status", "Type":"string"} \
            ], \
            "Location":"glue-topic", \
            "Parameters":{ \
                "typeOfData":"kafka","topicName":"glue-topic", \
                "connectionName":"my-kafka-connection" \
            }, \
            "SerdeInfo":{ \
                "SerializationLibrary":"org.apache.hadoop.hive.serde2.OpenCSVSerde"}
 \
        }, \
        "Parameters":{ \
            "separatorChar":","} \
        }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 开发人员指南》中的在 Amazon Glue 数据目录中定义表。

示例 3：为 Amazon S3 数据存储创建表

以下 `create-table` 示例在 Amazon Glue 数据目录中创建一个表，用于描述 Amazon Simple Storage Service（Amazon S3）数据存储。

```
aws glue create-table \
    --database-name tempdb \
    --table-input  '{"Name":"s3-output", "StorageDescriptor":{ \
            "Columns":[ \
                {"Name":"s1", "Type":"string"}, \
                {"Name":"s2", "Type":"int"}, \
                {"Name":"s3", "Type":"string"}
```

```
            ], \
            "Location":"s3://bucket-path/", \
            "SerdeInfo":{ \
                "SerializationLibrary":"org.openx.data.jsonserde.JsonSerDe"} \
        }, \
        "Parameters":{ \
            "classification":"json"} \
        }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 开发人员指南》中的在 Amazon Glue 数据目录中定义表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTable。

delete-job

以下代码示例演示了如何使用 delete-job。

Amazon CLI

删除作业

以下 delete-job 示例删除了不再需要的作业。

```
aws glue delete-job \
    --job-name my-testing-job
```

输出：

```
{
    "JobName": "my-testing-job"
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在 Amazon Glue 控制台上处理作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteJob。

get-databases

以下代码示例演示了如何使用 get-databases。

Amazon CLI

在 Amazon Glue 数据目录中列出部分或全部数据库的定义

以下 get-databases 示例返回有关数据目录中数据库的信息。

```
aws glue get-databases
```

输出：

```
{
    "DatabaseList": [
        {
            "Name": "default",
            "Description": "Default Hive database",
            "LocationUri": "file:/spark-warehouse",
            "CreateTime": 1602084052.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "flights-db",
            "CreateTime": 1587072847.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
```

```
        },
        {
            "Name": "legislators",
            "CreateTime": 1601415625.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "tempdb",
            "CreateTime": 1601498566.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在数据目录中定义数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDatabases。

get-job-run

以下代码示例演示了如何使用 get-job-run。

Amazon CLI

获取有关作业运行的信息

以下 get-job-run 示例检索有关作业运行的信息。

```
aws glue get-job-run \
    --job-name "Combine legistators data" \
    --run-id jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e
```

输出：

```
{
    "JobRun": {
        "Id": "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
        "Attempt": 0,
        "JobName": "Combine legistators data",
        "StartedOn": 1602873931.255,
        "LastModifiedOn": 1602874075.985,
        "CompletedOn": 1602874075.985,
        "JobRunState": "SUCCEEDED",
        "Arguments": {
            "--enable-continuous-cloudwatch-log": "true",
            "--enable-metrics": "",
            "--enable-spark-ui": "true",
            "--job-bookmark-option": "job-bookmark-enable",
            "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-east-1/
sparkHistoryLogs/"
        },
        "PredecessorRuns": [],
        "AllocatedCapacity": 10,
        "ExecutionTime": 117,
        "Timeout": 2880,
        "MaxCapacity": 10.0,
        "WorkerType": "G.1X",
        "NumberOfWorkers": 10,
        "LogGroupName": "/aws-glue/jobs",
        "GlueVersion": "2.0"
    }
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的任务运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobRun。

get-job-runs

以下代码示例演示了如何使用 get-job-runs。

Amazon CLI

获取有关作业的所有作业运行的信息

以下 get-job-runs 示例检索有关作业的作业运行的信息。

```
aws glue get-job-runs \
    --job-name "my-testing-job"
```

输出：

```
{
    "JobRuns": [
        {
            "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
            "Attempt": 0,
            "JobName": "my-testing-job",
            "StartedOn": 1602873931.255,
            "LastModifiedOn": 1602874075.985,
            "CompletedOn": 1602874075.985,
            "JobRunState": "SUCCEEDED",
            "Arguments": {
                "--enable-continuous-cloudwatch-log": "true",
                "--enable-metrics": "",
                "--enable-spark-ui": "true",
                "--job-bookmark-option": "job-bookmark-enable",
                "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
            },
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 117,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
```

```
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_2",
            "Attempt": 2,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "JobName": "my-testing-job",
            "StartedOn": 1602811168.496,
            "LastModifiedOn": 1602811282.39,
            "CompletedOn": 1602811282.39,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 021AAB703DB20A2D;
                S3 Extended Request ID: teZk24Y09TkXzBvMPG502L5VJBhe9DJuWA9/
TXtuGOqfByajkfL/Tlqt5JBGdEGpigAqzdMDM/U=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 110,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "Attempt": 1,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f",
            "JobName": "my-testing-job",
            "StartedOn": 1602811020.518,
            "LastModifiedOn": 1602811138.364,
            "CompletedOn": 1602811138.364,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                 Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                 Request ID: 2671D37856AE7ABB;
```

```
                    S3 Extended Request ID: RLJCJw20brV
+PpC6GpORahyF2fp9flB5SSb2bTGPnUSPVizLXRl1PN3QZldb+v1o9qRVktNYbW8=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 113,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的任务运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobRuns。

get-job

以下代码示例演示了如何使用 get-job。

Amazon CLI

检索有关作业的信息

以下 get-job 示例检索有关作业的信息。

```
aws glue get-job \
    --job-name my-testing-job
```

输出：

```
{
    "Job": {
        "Name": "my-testing-job",
        "Role": "Glue_DefaultRole",
        "CreatedOn": 1602805698.167,
        "LastModifiedOn": 1602805698.167,
        "ExecutionProperty": {
            "MaxConcurrentRuns": 1
        },
```

```
        "Command": {
            "Name": "gluestreaming",
            "ScriptLocation": "s3://janetst-bucket-01/Scripts/test_script.scala",
            "PythonVersion": "2"
        },
        "DefaultArguments": {
            "--class": "GlueApp",
            "--job-language": "scala"
        },
        "MaxRetries": 0,
        "AllocatedCapacity": 10,
        "MaxCapacity": 10.0,
        "GlueVersion": "1.0"
    }
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJob。

get-plan

以下代码示例演示了如何使用 get-plan。

Amazon CLI

获取生成的代码，用于将数据从源表映射到目标表

以下 get-plan 检索生成的代码，用于将列从数据源映射到数据目标。

```
aws glue get-plan --mapping '[ \
    { \
        "SourcePath":"sensorid", \
        "SourceTable":"anything", \
        "SourceType":"int", \
        "TargetPath":"sensorid", \
        "TargetTable":"anything", \
        "TargetType":"int" \
    }, \
    { \
        "SourcePath":"currenttemperature", \
        "SourceTable":"anything", \
        "SourceType":"int", \
```

```
        "TargetPath":"currenttemperature", \
        "TargetTable":"anything", \
        "TargetType":"int" \
    }, \
    { \
        "SourcePath":"status", \
        "SourceTable":"anything", \
        "SourceType":"string", \
        "TargetPath":"status", \
        "TargetTable":"anything", \
        "TargetType":"string" \
    }]' \
    --source '{ \
        "DatabaseName":"tempdb", \
        "TableName":"s3-source" \
    }' \
    --sinks '[ \
        { \
            "DatabaseName":"tempdb", \
            "TableName":"my-s3-sink" \
        }]' 
    --language "scala"
    --endpoint https://glue.us-east-1.amazonaws.com
    --output "text"
```

输出：

```
import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
```

```
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    // @type: DataSource
    // @args: [database = "tempdb", table_name = "s3-source", transformation_ctx =
 "datasource0"]
    // @return: datasource0
    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "tempdb",
 tableName = "s3-source", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
    // @type: ApplyMapping
    // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
 ("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
 "status", "string")], transformation_ctx = "applymapping1"]
    // @return: applymapping1
    // @inputs: [frame = datasource0]
    val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
 "int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
 "int"), ("status", "string", "status", "string")), caseSensitive = false,
 transformationContext = "applymapping1")
    // @type: SelectFields
    // @args: [paths = ["sensorid", "currenttemperature", "status"],
 transformation_ctx = "selectfields2"]
    // @return: selectfields2
    // @inputs: [frame = applymapping1]
    val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
 "currenttemperature", "status"), transformationContext = "selectfields2")
    // @type: ResolveChoice
    // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name = "my-s3-
sink", transformation_ctx = "resolvechoice3"]
    // @return: resolvechoice3
    // @inputs: [frame = selectfields2]
    val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
 Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
 Some("my-s3-sink"), transformationContext = "resolvechoice3")
    // @type: DataSink
    // @args: [database = "tempdb", table_name = "my-s3-sink", transformation_ctx =
 "datasink4"]
    // @return: datasink4
    // @inputs: [frame = resolvechoice3]
    val datasink4 = glueContext.getCatalogSink(database = "tempdb",
 tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
 "datasink4").writeDynamicFrame(resolvechoice3)
    Job.commit()
```

```
  }
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的在 Amazon Glue 中编辑脚本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPlan。

get-tables

以下代码示例演示了如何使用 get-tables。

Amazon CLI

列出指定数据库中的部分或全部表的定义

以下 get-tables 示例返回有关指定数据库中表的信息。

```
aws glue get-tables --database-name 'tempdb'
```

输出：

```
{
    "TableList": [
        {
            "Name": "my-s3-sink",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730539.0,
            "UpdateTime": 1602730539.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
```

```
                ],
                "Location": "s3://janetst-bucket-01/test-s3-output/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "s3-source",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730658.0,
            "UpdateTime": 1602730658.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
```

```
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "test-kinesis-input",
            "DatabaseName": "tempdb",
            "CreateTime": 1601507001.0,
            "UpdateTime": 1601507001.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "my-testing-stream",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "Parameters": {
                    "kinesisUrl": "https://kinesis.us-east-1.amazonaws.com",
                    "streamName": "my-testing-stream",
                    "typeOfData": "kinesis"
                },
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
```

```
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 开发人员指南》中的在 Amazon Glue 数据目录中定义表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTables。

`start-crawler`

以下代码示例演示了如何使用 `start-crawler`。

Amazon CLI

启动爬网程序

以下 `start-crawler` 示例启动了一个爬网程序。

```
aws glue start-crawler --name my-crawler
```

输出：

```
None
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的定义爬网程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartCrawler。

`start-job-run`

以下代码示例演示了如何使用 `start-job-run`。

Amazon CLI

开始运行作业

以下 `start-job-run` 示例启动了一个作业。

```
aws glue start-job-run \
    --job-name my-job
```

输出：

```
{
    "JobRunId":
 "jr_22208b1f44eb5376a60569d4b21dd20fcb8621e1a366b4e7b2494af764b82ded"
}
```

有关更多信息，请参阅《Amazon Glue 开发人员指南》中的编写作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartJobRun。

使用 Amazon CLI 的 GuardDuty 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 GuardDuty 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-invitation`

以下代码示例演示了如何使用 `accept-invitation`。

Amazon CLI

接受成为当前区域中 GuardDuty 成员账户的邀请

以下 `accept-invitation` 示例演示如何接受成为当前区域中 GuardDuty 成员账户的邀请。

```
aws guardduty accept-invitation  \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --master-id 123456789111 \
    --invitation-id d6b94fb03a66ff665f7db8764example
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty 用户指南》中的通过邀请管理 GuardDuty 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptInvitation。

archive-findings

以下代码示例演示了如何使用 archive-findings。

Amazon CLI

将调查结果存档到当前区域中

此 archive-findings 示例演示如何将调查发现存档到当前区域中。

```
aws guardduty archive-findings \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --finding-ids d6b94fb03a66ff665f7db8764example 3eb970e0de00c16ec14e6910fexample
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty User Guide》中的 Creating suppression rules。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ArchiveFindings。

create-detector

以下代码示例演示了如何使用 create-detector。

Amazon CLI

在当前区域中启用 GuardDuty

此示例演示如何在当前区域中创建一个启用 GuardDuty 的新检测器。

```
aws guardduty create-detector \
    --enable
```

输出：

```
{
    "DetectorId": "b6b992d6d2f48e64bc59180bfexample"
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的启用 Amazon GuardDuty。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDetector。

create-filter

以下代码示例演示了如何使用 create-filter。

Amazon CLI

示例 1：在当前区域中创建新筛选条件

以下 create-filter 示例创建一个筛选条件，该筛选条件与从特定映像中创建的实例的所有 Portscan 调查发现相匹配。这不会隐藏这些调查发现。

```
aws guardduty create-filter \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --name myFilterExample \
    --finding-criteria '{"Criterion": {"type": {"Eq": ["Recon:EC2/
Portscan"]},"resource.instanceDetails.imageId": {"Eq": ["ami-0a7a207083example"]}}}'
```

输出：

```
{
    "Name": "myFilterExample"
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Filtering GuardDuty findings。

示例 2：在当前区域中创建新筛选条件并隐藏调查发现

以下 create-filter 示例创建一个筛选条件，该筛选条件与从特定映像中创建的实例的所有 Portscan 调查发现相匹配。此筛选条件归档这些调查发现，这样它们就不会出现在当前的调查发现中。

```
aws guardduty create-filter \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --action ARCHIVE \
    --name myFilterSecondExample \
    --finding-criteria '{"Criterion": {"type": {"Eq": ["Recon:EC2/
Portscan"]},"resource.instanceDetails.imageId": {"Eq": ["ami-0a7a207083example"]}}}'
```

输出：

```
{
    "Name": "myFilterSecondExample"
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Filtering GuardDuty findings。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFilter。

create-ip-set

以下代码示例演示了如何使用 create-ip-set。

Amazon CLI

创建并激活可信 IP 集

以下 create-ip-set 示例在当前区域中创建并激活可信 IP 集。

```
aws guardduty create-ip-set \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --name new-ip-set-example \
    --format TXT \
    --location s3://amzn-s3-demo-bucket/customtrustlist.csv \
    --activate
```

输出：

```
{
    "IpSetId": "d4b94fc952d6912b8f3060768example"
```

```
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Working with Trusted IP Lists and Threat Lists。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpSet。

`create-members`

以下代码示例演示了如何使用 `create-members`。

Amazon CLI

将新成员与您在当前区域中的 GuardDuty 主账户相关联

此示例演示如何关联成员账户，使其由当前账户作为 GuardDuty 主账户进行托管。

```
aws guardduty create-members
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --account-details AccountId=111122223333,Email=first
+member@example.com AccountId=111111111111 ,Email=another+member@example.com
```

输出：

```
{
   "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的管理多个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMembers。

`create-publishing-destination`

以下代码示例演示了如何使用 `create-publishing-destination`。

Amazon CLI

创建发布目标以将当前区域中的 GuardDuty 调查结果导出到其中

以下 `create-publishing-destination` 示例演示如何设置发布目标以导出当前（未归档）的 GuardDuty 调查发现，从而跟踪历史调查发现数据。

```
aws guardduty create-publishing-destination \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --destination-type S3 \
    --destination-properties 'DestinationArn=arn:aws:s3:::amzn-s3-demo-
bucket,KmsKeyArn=arn:aws:kms:us-west-1:111122223333:key/84cee9c5-dea1-401a-ab6d-
e1de7example'
```

输出：

```
{
    "DestinationId": "46b99823849e1bbc242dfbe3cexample"
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Exporting generated GuardDuty findings to Amazon S3 buckets。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePublishingDestination。

create-sample-findings

以下代码示例演示了如何使用 create-sample-findings。

Amazon CLI

在当前区域中创建示例 GuardDuty 调查结果

此示例演示如何创建所提供类型的示例调查结果。

```
aws guardduty create-sample-findings \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --finding-types UnauthorizedAccess:EC2/TorClient UnauthorizedAccess:EC2/TorRelay
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty 用户指南》中的示例调查结果。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSampleFindings。

create-threat-intel-set

以下代码示例演示了如何使用 create-threat-intel-set。

Amazon CLI

创建并激活新的威胁情报集

以下 create-threat-intel-set 示例在当前区域中创建并激活威胁情报集。

```
aws guardduty create-threat-intel-set \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --name myThreatSet-example \
    --format TXT \
    --location s3://amzn-s3-demo-bucket/threatlist.csv \
    --activate
```

输出：

```
{
    "ThreatIntelSetId": "20b9a4691aeb33506b808878cexample"
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Working with Trusted IP Lists and Threat Lists。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateThreatIntelSet。

decline-invitations

以下代码示例演示了如何使用 decline-invitations。

Amazon CLI

拒绝由当前区域中的另一账户托管 GuardDuty 的邀请

此示例演示如何拒绝成员资格邀请。

```
aws guardduty decline-invitations \
    --account-ids 111122223333
```

输出：

```
{
    "UnprocessedAccounts": []
```

```
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的通过邀请管理 GuardDuty 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeclineInvitations。

delete-detector

以下代码示例演示了如何使用 delete-detector。

Amazon CLI

删除当前区域中的检测器并禁用 GuardDuty

此示例演示如何删除检测器，如果成功，则将在与该检测器关联的区域中禁用 GuardDuty。

```
aws guardduty delete-detector \
    --detector-id b6b992d6d2f48e64bc59180bfexample
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty 用户指南》中的暂停或禁用 GuardDuty。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDetector。

delete-filter

以下代码示例演示了如何使用 delete-filter。

Amazon CLI

删除当前区域中的现有筛选条件

此示例演示如何删除筛选条件。

```
aws guardduty delete-filter \
    --detector-id b6b992d6d2f48e64bc59180bfexample \
    --filter-name byebyeFilter
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty 用户指南》中的筛选调查结果。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFilter。

`disable-organization-admin-account`

以下代码示例演示了如何使用 `disable-organization-admin-account`。

Amazon CLI

删除组织内作为 GuardDuty 委派管理员的账户

此示例演示如何删除作为 GuardDuty 委派管理员的账户。

```
aws guardduty disable-organization-admin-account \
    --admin-account-id 111122223333
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty 用户指南》中的使用 Amazon Organizations 管理账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableOrganizationAdminAccount。

`disassociate-from-master-account`

以下代码示例演示了如何使用 `disassociate-from-master-account`。

Amazon CLI

取消与当前区域中的当前管理员账户的关联

以下 `disassociate-from-master-account` 示例取消您的账户与当前 Amazon 区域中的当前 GuardDuty 管理员账户的关联。

```
aws guardduty disassociate-from-master-account \
    --detector-id d4b040365221be2b54a6264dcexample
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty User Guide》中的 Understanding the relationship between GuardDuty administrator account and member accounts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateFromMasterAccount。

get-detector

以下代码示例演示了如何使用 get-detector。

Amazon CLI

检索特定检测器的详细信息

以下 get-detector 示例显示指定检测器的配置详细信息。

```
aws guardduty get-detector \
    --detector-id 12abc34d567e8fa901bc2d34eexample
```

输出：

```
{
    "Status": "ENABLED",
    "ServiceRole": "arn:aws:iam::111122223333:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",
    "Tags": {},
    "FindingPublishingFrequency": "SIX_HOURS",
    "UpdatedAt": "2018-11-07T03:24:22.938Z",
    "CreatedAt": "2017-12-22T22:51:31.940Z"
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的概念和术语。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDetector。

get-findings

以下代码示例演示了如何使用 get-findings。

Amazon CLI

示例 1：检索特定调查结果的详细信息

以下 get-findings 示例检索指定调查结果的完整 JSON 调查结果详细信息。

```
aws guardduty get-findings \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --finding-id 1ab92989eaf0e742df4a014d5example
```

输出：

```
{
    "Findings": [
        {
            "Resource": {
                "ResourceType": "AccessKey",
                "AccessKeyDetails": {
                    "UserName": "testuser",
                    "UserType": "IAMUser",
                    "PrincipalId": "AIDACKCEVSQ6C2EXAMPLE",
                    "AccessKeyId": "ASIASZ4SI7REEEXAMPLE"
                }
            },
            "Description": "APIs commonly used to discover the users, groups,
 policies and permissions in an account, was invoked by IAM principal testuser under
 unusual circumstances. Such activity is not typically seen from this principal.",
            "Service": {
                "Count": 5,
                "Archived": false,
                "ServiceName": "guardduty",
                "EventFirstSeen": "2020-05-26T22:02:24Z",
                "ResourceRole": "TARGET",
                "EventLastSeen": "2020-05-26T22:33:55Z",
                "DetectorId": "d4b040365221be2b54a6264dcexample",
                "Action": {
                    "ActionType": "AWS_API_CALL",
                    "AwsApiCallAction": {
                        "RemoteIpDetails": {
                            "GeoLocation": {
                                "Lat": 51.5164,
                                "Lon": -0.093
                            },
                            "City": {
                                "CityName": "London"
                            },
                            "IpAddressV4": "52.94.36.7",
                            "Organization": {
                                "Org": "Amazon.com",
```

```
                                "Isp": "Amazon.com",
                                "Asn": "16509",
                                "AsnOrg": "AMAZON-02"
                            },
                            "Country": {
                                "CountryName": "United Kingdom"
                            }
                        },
                        "Api": "ListPolicyVersions",
                        "ServiceName": "iam.amazonaws.com",
                        "CallerType": "Remote IP"
                    }
                }
            },
            "Title": "Unusual user permission reconnaissance activity by testuser.",
            "Type": "Recon:IAMUser/UserPermissions",
            "Region": "us-east-1",
            "Partition": "aws",
            "Arn": "arn:aws:guardduty:us-east-1:111122223333:detector/
d4b040365221be2b54a6264dcexample/finding/1ab92989eaf0e742df4a014d5example",
            "UpdatedAt": "2020-05-26T22:55:21.703Z",
            "SchemaVersion": "2.0",
            "Severity": 5,
            "Id": "1ab92989eaf0e742df4a014d5example",
            "CreatedAt": "2020-05-26T22:21:48.385Z",
            "AccountId": "111122223333"
        }
    ]
}
```


有关更多信息，请参阅《GuardDuty 用户指南》中的调查结果。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFindings。

get-ip-set

以下代码示例演示了如何使用 get-ip-set。

Amazon CLI

列出指定可信 IP 集的详细信息

以下 get-ip-set 示例显示指定可信 IP 集的状态和详细信息。

```
aws guardduty get-ip-set \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --ip-set-id d4b94fc952d6912b8f3060768example
```

输出：

```
{
    "Status": "ACTIVE",
    "Location": "s3://amzn-s3-demo-bucket.s3-us-west-2.amazonaws.com/
customlist.csv",
    "Tags": {},
    "Format": "TXT",
    "Name": "test-ip-set-example"
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Working with Trusted IP Lists and Threat Lists。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpSet。

get-master-account

以下代码示例演示了如何使用 get-master-account。

Amazon CLI

检索有关当前区域中主账户的详细信息

以下 get-master-account 示例显示与当前区域中您的检测器关联的主账户的状态和详细信息。

```
aws guardduty get-master-account \
    --detector-id 12abc34d567e8fa901bc2d34eexample
```

输出：

```
{
    "Master": {
        "InvitationId": "04b94d9704854a73f94e061e8example",
        "InvitedAt": "2020-06-09T22:23:04.970Z",
        "RelationshipStatus": "Enabled",
        "AccountId": "111122223333"
```

```
    }
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Understanding the relationship between GuardDuty administrator account and member account。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMasterAccount。

`list-detectors`

以下代码示例演示了如何使用 `list-detectors`。

Amazon CLI

列出当前区域中的可用检测器

以下 `list-detectors` 示例列出了您当前 Amazon 区域中的可用检测器。

```
aws guardduty list-detectors
```

输出：

```
{
    "DetectorIds": [
        "12abc34d567e8fa901bc2d34eexample"
    ]
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的概念和术语。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectors。

`list-findings`

以下代码示例演示了如何使用 `list-findings`。

Amazon CLI

示例 1：列出当前区域的所有调查结果

以下 `list-findings` 示例显示了当前区域按严重性从最高到最低排序的所有 findingIds 的列表。

```
aws guardduty list-findings \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --sort-criteria '{"AttributeName": "severity","OrderBy":"DESC"}'
```

输出：

```
{
    "FindingIds": [
        "04b8ab50fd29c64fc771b232dexample",
        "5ab8ab50fd21373735c826d3aexample",
        "90b93de7aba69107f05bbe60bexample",
        ...
    ]
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的调查结果。

示例 2：列出与当前区域特定调查结果标准相匹配的调查结果

以下 `list-findings` 示例显示了与指定调查结果类型相匹配的所有 findingIds 的列表。

```
aws guardduty list-findings \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --finding-criteria  '{"Criterion":{"type": {"Eq":["UnauthorizedAccess:EC2/
SSHBruteForce"]}}}'
```

输出：

```
{
    "FindingIds": [
        "90b93de7aba69107f05bbe60bexample",
        "6eb9430d7023d30774d6f05e3example",
        "2eb91a2d060ac9a21963a5848example",
        "44b8ab50fd2b0039a9e48f570example",
        "9eb8ab4cd2b7e5b66ba4f5e96example",
        "e0b8ab3a38e9b0312cc390ceeexample"
    ]
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的调查结果。

示例 3：列出与 JSON 文件中定义的一组特定调查结果标准相匹配的当前区域的调查结果

以下 `list-findings` 示例显示了未存档的所有 findingIds 的列表，其中涉及 JSON 文件中指定的名为“testuser”的 IAM 用户。

```
aws guardduty list-findings \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --finding-criteria  file://myfile.json
```

的内容`myfile.json`：

```
{"Criterion": {
    "resource.accessKeyDetails.userName":{
                "Eq":[
                    "testuser"
                    ]
                },
    "service.archived": {
                "Eq": [
                    "false"
                ]
            }
        }
}
```

输出：

```
{
    "FindingIds": [
        "1ab92989eaf0e742df4a014d5example"
    ]
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的调查结果。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFindings。

`list-invitations`

以下代码示例演示了如何使用 `list-invitations`。

Amazon CLI

列出有关您成为当前区域中的成员账户的邀请的详细信息

以下 list-invitations 示例列出了有关您成为当前区域中 GuardDuty 成员账户的邀请的详细信息和状态。

```
aws guardduty list-invitations
```

输出：

```
{
    "Invitations": [
        {
            "InvitationId": "d6b94fb03a66ff665f7db8764example",
            "InvitedAt": "2020-06-10T17:56:38.221Z",
            "RelationshipStatus": "Invited",
            "AccountId": "123456789111"
        }
    ]
}
```

有关更多信息，请参阅《GuardDuty 用户指南》中的通过邀请管理 GuardDuty 账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInvitations。

list-ip-sets

以下代码示例演示了如何使用 list-ip-sets。

Amazon CLI

列出当前区域中的可信 IP 集

以下 list-ip-sets 示例列出了您当前 Amazon 区域中的可信 IP 集。

```
aws guardduty list-ip-sets \
    --detector-id 12abc34d567e8fa901bc2d34eexample
```

输出：

```
{
    "IpSetIds": [
        "d4b94fc952d6912b8f3060768example"
    ]
```

```
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Working with Trusted IP Lists and Threat Lists。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIpSets。

`list-members`

以下代码示例演示了如何使用 `list-members`。

Amazon CLI

示例 1：仅列出当前区域中的当前成员

以下 `list-members` 示例仅列出当前区域中与 GuardDuty 管理员账户关联的当前成员账户并提供其详细信息。

```
aws guardduty list-members \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --only-associated="true"
```

输出：

```
{
    "Members": [
        {
            "RelationshipStatus": "Enabled",
            "InvitedAt": "2020-06-09T22:49:00.910Z",
            "MasterId": "111122223333",
            "DetectorId": "7ab8b2f61b256c87f793f6a86example",
            "UpdatedAt": "2020-06-09T23:08:22.512Z",
            "Email": "your+member@example.com",
            "AccountId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Understanding the relationship between GuardDuty administrator account and member accounts。

示例 2：列出当前区域中的所有成员

以下 list-members 示例列出当前区域中的所有成员账户（包括那些已取消关联或尚未接受来自 GuardDuty 管理员的邀请的成员账户）并提供其详细信息。

```
aws guardduty list-members \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --only-associated="false"
```

输出：

```
{
    "Members": [
        {
            "RelationshipStatus": "Enabled",
            "InvitedAt": "2020-06-09T22:49:00.910Z",
            "MasterId": "111122223333",
            "DetectorId": "7ab8b2f61b256c87f793f6a86example",
            "UpdatedAt": "2020-06-09T23:08:22.512Z",
            "Email": "your+other+member@example.com",
            "AccountId": "555555555555"
        }
    ]
}
```

有关更多信息，请参阅《GuardDuty User Guide》中的 Understanding the relationship between GuardDuty administrator account and member accounts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMembers。

update-ip-set

以下代码示例演示了如何使用 update-ip-set。

Amazon CLI

更新可信 IP 集

以下 update-ip-set 示例演示如何更新可信 IP 集的详细信息。

```
aws guardduty update-ip-set \
    --detector-id 12abc34d567e8fa901bc2d34eexample \
    --ip-set-id d4b94fc952d6912b8f3060768example \
```

```
    --location https://amzn-s3-demo-bucket.s3-us-west-2.amazonaws.com/
customtrustlist2.csv
```

此命令不生成任何输出。

有关更多信息，请参阅《GuardDuty User Guide》中的 Working with Trusted IP Lists and Threat Lists。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIpSet。

Amazon Health使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Health 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-affected-entities`

以下代码示例演示了如何使用 `describe-affected-entities`。

Amazon CLI

列出受指定 Amazon Health 事件影响的实体

以下 `describe-affected-entities` 示例列出了受指定 Amazon Health 事件影响的实体。此事件是 Amazon 账户的账单通知。

```
aws health describe-affected-entities \
    --filter "eventArns=arn:aws:health:global::event/BILLING/
AWS_BILLING_NOTIFICATION/AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-
EXAMPLE11145" \
```

```
    --region us-east-1
```

输出：

```
{
    "entities": [
        {
            "entityArn": "arn:aws:health:global:123456789012:entity/
EXAMPLEimSMoULmWHpb",
            "eventArn": "arn:aws:health:global::event/BILLING/
AWS_BILLING_NOTIFICATION/AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-
EXAMPLE11145",
            "entityValue": "AWS_ACCOUNT",
            "awsAccountId": "123456789012",
            "lastUpdatedTime": 1588356454.08
        }
    ]
}
```

有关更多信息，请参阅《Amazon Health 用户指南》中的事件日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAffectedEntities。

describe-event-details

以下代码示例演示了如何使用 describe-event-details。

Amazon CLI

列出有关 Amazon Health 事件的信息

以下 describe-event-details 示例列出了有关指定 Amazon Health 事件的信息。

```
aws health describe-event-details \
    --event-arns "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111" \
    --region us-east-1
```

输出：

```
{
```

```
    "successfulSet": [
        {
            "event": {
                "arn": "arn:aws:health:us-east-1::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111",
                "service": "EC2",
                "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
                "eventTypeCategory": "issue",
                "region": "us-east-1",
                "startTime": 1587462325.096,
                "endTime": 1587464204.774,
                "lastUpdatedTime": 1587464204.865,
                "statusCode": "closed"
            },
            "eventDescription": {
                "latestDescription": "[RESOLVED] Increased API Error Rates and
 Latencies\n\n[02:45 AM PDT] We are investigating increased API error rates and
 latencies in the US-EAST-1 Region.\n\n[03:16 AM PDT] Between 2:10 AM and 2:59 AM
 PDT we experienced increased API error rates and latencies in the US-EAST-1 Region.
 The issue has been resolved and the service is operating normally."
            }
        }
    ],
    "failedSet": []
}
```

有关更多信息，请参阅《Amazon Health 用户指南》中的事件详细信息窗格。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventDetails。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

示例 1：列出 Amazon Health 事件

以下 describe-events 示例列出了最近的 Amazon Health 事件。

```
aws health describe-events \
    --region us-east-1
```

输出：

```
{
    "events": [
        {
            "arn": "arn:aws:health:us-west-1::event/ECS/AWS_ECS_OPERATIONAL_ISSUE/
AWS_ECS_OPERATIONAL_ISSUE_KWQPY_EXAMPLE111",
            "service": "ECS",
            "eventTypeCode": "AWS_ECS_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-west-1",
            "startTime": 1589077890.53,
            "endTime": 1589086345.597,
            "lastUpdatedTime": 1589086345.905,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:global::event/BILLING/AWS_BILLING_NOTIFICATION/
AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-EXAMPLE1118b",
            "service": "BILLING",
            "eventTypeCode": "AWS_BILLING_NOTIFICATION",
            "eventTypeCategory": "accountNotification",
            "region": "global",
            "startTime": 1588356000.0,
            "lastUpdatedTime": 1588356524.358,
            "statusCode": "open",
            "eventScopeCode": "ACCOUNT_SPECIFIC"
        },
        {
            "arn": "arn:aws:health:us-west-2::event/
CLOUDFORMATION/AWS_CLOUDFORMATION_OPERATIONAL_ISSUE/
AWS_CLOUDFORMATION_OPERATIONAL_ISSUE_OHTWY_EXAMPLE111",
            "service": "CLOUDFORMATION",
            "eventTypeCode": "AWS_CLOUDFORMATION_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-west-2",
            "startTime": 1588279630.761,
            "endTime": 1588284650.0,
            "lastUpdatedTime": 1588284691.941,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
```

```
            "arn": "arn:aws:health:ap-northeast-1::event/LAMBDA/
AWS_LAMBDA_OPERATIONAL_ISSUE/AWS_LAMBDA_OPERATIONAL_ISSUE_JZDND_EXAMPLE111",
            "service": "LAMBDA",
            "eventTypeCode": "AWS_LAMBDA_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "ap-northeast-1",
            "startTime": 1587379534.08,
            "endTime": 1587391771.0,
            "lastUpdatedTime": 1587395689.316,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_COBXJ_EXAMPLE111",
            "service": "EC2",
            "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-east-1",
            "startTime": 1586473044.284,
            "endTime": 1586479706.091,
            "lastUpdatedTime": 1586479706.153,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:global::event/SECURITY/AWS_SECURITY_NOTIFICATION/
AWS_SECURITY_NOTIFICATION_42007387-8129-42da-8c88-EXAMPLE11139",
            "service": "SECURITY",
            "eventTypeCode": "AWS_SECURITY_NOTIFICATION",
            "eventTypeCategory": "accountNotification",
            "region": "global",
            "startTime": 1585674000.0,
            "lastUpdatedTime": 1585674004.132,
            "statusCode": "open",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:global::event/CLOUDFRONT/
AWS_CLOUDFRONT_OPERATIONAL_ISSUE/AWS_CLOUDFRONT_OPERATIONAL_ISSUE_FRQXG_EXAMPLE111",
            "service": "CLOUDFRONT",
            "eventTypeCode": "AWS_CLOUDFRONT_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "global",
```

```
            "startTime": 1585610898.589,
            "endTime": 1585617671.0,
            "lastUpdatedTime": 1585620638.869,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:us-east-1::event/SES/AWS_SES_OPERATIONAL_ISSUE/
AWS_SES_OPERATIONAL_ISSUE_URNDF_EXAMPLE111",
            "service": "SES",
            "eventTypeCode": "AWS_SES_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-east-1",
            "startTime": 1585342008.46,
            "endTime": 1585344017.0,
            "lastUpdatedTime": 1585344355.989,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:global::event/IAM/
AWS_IAM_OPERATIONAL_NOTIFICATION/
AWS_IAM_OPERATIONAL_NOTIFICATION_b6771c34-6ecd-4aea-9d3e-EXAMPLE1117e",
            "service": "IAM",
            "eventTypeCode": "AWS_IAM_OPERATIONAL_NOTIFICATION",
            "eventTypeCategory": "accountNotification",
            "region": "global",
            "startTime": 1584978300.0,
            "lastUpdatedTime": 1584978553.572,
            "statusCode": "open",
            "eventScopeCode": "ACCOUNT_SPECIFIC"
        },
        {
            "arn": "arn:aws:health:ap-southeast-2::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_HNGHE_EXAMPLE111",
            "service": "EC2",
            "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "ap-southeast-2",
            "startTime": 1583881487.483,
            "endTime": 1583885056.785,
            "lastUpdatedTime": 1583885057.052,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Health 用户指南》中的 Amazon Personal Health Dashboard 入门。

示例 2：按服务和事件状态代码列出 Amazon Health 事件

以下 `describe-events` 示例列出了 Amazon Elastic Compute Cloud（Amazon EC2）的 Amazon Health 事件，事件状态为关闭。

```
aws health describe-events \
    --filter "services=EC2,eventStatusCodes=closed"
```

输出：

```
{
    "events": [
        {
            "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111",
            "service": "EC2",
            "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-east-1",
            "startTime": 1587462325.096,
            "endTime": 1587464204.774,
            "lastUpdatedTime": 1587464204.865,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_COBXJ_EXAMPLE111",
            "service": "EC2",
            "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "us-east-1",
            "startTime": 1586473044.284,
            "endTime": 1586479706.091,
            "lastUpdatedTime": 1586479706.153,
```

```
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        },
        {
            "arn": "arn:aws:health:ap-southeast-2::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_HNGHE_EXAMPLE111",
            "service": "EC2",
            "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
            "eventTypeCategory": "issue",
            "region": "ap-southeast-2",
            "startTime": 1583881487.483,
            "endTime": 1583885056.785,
            "lastUpdatedTime": 1583885057.052,
            "statusCode": "closed",
            "eventScopeCode": "PUBLIC"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Health 用户指南》中的 Amazon Personal Health Dashboard 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

使用 Amazon CLI 的 HealthImaging 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 HealthImaging 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

copy-image-set

以下代码示例演示了如何使用 copy-image-set。

Amazon CLI

示例 1：复制没有目标的图像集

以下 copy-image-set 示例制作没有目标的图像集的副本。

```
aws medical-imaging copy-image-set \
    --datastore-id 12345678901234567890123456789012 \
    --source-image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --copy-image-set-information '{"sourceImageSet": {"latestVersionId": "1" } }'
```

输出：

```
{
    "destinationImageSetProperties": {
        "latestVersionId": "2",
        "imageSetWorkflowStatus": "COPYING",
        "updatedAt": 1680042357.432,
        "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
        "imageSetState": "LOCKED",
        "createdAt": 1680042357.432
    },
    "sourceImageSetProperties": {
        "latestVersionId": "1",
        "imageSetWorkflowStatus": "COPYING_WITH_READ_ONLY_ACCESS",
        "updatedAt": 1680042357.432,
        "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
        "imageSetState": "LOCKED",
        "createdAt": 1680027126.436
    },
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 2：复制带有目标的图像集

以下 copy-image-set 示例制作带有目标的图像集的副本。

```
aws medical-imaging copy-image-set \
    --datastore-id 12345678901234567890123456789012 \
    --source-image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --copy-image-set-information '{"sourceImageSet": {"latestVersionId": "1" },
 "destinationImageSet": { "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
 "latestVersionId": "1"} }'
```

输出：

```
{
    "destinationImageSetProperties": {
        "latestVersionId": "2",
        "imageSetWorkflowStatus": "COPYING",
        "updatedAt": 1680042505.135,
        "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
        "imageSetState": "LOCKED",
        "createdAt": 1680042357.432
    },
    "sourceImageSetProperties": {
        "latestVersionId": "1",
        "imageSetWorkflowStatus": "COPYING_WITH_READ_ONLY_ACCESS",
        "updatedAt": 1680042505.135,
        "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
        "imageSetState": "LOCKED",
        "createdAt": 1680027126.436
    },
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 3：将实例子集从源图像集复制到目标图像集

以下 copy-image-set 示例将一个 DICOM 实例从源图像集复制到目标图像集。提供 force 参数是为了覆盖“患者”、“研究”和“系列”级别属性中的不一致。

```
aws medical-imaging copy-image-set \
    --datastore-id 12345678901234567890123456789012 \
    --source-image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --copy-image-set-information '{"sourceImageSet": {"latestVersionId":
 "1","DICOMCopies": {"copiableAttributes": "{\"SchemaVersion\":\"1.1\",\"Study\":
{\"Series\":{\"1.3.6.1.4.1.5962.99.1.3673257865.2104868982.1369432891697.3666.0\":
{\"Instances\":
{\"1.3.6.1.4.1.5962.99.1.3673257865.2104868982.1369432891697.3669.0\":
```

```
{}}}}}}"}},"destinationImageSet": {"imageSetId":
 "b9eb50d8ee682eb9fcf4acbf92f62bb7","latestVersionId": "1"}}' \
    --force
```

输出：

```
{
    "destinationImageSetProperties": {
        "latestVersionId": "2",
        "imageSetWorkflowStatus": "COPYING",
        "updatedAt": 1680042505.135,
        "imageSetId": "b9eb50d8ee682eb9fcf4acbf92f62bb7",
        "imageSetState": "LOCKED",
        "createdAt": 1680042357.432
    },
    "sourceImageSetProperties": {
        "latestVersionId": "1",
        "imageSetWorkflowStatus": "COPYING_WITH_READ_ONLY_ACCESS",
        "updatedAt": 1680042505.135,
        "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
        "imageSetState": "LOCKED",
        "createdAt": 1680027126.436
    },
    "datastoreId": "12345678901234567890123456789012"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的复制图像集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyImageSet。

create-datastore

以下代码示例演示了如何使用 create-datastore。

Amazon CLI

示例 1：创建数据存储

以下 create-datastore 代码示例创建名称为 my-datastore 的数据存储。当您在不指定 --lossless-storage-format 的情况下创建数据存储时，Amazon HealthImaging 默认采用 HTJ2K（高吞吐量 JPEG 2000）。

```
aws medical-imaging create-datastore \
```

```
    --datastore-name "my-datastore"
```

输出：

```
{
    "datastoreId": "12345678901234567890123456789012",
    "datastoreStatus": "CREATING"
}
```

示例 2：采用 JPEG 2000 无损存储格式创建数据存储

使用 JPEG 2000 无损存储格式配置的数据存储将会转码并以 JPEG 2000 格式保存无损图像帧。然后，无需转码即可在 JPEG 2000 无损存储中检索图像帧。以下 create-datastore 代码示例创建配置为 JPEG 2000 无损存储格式的数据存储，其名称为 my-datastore。

```
aws medical-imaging create-datastore \
    --datastore-name "my-datastore" \
    --lossless-storage-format JPEG_2000_LOSSLESS
```

输出：

```
{
    "datastoreId": "12345678901234567890123456789012",
    "datastoreStatus": "CREATING"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的创建数据存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDatastore。

delete-datastore

以下代码示例演示了如何使用 delete-datastore。

Amazon CLI

删除数据存储

以下 delete-datastore 代码示例可删除数据存储。

```
aws medical-imaging delete-datastore \
    --datastore-id "12345678901234567890123456789012"
```

输出：

```
{
    "datastoreId": "12345678901234567890123456789012",
    "datastoreStatus": "DELETING"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的删除数据存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDatastore。

delete-image-set

以下代码示例演示了如何使用 delete-image-set。

Amazon CLI

删除图像集

以下 delete-image-set 代码示例删除图像集。

```
aws medical-imaging delete-image-set \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e
```

输出：

```
{
    "imageSetWorkflowStatus": "DELETING",
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "datastoreId": "12345678901234567890123456789012"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的删除图像集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteImageSet。

get-datastore

以下代码示例演示了如何使用 get-datastore。

Amazon CLI

示例 1：获取数据存储的属性

以下 get-datastore 代码示例可获取数据存储的属性。

```
aws medical-imaging get-datastore \
    --datastore-id 12345678901234567890123456789012
```

输出：

```
{
    "datastoreProperties": {
        "datastoreId": "12345678901234567890123456789012",
        "datastoreName": "TestDatastore123",
        "datastoreStatus": "ACTIVE",
        "losslessStorageFormat": "HTJ2K"
        "datastoreArn": "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012",
        "createdAt": "2022-11-15T23:33:09.643000+00:00",
        "updatedAt": "2022-11-15T23:33:09.643000+00:00"
    }
}
```

示例 2：获取配置为 JPEG2000 的数据存储的属性

以下 get-datastore 代码示例获取配置为 JPEG 2000 无损存储格式的数据存储的属性。

```
aws medical-imaging get-datastore \
    --datastore-id 12345678901234567890123456789012
```

输出：

```
{
    "datastoreProperties": {
        "datastoreId": "12345678901234567890123456789012",
        "datastoreName": "TestDatastore123",
```

```
        "datastoreStatus": "ACTIVE",
        "losslessStorageFormat": "JPEG_2000_LOSSLESS",
        "datastoreArn": "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012",
        "createdAt": "2022-11-15T23:33:09.643000+00:00",
        "updatedAt": "2022-11-15T23:33:09.643000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的获取数据存储属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDatastore。

get-dicom-import-job

以下代码示例演示了如何使用 get-dicom-import-job。

Amazon CLI

获取 dicom 导入作业的属性

以下 get-dicom-import-job 代码示例获取 dicom 导入作业的属性。

```
aws medical-imaging get-dicom-import-job \
    --datastore-id "12345678901234567890123456789012" \
    --job-id "09876543210987654321098765432109"
```

输出：

```
{
    "jobProperties": {
        "jobId": "09876543210987654321098765432109",
        "jobName": "my-job",
        "jobStatus": "COMPLETED",
        "datastoreId": "12345678901234567890123456789012",
        "dataAccessRoleArn": "arn:aws:iam::123456789012:role/
ImportJobDataAccessRole",
        "endedAt": "2022-08-12T11:29:42.285000+00:00",
        "submittedAt": "2022-08-12T11:28:11.152000+00:00",
        "inputS3Uri": "s3://medical-imaging-dicom-input/dicom_input/",
        "outputS3Uri": "s3://medical-imaging-output/
job_output/12345678901234567890123456789012-
DicomImport-09876543210987654321098765432109/"
```

```
    }
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的获取导入作业属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDICOMImportJob。

get-image-frame

以下代码示例演示了如何使用 get-image-frame。

Amazon CLI

获取图像集像素数据

以下 get-image-frame 代码示例获取图像帧。

```
aws medical-imaging get-image-frame \
    --datastore-id "12345678901234567890123456789012" \
    --image-set-id "98765412345612345678907890789012" \
    --image-frame-information imageFrameId=3abf5d5d7ae72f80a0ec81b2c0de3ef4 \
    imageframe.jph
```

注意：此代码示例不包括输出，因为 GetImageFrame 操作会向 imageframe.jph 文件返回像素数据流。有关解码和查看图像帧的信息，请参阅 HTJ2K 解码库。

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的获取图像集像素数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImageFrame。

get-image-set-metadata

以下代码示例演示了如何使用 get-image-set-metadata。

Amazon CLI

示例 1：获取没有版本的图像集元数据

以下 get-image-set-metadata 代码示例获取未指定版本的图像集的元数据。

注意：outfile 是必需的参数。

```
aws medical-imaging get-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    studymetadata.json.gz
```

返回的元数据使用 gzip 压缩并存储在 studymetadata.json.gz 文件中。要查看返回的 JSON 对象的内容，必须先将其解压。

输出：

```
{
    "contentType": "application/json",
    "contentEncoding": "gzip"
}
```

示例 2：获取带有版本的图像集元数据

以下 `get-image-set-metadata` 代码示例获取指定版本的图像集的元数据。

注意：`outfile` 是必需的参数。

```
aws medical-imaging get-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --version-id 1 \
    studymetadata.json.gz
```

返回的元数据使用 gzip 压缩并存储在 studymetadata.json.gz 文件中。要查看返回的 JSON 对象的内容，必须先将其解压。

输出：

```
{
    "contentType": "application/json",
    "contentEncoding": "gzip"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的获取图像集元数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImageSetMetadata。

get-image-set

以下代码示例演示了如何使用 get-image-set。

Amazon CLI

获取图像集属性

以下 get-image-set 代码示例可获取图像集的属性。

```
aws medical-imaging get-image-set \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id 18f88ac7870584f58d56256646b4d92b \
    --version-id 1
```

输出：

```
{
    "versionId": "1",
    "imageSetWorkflowStatus": "COPIED",
    "updatedAt": 1680027253.471,
    "imageSetId": "18f88ac7870584f58d56256646b4d92b",
    "imageSetState": "ACTIVE",
    "createdAt": 1679592510.753,
    "datastoreId": "12345678901234567890123456789012"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的获取映像集属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImageSet。

list-datastores

以下代码示例演示了如何使用 list-datastores。

Amazon CLI

列出数据存储

以下 list-datastores 代码示例列出可用的数据存储。

```
aws medical-imaging list-datastores
```

输出：

```
{
    "datastoreSummaries": [
        {
            "datastoreId": "12345678901234567890123456789012",
            "datastoreName": "TestDatastore123",
            "datastoreStatus": "ACTIVE",
            "datastoreArn": "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012",
            "createdAt": "2022-11-15T23:33:09.643000+00:00",
            "updatedAt": "2022-11-15T23:33:09.643000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的列出数据存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatastores。

`list-dicom-import-jobs`

以下代码示例演示了如何使用 `list-dicom-import-jobs`。

Amazon CLI

列出 dicom 导入作业

以下 `list-dicom-import-jobs` 代码示例列出 dicom 导入作业。

```
aws medical-imaging list-dicom-import-jobs \
    --datastore-id "12345678901234567890123456789012"
```

输出：

```
{
    "jobSummaries": [
        {
            "jobId": "09876543210987654321098765432109",
            "jobName": "my-job",
            "jobStatus": "COMPLETED",
            "datastoreId": "12345678901234567890123456789012",
```

```
            "dataAccessRoleArn": "arn:aws:iam::123456789012:role/
ImportJobDataAccessRole",
            "endedAt": "2022-08-12T11:21:56.504000+00:00",
            "submittedAt": "2022-08-12T11:20:21.734000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的列出导入作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDICOMImportJobs。

list-image-set-versions

以下代码示例演示了如何使用 list-image-set-versions。

Amazon CLI

列出图像集版本

以下 list-image-set-versions 代码示例列出了图像集的版本历史记录。

```
aws medical-imaging list-image-set-versions \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e
```

输出：

```
{
    "imageSetPropertiesList": [
        {
            "ImageSetWorkflowStatus": "UPDATED",
            "versionId": "4",
            "updatedAt": 1680029436.304,
            "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
            "imageSetState": "ACTIVE",
            "createdAt": 1680027126.436
        },
        {
            "ImageSetWorkflowStatus": "UPDATED",
            "versionId": "3",
            "updatedAt": 1680029163.325,
```

```
            "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
            "imageSetState": "ACTIVE",
            "createdAt": 1680027126.436
        },
        {
            "ImageSetWorkflowStatus": "COPY_FAILED",
            "versionId": "2",
            "updatedAt": 1680027455.944,
            "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
            "imageSetState": "ACTIVE",
            "message": "INVALID_REQUEST:  Series of SourceImageSet and
 DestinationImageSet don't match.",
            "createdAt": 1680027126.436
        },
        {
            "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
            "imageSetState": "ACTIVE",
            "versionId": "1",
            "ImageSetWorkflowStatus": "COPIED",
            "createdAt": 1680027126.436
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的列出图像集版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImageSetVersions。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

示例 1：列出数据存储的资源标签

以下 list-tags-for-resource 代码示例列出数据存储的标签。

```
aws medical-imaging list-tags-for-resource \
    --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012"
```

输出：

```
{
    "tags":{
        "Deployment":"Development"
    }
}
```

例 2：列出图像集的资源标签

以下 `list-tags-for-resource` 代码示例列出图像集的标签。

```
aws medical-imaging list-tags-for-resource \
    --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012/
imageset/18f88ac7870584f58d56256646b4d92b"
```

输出：

```
{
    "tags":{
        "Deployment":"Development"
    }
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的标记 Amazon HealthImaging 中的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

search-image-sets

以下代码示例演示了如何使用 `search-image-sets`。

Amazon CLI

示例 1：使用 EQUAL 运算符搜索图像集

以下 `search-image-sets` 代码示例使用 EQUAL 运算符根据特定值搜索图像集。

```
aws medical-imaging search-image-sets \
    --datastore-id 12345678901234567890123456789012 \
    --search-criteria file://search-criteria.json
```

search-criteria.json 的内容

```
{
    "filters": [{
        "values": [{"DICOMPatientId" : "SUBJECT08701"}],
        "operator": "EQUAL"
    }]
}
```

输出：

```
{
    "imageSetsMetadataSummaries": [{
        "imageSetId": "09876543210987654321098765432109",
        "createdAt": "2022-12-06T21:40:59.429000+00:00",
        "version": 1,
        "DICOMTags": {
            "DICOMStudyId": "2011201407",
            "DICOMStudyDate": "19991122",
             "DICOMPatientSex": "F",
             "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
             "DICOMPatientBirthDate": "19201120",
             "DICOMStudyDescription": "UNKNOWN",
             "DICOMPatientId": "SUBJECT08701",
             "DICOMPatientName": "Melissa844 Huel628",
             "DICOMNumberOfStudyRelatedInstances": 1,
             "DICOMStudyTime": "140728",
             "DICOMNumberOfStudyRelatedSeries": 1
            },
        "updatedAt": "2022-12-06T21:40:59.429000+00:00"
    }]
}
```

例 2：使用 DICOMStudyDate 和 DICOMStudyTime，通过 BETWEEN 运算符搜索图像集

以下 search-image-sets 代码示例搜索在 1990 年 1 月 1 日（12:00 AM）至 2023 年 1 月 1 日（12:00 AM）之间生成的 DICOM 研究的图像集。

注意：DICOMStudyTime 为可选项。如果不存在，则上午 12:00（一天的开始）是提供用于筛选的日期的时间值。

```
aws medical-imaging search-image-sets \
```

```
    --datastore-id 12345678901234567890123456789012 \
    --search-criteria file://search-criteria.json
```

search-criteria.json 的内容

```
{
    "filters": [{
        "values": [{
            "DICOMStudyDateAndTime": {
                "DICOMStudyDate": "19900101",
                "DICOMStudyTime": "000000"
            }
        },
        {
            "DICOMStudyDateAndTime": {
                "DICOMStudyDate": "20230101",
                "DICOMStudyTime": "000000"
            }
        }],
        "operator": "BETWEEN"
    }]
}
```

输出：

```
{
    "imageSetsMetadataSummaries": [{
        "imageSetId": "09876543210987654321098765432109",
        "createdAt": "2022-12-06T21:40:59.429000+00:00",
        "version": 1,
        "DICOMTags": {
            "DICOMStudyId": "2011201407",
            "DICOMStudyDate": "19991122",
            "DICOMPatientSex": "F",
            "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
            "DICOMPatientBirthDate": "19201120",
            "DICOMStudyDescription": "UNKNOWN",
            "DICOMPatientId": "SUBJECT08701",
            "DICOMPatientName": "Melissa844 Huel628",
            "DICOMNumberOfStudyRelatedInstances": 1,
            "DICOMStudyTime": "140728",
            "DICOMNumberOfStudyRelatedSeries": 1
        },
```

```
        "updatedAt": "2022-12-06T21:40:59.429000+00:00"
    }]
}
```

例 3：使用 CreatedAt，通过 BETWEEN 运算符搜索图像集（之前保留了时间研究）

以下 `search-image-sets` 代码示例在 UTC 时区的时间范围内搜索 HealthImaging 中保存的 DICOM 研究的图像集。

注意：采用示例中的格式（"1985-04-12T23:20:50.52Z"）提供 createdAt。

```
aws medical-imaging search-image-sets \
    --datastore-id 12345678901234567890123456789012 \
    --search-criteria  file://search-criteria.json
```

`search-criteria.json` 的内容

```
{
    "filters": [{
        "values": [{
            "createdAt": "1985-04-12T23:20:50.52Z"
        },
        {
            "createdAt": "2022-04-12T23:20:50.52Z"
        }],
        "operator": "BETWEEN"
    }]
}
```

输出：

```
{
    "imageSetsMetadataSummaries": [{
        "imageSetId": "09876543210987654321098765432109",
        "createdAt": "2022-12-06T21:40:59.429000+00:00",
        "version": 1,
        "DICOMTags": {
            "DICOMStudyId": "2011201407",
            "DICOMStudyDate": "19991122",
            "DICOMPatientSex": "F",
            "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
            "DICOMPatientBirthDate": "19201120",
```

```
            "DICOMStudyDescription": "UNKNOWN",
            "DICOMPatientId": "SUBJECT08701",
            "DICOMPatientName": "Melissa844 Huel628",
            "DICOMNumberOfStudyRelatedInstances": 1,
            "DICOMStudyTime": "140728",
            "DICOMNumberOfStudyRelatedSeries": 1
        },
        "lastUpdatedAt": "2022-12-06T21:40:59.429000+00:00"
    }]
}
```

示例 4：通过对 DICOMSeriesInstanceUID 使用 EQUAL 运算符，对 updatedAt 使用 BETWEEN 运算符并对 updatedAt 字段按照 ASC 顺序排序响应来搜索图像集

以下 `search-image-sets` 代码示例通过对 DICOMSeriesInstanceUID 使用 EQUAL 运算符，对 updatedAt 使用 BETWEEN 运算符并对 updatedAt 字段按照 ASC 顺序排序响应来搜索图像集。

注意：采用示例中的格式（"1985-04-12T23:20:50.52Z"）提供 updatedAt。

```
aws medical-imaging search-image-sets \
    --datastore-id 12345678901234567890123456789012 \
    --search-criteria  file://search-criteria.json
```

`search-criteria.json` 的内容

```
{
    "filters": [{
        "values": [{
            "updatedAt": "2024-03-11T15:00:05.074000-07:00"
        }, {
            "updatedAt": "2024-03-11T16:00:05.074000-07:00"
        }],
        "operator": "BETWEEN"
    }, {
        "values": [{
            "DICOMSeriesInstanceUID": "1.2.840.99999999.84710745.943275268089"
        }],
        "operator": "EQUAL"
    }],
    "sort": {
        "sortField": "updatedAt",
        "sortOrder": "ASC"
    }
```

```
}
```

输出：

```
{
    "imageSetsMetadataSummaries": [{
        "imageSetId": "09876543210987654321098765432109",
        "createdAt": "2022-12-06T21:40:59.429000+00:00",
        "version": 1,
        "DICOMTags": {
            "DICOMStudyId": "2011201407",
            "DICOMStudyDate": "19991122",
            "DICOMPatientSex": "F",
            "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
            "DICOMPatientBirthDate": "19201120",
            "DICOMStudyDescription": "UNKNOWN",
            "DICOMPatientId": "SUBJECT08701",
            "DICOMPatientName": "Melissa844 Huel628",
            "DICOMNumberOfStudyRelatedInstances": 1,
            "DICOMStudyTime": "140728",
            "DICOMNumberOfStudyRelatedSeries": 1
        },
        "lastUpdatedAt": "2022-12-06T21:40:59.429000+00:00"
    }]
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的搜索图像集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchImageSets。

start-dicom-import-job

以下代码示例演示了如何使用 start-dicom-import-job。

Amazon CLI

启动 dicom 导入作业

以下 start-dicom-import-job 代码示例启动 dicom 导入作业。

```
aws medical-imaging start-dicom-import-job \
    --job-name "my-job" \
    --datastore-id "12345678901234567890123456789012" \
```

```
    --input-s3-uri "s3://medical-imaging-dicom-input/dicom_input/" \
    --output-s3-uri "s3://medical-imaging-output/job_output/" \
    --data-access-role-arn "arn:aws:iam::123456789012:role/ImportJobDataAccessRole"
```

输出：

```
{
    "datastoreId": "12345678901234567890123456789012",
    "jobId": "09876543210987654321098765432109",
    "jobStatus": "SUBMITTED",
    "submittedAt": "2022-08-12T11:28:11.152000+00:00"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的启动导入作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDICOMImportJob。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

示例 1：标记数据存储

以下 tag-resource 代码示例可标记数据存储。

```
aws medical-imaging tag-resource \
  --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012" \
  --tags '{"Deployment":"Development"}'
```

此命令不生成任何输出。

例 2：标记图像集

以下 tag-resource 代码示例可标记图像集。

```
aws medical-imaging tag-resource \
    --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012/
imageset/18f88ac7870584f58d56256646b4d92b" \
```

```
    --tags '{"Deployment":"Development"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的标记 Amazon HealthImaging 中的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：取消标记数据存储

以下 untag-resource 代码示例可取消标记数据存储。

```
aws medical-imaging untag-resource \
    --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012" \
    --tag-keys '["Deployment"]'
```

此命令不生成任何输出。

例 2：取消标记图像集

以下 untag-resource 代码示例可取消标记图像集。

```
aws medical-imaging untag-resource \
    --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012/
imageset/18f88ac7870584f58d56256646b4d92b" \
    --tag-keys '["Deployment"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的标记 Amazon HealthImaging 中的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-image-set-metadata

以下代码示例演示了如何使用 update-image-set-metadata。

Amazon CLI

示例 1：在图像集元数据中插入或更新属性

以下 update-image-set-metadata 示例在图像集元数据中插入或更新属性。

```
aws medical-imaging update-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --latest-version-id 1 \
    --cli-binary-format raw-in-base64-out \
    --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
        "updatableAttributes": "{\"SchemaVersion\":1.1,\"Patient\":{\"DICOM\":
{\"PatientName\":\"MX^MX\"}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 2：从图像集元数据中删除属性

以下 update-image-set-metadata 示例从图像集元数据中删除属性。

```
aws medical-imaging update-image-set-metadata \
```

```
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --latest-version-id 1 \
    --cli-binary-format raw-in-base64-out \
    --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
        "removableAttributes": "{\"SchemaVersion\":1.1,\"Study\":{\"DICOM\":
{\"StudyDescription\":\"CHEST\"}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 3：从图像集元数据中删除实例

以下 update-image-set-metadata 示例从图像集元数据中删除实例。

```
aws medical-imaging update-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
    --latest-version-id 1 \
    --cli-binary-format raw-in-base64-out \
    --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
```

```
        "removableAttributes": "{\"SchemaVersion\": 1.1,\"Study\": {\"Series
\": {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {\"Instances\":
 {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {}}}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 4：将图像集恢复到以前的版本

以下 update-image-set-metadata 示例演示如何将图像集恢复到以前的版本。CopyImageSet 和 UpdateImageSetMetadata 操作可创建图像集的新版本。

```
aws medical-imaging update-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id 53d5fdb05ca4d46ac7ca64b06545c66e \
    --latest-version-id 3 \
    --cli-binary-format raw-in-base64-out \
    --update-image-set-metadata-updates '{"revertToVersionId": "1"}'
```

输出：

```
{
    "datastoreId": "12345678901234567890123456789012",
    "imageSetId": "53d5fdb05ca4d46ac7ca64b06545c66e",
    "latestVersionId": "4",
    "imageSetState": "LOCKED",
    "imageSetWorkflowStatus": "UPDATING",
    "createdAt": 1680027126.436,
    "updatedAt": 1680042257.908
}
```

示例 5：向实例添加私有 DICOM 数据元素

以下 update-image-set-metadata 示例演示如何将私有元素添加到图像集中的指定实例。DICOM 标准允许将私有数据元素用于通信标准数据元素中无法包含的信息。您可以使用 UpdateImageSetMetadata 操作来创建、更新和删除私有数据元素。

```
aws medical-imaging update-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id 53d5fdb05ca4d46ac7ca64b06545c66e \
    --latest-version-id 1 \
    --cli-binary-format raw-in-base64-out \
    --force \
    --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
        "updatableAttributes": "{\"SchemaVersion\": 1.1,\"Study\": {\"Series
\": {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {\"Instances
\": {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {\"DICOM\":
 {\"001910F9\": \"97\"},\"DICOMVRs\": {\"001910F9\": \"DS\"}}}}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "53d5fdb05ca4d46ac7ca64b06545c66e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 6：更新实例的私有 DICOM 数据元素

以下 update-image-set-metadata 示例演示如何更新属于图像集内某个实例的私有数据元素的值。

```
aws medical-imaging update-image-set-metadata \
    --datastore-id 12345678901234567890123456789012 \
    --image-set-id 53d5fdb05ca4d46ac7ca64b06545c66e \
    --latest-version-id 1 \
    --cli-binary-format raw-in-base64-out \
    --force \
    --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
        "updatableAttributes": "{\"SchemaVersion\": 1.1,\"Study\": {\"Series
\": {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {\"Instances
\": {\"1.1.1.1.1.1.12345.123456789012.123.12345678901234.1\": {\"DICOM\":
 {\"00091001\": \"GE_GENESIS_DD\"}}}}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "53d5fdb05ca4d46ac7ca64b06545c66e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

示例 7：使用 force 参数更新 SOPInstanceUID

以下 update-image-set-metadata 示例演示如何通过使用 force 参数覆盖 DICOM 元数据约束来更新 SOPInstanceUID。

```
aws medical-imaging update-image-set-metadata \
        --datastore-id 12345678901234567890123456789012 \
        --image-set-id 53d5fdb05ca4d46ac7ca64b06545c66e \
        --latest-version-id 1 \
        --cli-binary-format raw-in-base64-out \
        --force \
```

```
        --update-image-set-metadata-updates file://metadata-updates.json
```

metadata-updates.json 的内容

```
{
    "DICOMUpdates": {
        "updatableAttributes": "{\"SchemaVersion\":1.1,\"Study\":{\"Series\":
{\"1.3.6.1.4.1.5962.99.1.3633258862.2104868982.1369432891697.3656.0\":{\"Instances
\":{\"1.3.6.1.4.1.5962.99.1.3633258862.2104868982.1369432891697.3659.0\":{\"DICOM\":
{\"SOPInstanceUID\":
\"1.3.6.1.4.1.5962.99.1.3633258862.2104868982.1369432891697.3659.9\"}}}}}}"
    }
}
```

输出：

```
{
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "UPDATING",
    "updatedAt": 1680042257.908,
    "imageSetId": "53d5fdb05ca4d46ac7ca64b06545c66e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
}
```

有关更多信息，请参阅《Amazon HealthImaging 开发人员指南》中的更新图像集元数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateImageSetMetadata。

使用 Amazon CLI 的 HealthLake 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 HealthLake 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

create-fhir-datastore

以下代码示例演示了如何使用 create-fhir-datastore。

Amazon CLI

示例 1：创建已启用 SigV4 的 HealthLake 数据存储

以下 create-fhir-datastore 示例演示如何在 Amazon HealthLake 中创建新的数据存储。

```
aws healthlake create-fhir-datastore \
    --datastore-type-version R4 \
    --datastore-name "FhirTestDatastore"
```

输出：

```
{
    "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/(Data
 store ID)/r4/",
    "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/(Data
 store ID)",
    "DatastoreStatus": "CREATING",
    "DatastoreId": "(Data store ID)"
}
```

示例 2：在已启用 FHIR 的 HealthLake 数据存储中创建 SMART

以下 create-fhir-datastore 示例演示如何在 Amazon HealthLake 中已启用 FHIR 的数据存储中创建新的 SMART。

```
aws healthlake create-fhir-datastore \
    --datastore-name "your-data-store-name" \
    --datastore-type-version R4 \
    --preload-data-config PreloadDataType="SYNTHEA" \
    --sse-configuration '{ "KmsEncryptionConfig": {  "CmkType":
 "CUSTOMER_MANAGED_KMS_KEY", "KmsKeyId": "arn:aws:kms:us-east-1:your-account-id:key/
your-key-id" } }' \
    --identity-provider-configuration  file://identity_provider_configuration.json
```

的内容identity_provider_configuration.json：

```
{
    "AuthorizationStrategy": "SMART_ON_FHIR_V1",
    "FineGrainedAuthorizationEnabled": true,
    "IdpLambdaArn": "arn:aws:lambda:your-region:your-account-id:function:your-
lambda-name",
    "Metadata": "{\"issuer\":\"https://ehr.example.com\", \"jwks_uri\":\"https://
ehr.example.com/.well-known/jwks.json\",\"authorization_endpoint\":\"https://
ehr.example.com/auth/authorize\",\"token_endpoint\":\"https://ehr.token.com/auth/
token\",\"token_endpoint_auth_methods_supported\":[\"client_secret_basic\",\"foo\"],
\"grant_types_supported\":[\"client_credential\",\"foo\"],\"registration_endpoint\":
\"https://ehr.example.com/auth/register\",\"scopes_supported\":[\"openId\",\"profile
\",\"launch\"],\"response_types_supported\":[\"code\"],\"management_endpoint\":
\"https://ehr.example.com/user/manage\",\"introspection_endpoint\":\"https://
ehr.example.com/user/introspect\",\"revocation_endpoint\":\"https://ehr.example.com/
user/revoke\",\"code_challenge_methods_supported\":[\"S256\"],\"capabilities\":
[\"launch-ehr\",\"sso-openid-connect\",\"client-public\"]}"
}
```

输出：

```
{
    "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/(Data
 store ID)/r4/",
    "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/(Data
 store ID)",
    "DatastoreStatus": "CREATING",
    "DatastoreId": "(Data store ID)"
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Creating and monitoring a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateFHIRDatastore。

delete-fhir-datastore

以下代码示例演示了如何使用 delete-fhir-datastore。

Amazon CLI

删除 FHIR 数据存储

以下 delete-fhir-datastore 示例演示如何在 Amazon HealthLake 中删除数据存储及其所有内容。

```
aws healthlake delete-fhir-datastore \
    --datastore-id (Data store ID)
```

输出：

```
{
    "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/(Data
 store ID)/r4/",
    "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/(Data
 store ID)",
    "DatastoreStatus": "DELETING",
    "DatastoreId": "(Data store ID)"
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的“Creating and monitoring a FHIR data store”<https://docs.aws.amazon.com/healthlake/latest/devguide/working-with-FHIR-healthlake.html>。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteFHIRDatastore。

describe-fhir-datastore

以下代码示例演示了如何使用 describe-fhir-datastore。

Amazon CLI

描述 FHIR 数据存储

以下 describe-fhir-datastore 示例演示如何在 Amazon HealthLake 中查找数据存储的属性。

```
aws healthlake describe-fhir-datastore \
    --datastore-id "1f2f459836ac6c513ce899f9e4f66a59"
```

输出：

```
{
    "DatastoreProperties": {
        "PreloadDataConfig": {
```

```
            "PreloadDataType": "SYNTHEA"
        },
        "SseConfiguration": {
            "KmsEncryptionConfig": {
                "CmkType": "CUSTOMER_MANAGED_KMS_KEY",
                "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "DatastoreName": "Demo",
        "DatastoreArn": "arn:aws:healthlake:us-east-1:<AWS Account ID>:datastore/
<Data store ID>",
        "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
<Data store ID>/r4/",
        "DatastoreStatus": "ACTIVE",
        "DatastoreTypeVersion": "R4",
        "CreatedAt": 1603761064.881,
        "DatastoreId": "<Data store ID>",
        "IdentityProviderConfiguration": {
            "AuthorizationStrategy": "AWS_AUTH",
            "FineGrainedAuthorizationEnabled": false
        }
    }
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Creating and monitoring a FHIR data stores。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeFHIRDatastore。

`describe-fhir-export-job`

以下代码示例演示了如何使用 `describe-fhir-export-job`。

Amazon CLI

描述 FHIR 导出作业

以下 `describe-fhir-export-job` 示例演示如何在 Amazon HealthLake 中查找 FHIR 导出作业的属性。

```
aws healthlake describe-fhir-export-job \
```

```
    --datastore-id (Data store ID) \
    --job-id 9b9a51943afaedd0a8c0c26c49135a31
```

输出：

```
{
    "ExportJobProperties": {
        "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",
        "JobStatus": "IN_PROGRESS",
        "JobId": "9009813e9d69ba7cf79bcb3468780f16",
        "SubmitTime": "2024-11-20T11:31:46.672000-05:00",
        "EndTime": "2024-11-20T11:34:01.636000-05:00",
        "OutputDataConfig": {
            "S3Configuration": {
            "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
            "KmsKeyId": "arn:aws:kms:us-east-1:012345678910:key/d330e7fc-b56c-4216-
a250-f4c43ef46e83"
        }

        },
        "DatastoreId": "(Data store ID)"
    }
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Exporting files from a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeFHIRExportJob。

describe-fhir-import-job

以下代码示例演示了如何使用 describe-fhir-import-job。

Amazon CLI

描述 FHIR 导入作业

以下 describe-fhir-import-job 示例演示如何使用 Amazon HealthLake 了解 FHIR 导入作业的属性。

```
aws healthlake describe-fhir-import-job \
    --datastore-id (Data store ID) \
```

```
    --job-id c145fbb27b192af392f8ce6e7838e34f
```

输出：

```
{
    "ImportJobProperties": {
    "InputDataConfig": {
        "S3Uri": "s3://(Bucket Name)/(Prefix Name)/"
        { "arrayitem2": 2 }
    },
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",
    "JobStatus": "COMPLETED",
    "JobId": "c145fbb27b192af392f8ce6e7838e34f",
    "SubmitTime": 1606272542.161,
    "EndTime": 1606272609.497,
    "DatastoreId": "(Data store ID)"
    }
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Importing files to a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeFHIRImportJob。

`list-fhir-datastores`

以下代码示例演示了如何使用 `list-fhir-datastores`。

Amazon CLI

列出 FHIR 数据存储

以下 `list-fhir-datastores` 示例演示如何在 Amazon HealthLake 中使用该命令以及用户如何基于数据存储状态筛选结果。

```
aws healthlake list-fhir-datastores \
    --filter DatastoreStatus=ACTIVE
```

输出：

```
{
```

```
    "DatastorePropertiesList": [
    {
        "PreloadDataConfig": {
            "PreloadDataType": "SYNTHEA"
        },
        "SseConfiguration": {
            "KmsEncryptionConfig": {
                "CmkType": "CUSTOMER_MANAGED_KMS_KEY",
                "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "DatastoreName": "Demo",
        "DatastoreArn": "arn:aws:healthlake:us-east-1:<AWS Account ID>:datastore/
<Data store ID>",
        "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
<Data store ID>/r4/",
        "DatastoreStatus": "ACTIVE",
        "DatastoreTypeVersion": "R4",
        "CreatedAt": 1603761064.881,
        "DatastoreId": "<Data store ID>",
        "IdentityProviderConfiguration": {
            "AuthorizationStrategy": "AWS_AUTH",
            "FineGrainedAuthorizationEnabled": false
        }
    }
    ]
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Creating and monitoring a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListFHIRDatastores。

`list-fhir-export-jobs`

以下代码示例演示了如何使用 `list-fhir-export-jobs`。

Amazon CLI

列出所有 FHIR 导出作业

以下 `list-fhir-export-jobs` 示例演示了如何使用该命令查看与账户关联的导出作业列表。

```
aws healthlake list-fhir-export-jobs \
    --datastore-id (Data store ID) \
    --submitted-before (DATE like 2024-10-13T19:00:00Z)\
    --submitted-after (DATE like 2020-10-13T19:00:00Z )\
    --job-name "FHIR-EXPORT" \
    --job-status SUBMITTED  \
    --max-results (Integer between 1 and 500)
```

输出：

```
{
    "ExportJobPropertiesList": [
        {
            "ExportJobProperties": {
                "OutputDataConfig": {
                    "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
                    "S3Configuration": {
                        "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
                        "KmsKeyId": "(KmsKey Id)"
                    }
                },
                "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role
 Name)",
                "JobStatus": "COMPLETED",
                "JobId": "c145fbb27b192af392f8ce6e7838e34f",
                "JobName": "FHIR-EXPORT",
                "SubmitTime": "2024-11-20T11:31:46.672000-05:00",
                "EndTime": "2024-11-20T11:34:01.636000-05:00",
                "DatastoreId": "(Data store ID)"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Exporting files from a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListFHIRExportJobs。

`list-fhir-import-jobs`

以下代码示例演示了如何使用 `list-fhir-import-jobs`。

Amazon CLI

列出所有 FHIR 导入作业

以下 `list-fhir-import-jobs` 示例演示了如何使用该命令查看与账户关联的所有导入作业的列表。

```
aws healthlake list-fhir-import-jobs \
    --datastore-id (Data store ID) \
    --submitted-before (DATE like 2024-10-13T19:00:00Z) \
    --submitted-after (DATE like 2020-10-13T19:00:00Z ) \
    --job-name "FHIR-IMPORT" \
    --job-status SUBMITTED  \
    -max-results (Integer between 1 and 500)
```

输出：

```
{
    "ImportJobPropertiesList": [
        {
            "JobId": "c0fddbf76f238297632d4aebdbfc9ddf",
            "JobStatus": "COMPLETED",
            "SubmitTime": "2024-11-20T10:08:46.813000-05:00",
            "EndTime": "2024-11-20T10:10:09.093000-05:00",
            "DatastoreId": "(Data store ID)",
            "InputDataConfig": {
                "S3Uri": "s3://(Bucket Name)/(Prefix Name)/"
            },
            "JobOutputDataConfig": {
                "S3Configuration": {
                    "S3Uri": "s3://(Bucket Name)/
import/6407b9ae4c2def3cb6f1a46a0c599ec0-FHIR_IMPORT-
c0fddbf76f238297632d4aebdbfc9ddf/",
                    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/b7f645cb-
e564-4981-8672-9e012d1ff1a0"
                }
            },
            "JobProgressReport": {
                "TotalNumberOfScannedFiles": 1,
                "TotalSizeOfScannedFilesInMB": 0.001798,
                "TotalNumberOfImportedFiles": 1,
                "TotalNumberOfResourcesScanned": 1,
                "TotalNumberOfResourcesImported": 1,
```

```
                "TotalNumberOfResourcesWithCustomerError": 0,
                "TotalNumberOfFilesReadWithCustomerError": 0,
                "Throughput": 0.0
            },
            "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Importing files to FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListFHIRImportJobs。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出数据存储的标签

以下 `list-tags-for-resource` 示例列出与指定数据存储关联的标签。

```
aws healthlake list-tags-for-resource \
    --resource-arn "arn:aws:healthlake:us-east-1:123456789012:datastore/
fhir/0725c83f4307f263e16fd56b6d8ebdbe"
```

输出：

```
{
    "tags": {
        "key": "value",
        "key1": "value1"
    }
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Tagging resources in Amazon HealthLake。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-fhir-export-job

以下代码示例演示了如何使用 start-fhir-export-job。

Amazon CLI

启动 FHIR 导出作业

以下 start-fhir-export-job 示例演示如何使用 Amazon HealthLake 启动 FHIR 导出作业。

```
aws healthlake start-fhir-export-job \
    --output-data-config '{"S3Configuration": {"S3Uri":"s3://(Bucket Name)/(Prefix Name)/","KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/d330e7fc-b56c-4216-a250-f4c43ef46e83"}}' \
    --datastore-id (Data store ID) \
    --data-access-role-arn arn:aws:iam::(AWS Account ID):role/(Role Name)
```

输出：

```
{
    "DatastoreId": "(Data store ID)",
    "JobStatus": "SUBMITTED",
    "JobId": "9b9a51943afaedd0a8c0c26c49135a31"
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Exporting files from a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartFHIRExportJob。

start-fhir-import-job

以下代码示例演示了如何使用 start-fhir-import-job。

Amazon CLI

启动 FHIR 导入作业

以下 start-fhir-import-job 示例演示如何使用 Amazon HealthLake 启动 FHIR 导入作业。

```
aws healthlake start-fhir-import-job \
```

```
    --input-data-config S3Uri="s3://(Bucket Name)/(Prefix Name)/" \
    --job-output-data-config '{"S3Configuration": {"S3Uri":"s3://(Bucket Name)/
(Prefix Name)/","KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/d330e7fc-
b56c-4216-a250-f4c43ef46e83"}}' \
    --datastore-id (Data store ID) \
    --data-access-role-arn "arn:aws:iam::(AWS Account ID):role/(Role Name)"
```

输出：

```
{
    "DatastoreId": "(Data store ID)",
    "JobStatus": "SUBMITTED",
    "JobId": "c145fbb27b192af392f8ce6e7838e34f"
}
```

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Importing files to a FHIR data store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartFHIRImportJob。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向数据存储中添加标签

以下 `tag-resource` 示例演示如何向数据存储中添加标签。

```
aws healthlake tag-resource \
    --resource-arn "arn:aws:healthlake:us-east-1:123456789012:datastore/
fhir/0725c83f4307f263e16fd56b6d8ebdbe" \
    --tags '[{"Key": "key1", "Value": "value1"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Adding a tag to a data store。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从数据存储中移除标签

以下 untag-resource 示例演示如何从数据存储中移除标签。

```
aws healthlake untag-resource \
    --resource-arn "arn:aws:healthlake:us-east-1:123456789012:datastore/fhir/
b91723d65c6fdeb1d26543a49d2ed1fa" \
    --tag-keys '["key1"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon HealthLake Developer Guide》中的 Removing tags from a data store。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 HealthOmics 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 HealthOmics 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

abort-multipart-read-set-upload

以下代码示例演示了如何使用 abort-multipart-read-set-upload。

Amazon CLI

停止分段读取集上传

以下 abort-multipart-read-set-upload 示例停止将读取集分段上传到 HealthOmics 序列存储。

```
aws omics abort-multipart-read-set-upload \
    --sequence-store-id 0123456789 \
    --upload-id 1122334455
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortMultipartReadSetUpload。

accept-share

以下代码示例演示了如何使用 accept-share。

Amazon CLI

接受分析存储数据共享

以下 accept-share 示例接受 HealthOmics 分析存储数据共享。

```
aws omics accept-share \
    ----share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

输出：

```
{
    "status": "ACTIVATING"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的跨账户共享。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptShare。

batch-delete-read-set

以下代码示例演示了如何使用 batch-delete-read-set。

Amazon CLI

删除多个读取集

以下 batch-delete-read-set 示例删除了两个读取集。

```
aws omics batch-delete-read-set \
    --sequence-store-id 1234567890 \
    --ids 1234567890 0123456789
```

如果删除任意指定的读取集时出错，服务将返回错误列表。

```
{
    "errors": [
        {
            "code": "",
            "id": "0123456789",
            "message": "The specified readset does not exist."
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteReadSet。

cancel-annotation-import-job

以下代码示例演示了如何使用 cancel-annotation-import-job。

Amazon CLI

取消注释导入作业

以下 cancel-annotation-import-job 示例取消了 ID 为 04f57618-xmpl-4fd0-9349-e5a85aefb997 的注释导入作业。

```
aws omics cancel-annotation-import-job \
    --job-id 04f57618-xmpl-4fd0-9349-e5a85aefb997
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelAnnotationImportJob。

cancel-run

以下代码示例演示了如何使用 cancel-run。

Amazon CLI

取消运行

以下 cancel-run 示例取消了 ID 为 1234567 的运行。

```
aws omics cancel-run \
    --id 1234567
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Run lifecycle in a workflow。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelRun。

cancel-variant-import-job

以下代码示例演示了如何使用 cancel-variant-import-job。

Amazon CLI

取消变体导入作业

以下 cancel-variant-import-job 示例取消了 ID 为 69cb65d6-xmpl-4a4a-9025-4565794b684e 的变体导入作业。

```
aws omics cancel-variant-import-job \
    --job-id 69cb65d6-xmpl-4a4a-9025-4565794b684e
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelVariantImportJob。

complete-multipart-read-set-upload

以下代码示例演示了如何使用 complete-multipart-read-set-upload。

Amazon CLI

上传所有组件后结束分段上传。

以下 complete-multipart-read-set-upload 示例将在所有组件都上传完毕后结束向序列存储的分段上传。

```
aws omics complete-multipart-read-set-upload \
    --sequence-store-id 0123456789 \
    --upload-id 1122334455 \
    --parts '[{"checksum":"gaCBQMe+rpCFZxLpoP6gydBoXaKKDA/
Vobh5zBDb4W4=","partNumber":1,"partSource":"SOURCE1"}]'
```

输出：

```
{
    "readSetId": "0000000001"
    "readSetId": "0000000002"
    "readSetId": "0000000003"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteMultipartReadSetUpload。

create-annotation-store-version

以下代码示例演示了如何使用 create-annotation-store-version。

Amazon CLI

创建注释存储的新版本

以下 create-annotation-store-version 示例创建注释存储的新版本。

```
aws omics create-annotation-store-version \
    --name my_annotation_store \
    --version-name my_version
```

输出：

```
{
    "creationTime": "2023-07-21T17:15:49.251040+00:00",
    "id": "3b93cdef69d2",
    "name": "my_annotation_store",
    "reference": {
```

```
        "referenceArn": "arn:aws:omics:us-
west-2:555555555555:referenceStore/6505293348/reference/5987565360"
    },
    "status": "CREATING",
    "versionName": "my_version"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的创建注释存储的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAnnotationStoreVersion。

create-annotation-store

以下代码示例演示了如何使用 create-annotation-store。

Amazon CLI

示例 1：创建 VCF 注释存储

以下 create-annotation-store 示例创建 VCF 格式的注释存储。

```
aws omics create-annotation-store \
    --name my_ann_store \
    --store-format VCF \
    --reference referenceArn=arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890
```

输出：

```
{
    "creationTime": "2022-11-23T22:48:39.226492Z",
    "id": "0a91xmplc71f",
    "name": "my_ann_store",
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "CREATING",
    "storeFormat": "VCF"
}
```

示例 2：创建 TSV 注释存储

以下 create-annotation-store 示例创建 TSV 格式的注释存储。

```
aws omics create-annotation-store \
    --name tsv_ann_store \
    --store-format TSV \
    --reference referenceArn=arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890 \
    --store-options file://tsv-store-options.json
```

tsv-store-options.json 为注释配置格式选项。

```
{
    "tsvStoreOptions": {
        "annotationType": "CHR_START_END_ZERO_BASE",
        "formatToHeader": {
            "CHR": "chromosome",
            "START": "start",
            "END": "end"
        },
        "schema": [
            {
                "chromosome": "STRING"
            },
            {
                "start": "LONG"
            },
            {
                "end": "LONG"
            },
            {
                "name": "STRING"
            }
        ]
    }
}
```

输出：

```
{
    "creationTime": "2022-11-30T01:28:08.525586Z",
    "id": "861cxmpl96b0",
    "name": "tsv_ann_store",
```

```
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "CREATING",
    "storeFormat": "TSV",
    "storeOptions": {
        "tsvStoreOptions": {
            "annotationType": "CHR_START_END_ZERO_BASE",
            "formatToHeader": {
                "CHR": "chromosome",
                "END": "end",
                "START": "start"
            },
            "schema": [
                {
                    "chromosome": "STRING"
                },
                {
                    "start": "LONG"
                },
                {
                    "end": "LONG"
                },
                {
                    "name": "STRING"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAnnotationStore。

create-multipart-read-set-upload

以下代码示例演示了如何使用 create-multipart-read-set-upload。

Amazon CLI

开始分段读取集上传。

以下 create-multipart-read-set-upload 示例启动了分段读取集上传。

```
aws omics create-multipart-read-set-upload \
    --sequence-store-id 0123456789 \
    --name HG00146 \
    --source-file-type FASTQ \
    --subject-id mySubject\
    --sample-id mySample\
    --description "FASTQ for HG00146"\
    --generated-from "1000 Genomes"
```

输出：

```
{
    "creationTime": "2022-07-13T23:25:20Z",
    "description": "FASTQ for HG00146",
    "generatedFrom": "1000 Genomes",
    "name": "HG00146",
    "sampleId": "mySample",
    "sequenceStoreId": "0123456789",
    "sourceFileType": "FASTQ",
    "subjectId": "mySubject",
    "uploadId": "1122334455"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMultipartReadSetUpload。

create-reference-store

以下代码示例演示了如何使用 create-reference-store。

Amazon CLI

创建参考存储

以下 create-reference-store 示例创建参考存储 my-ref-store。

```
aws omics create-reference-store \
    --name my-ref-store
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",
    "creationTime": "2022-11-22T22:13:25.947Z",
    "id": "1234567890",
    "name": "my-ref-store"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReferenceStore。

`create-run-group`

以下代码示例演示了如何使用 `create-run-group`。

Amazon CLI

创建运行组

以下 `create-run-group` 示例创建名为 `cram-converter` 的运行组。

```
aws omics create-run-group \
    --name cram-converter \
    --max-cpus 20 \
    --max-gpus 10 \
    --max-duration 600 \
    --max-runs 5
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
    "id": "1234567",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating run groups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRunGroup。

create-sequence-store

以下代码示例演示了如何使用 create-sequence-store。

Amazon CLI

创建序列存储

以下 create-sequence-store 示例创建序列存储。

```
aws omics create-sequence-store \
    --name my-seq-store
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890",
    "creationTime": "2022-11-23T01:24:33.629Z",
    "id": "1234567890",
    "name": "my-seq-store"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSequenceStore。

create-share

以下代码示例演示了如何使用 create-share。

Amazon CLI

创建 HealthOmics 分析存储的共享

以下 create-share 示例演示了如何创建可供账户外部订阅用户接受的 HealthOmics 分析存储共享。

```
aws omics create-share \
    --resource-arn "arn:aws:omics:us-west-2:555555555555:variantStore/
omics_dev_var_store" \
    --principal-subscriber "123456789012" \
    --name "my_Share-123"
```

输出：

```
{
    "shareId": "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a",
    "name": "my_Share-123",
    "status": "PENDING"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的跨账户共享。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateShare。

create-variant-store

以下代码示例演示了如何使用 create-variant-store。

Amazon CLI

创建变体存储

以下 create-variant-store 示例创建名为 my_var_store 的变体存储。

```
aws omics create-variant-store \
    --name my_var_store \
    --reference referenceArn=arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890
```

输出：

```
{
    "creationTime": "2022-11-23T22:09:07.534499Z",
    "id": "02dexmplcfdd",
    "name": "my_var_store",
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "CREATING"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVariantStore。

create-workflow

以下代码示例演示了如何使用 create-workflow。

Amazon CLI

创建工作流

以下 create-workflow 示例创建一个 WDL 工作流。

```
aws omics create-workflow \
    --name cram-converter \
    --engine WDL \
    --definition-zip fileb://workflow-crambam.zip \
    --parameter-template file://workflow-params.json
```

workflow-crambam.zip 是包含工作流定义的 ZIP 存档。workflow-params.json 定义了工作流的运行时参数。

```
{
    "ref_fasta" : {
        "description": "Reference genome fasta file",
        "optional": false
    },
    "ref_fasta_index" : {
        "description": "Index of the reference genome fasta file",
        "optional": false
    },
    "ref_dict" : {
        "description": "dictionary file for 'ref_fasta'",
        "optional": false
    },
    "input_cram" : {
        "description": "The Cram file to convert to BAM",
        "optional": false
    },
    "sample_name" : {
        "description": "The name of the input sample, used to name the output BAM",
        "optional": false
    }
}
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
    "id": "1234567",
    "status": "CREATING",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating private workflows。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWorkflow。

delete-annotation-store-versions

以下代码示例演示了如何使用 delete-annotation-store-versions。

Amazon CLI

删除注释存储版本

以下 delete-annotation-store-versions 示例删除注释存储版本。

```
aws omics delete-annotation-store-versions \
    --name my_annotation_store \
    --versions my_version
```

输出：

```
{
    "errors": []
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的创建注释存储的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAnnotationStoreVersions。

delete-annotation-store

以下代码示例演示了如何使用 delete-annotation-store。

Amazon CLI

删除注释存储

以下 delete-annotation-store 示例删除了名为 my_vcf_store 的注释存储。

```
aws omics delete-annotation-store \
    --name my_vcf_store
```

输出：

```
{
    "status": "DELETING"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAnnotationStore。

delete-reference-store

以下代码示例演示了如何使用 delete-reference-store。

Amazon CLI

删除参考存储

以下 delete-reference-store 示例删除了 ID 为 1234567890 的参考存储。

```
aws omics delete-reference-store \
    --id 1234567890
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReferenceStore。

delete-reference

以下代码示例演示了如何使用 delete-reference。

Amazon CLI

删除参考

以下 delete-reference 示例删除了参考。

```
aws omics delete-reference \
    --reference-store-id 1234567890 \
    --id 1234567890
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReference。

delete-run-group

以下代码示例演示了如何使用 delete-run-group。

Amazon CLI

删除运行组

以下 delete-run-group 示例删除了 ID 为 1234567 的运行组。

```
aws omics delete-run-group \
    --id 1234567
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Deleting runs and run groups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRunGroup。

delete-run

以下代码示例演示了如何使用 delete-run。

Amazon CLI

删除工作流运行

以下 delete-run 示例删除了 ID 为 1234567 的运行。

```
aws omics delete-run \
    --id 1234567
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Deleting runs and run groups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRun。

delete-sequence-store

以下代码示例演示了如何使用 delete-sequence-store。

Amazon CLI

删除序列存储

以下 delete-sequence-store 示例删除了 ID 为 1234567890 的序列存储。

```
aws omics delete-sequence-store \
    --id 1234567890
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSequenceStore。

delete-share

以下代码示例演示了如何使用 delete-share。

Amazon CLI

删除 HealthOmics 分析数据共享

以下 delete-share 示例删除了分析数据的跨账户共享。

```
aws omics delete-share \
    --share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

输出：

```
{
    "status": "DELETING"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的跨账户共享。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteShare。

delete-variant-store

以下代码示例演示了如何使用 delete-variant-store。

Amazon CLI

删除变体存储

以下 delete-variant-store 示例删除了名为 my_var_store 的变体存储。

```
aws omics delete-variant-store \
    --name my_var_store
```

输出：

```
{
    "status": "DELETING"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVariantStore。

delete-workflow

以下代码示例演示了如何使用 delete-workflow。

Amazon CLI

删除工作流

以下 delete-workflow 示例删除了 ID 为 1234567 的工作流。

```
aws omics delete-workflow \
    --id 1234567
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Delete a private workflow。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWorkflow。

get-annotation-import-job

以下代码示例演示了如何使用 get-annotation-import-job。

Amazon CLI

查看注释导入作业

以下 get-annotation-import-job 示例获取了有关注释导入作业的详细信息。

```
aws omics get-annotation-import-job \
    --job-id 984162c7-xmpl-4d23-ab47-286f7950bfbf
```

输出：

```
{
    "creationTime": "2022-11-30T01:40:11.017746Z",
    "destinationName": "tsv_ann_store",
    "id": "984162c7-xmpl-4d23-ab47-286f7950bfbf",
    "items": [
        {
            "jobStatus": "COMPLETED",
            "source": "s3://omics-artifacts-01d6xmpl4e72dd32/targetedregions.bed.gz"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
    "runLeftNormalization": false,
    "status": "COMPLETED",
    "updateTime": "2022-11-30T01:42:39.134009Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAnnotationImportJob。

get-annotation-store-version

以下代码示例演示了如何使用 get-annotation-store-version。

Amazon CLI

检索注解存储版本的元数据

以下 get-annotation-store-version 示例检索了所请求的注释存储版本的元数据。

```
aws omics get-annotation-store-version \
    --name my_annotation_store \
    --version-name my_version
```

输出：

```
{
    "storeId": "4934045d1c6d",
    "id": "2a3f4a44aa7b",
    "status": "ACTIVE",
    "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version",
    "name": "my_annotation_store",
    "versionName": "my_version",
    "creationTime": "2023-07-21T17:15:49.251040+00:00",
    "updateTime": "2023-07-21T17:15:56.434223+00:00",
    "statusMessage": "",
    "versionSizeBytes": 0
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的创建注释存储的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAnnotationStoreVersion。

get-annotation-store

以下代码示例演示了如何使用 get-annotation-store。

Amazon CLI

查看注释存储

以下 get-annotation-store 示例获取名为 my_ann_store 的注释存储。

```
aws omics get-annotation-store \
    --name my_ann_store
```

输出：

```
{
    "creationTime": "2022-11-23T22:48:39.226492Z",
    "id": "0a91xmplc71f",
    "name": "my_ann_store",
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
```

```
    "status": "CREATING",
    "storeArn": "arn:aws:omics:us-west-2:123456789012:annotationStore/my_ann_store",
    "storeFormat": "VCF",
    "storeSizeBytes": 0,
    "tags": {}
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAnnotationStore。

get-read-set-activation-job

以下代码示例演示了如何使用 get-read-set-activation-job。

Amazon CLI

查看读取集激活作业

以下 get-read-set-activation-job 示例获取有关读取集激活作业的详细信息。

```
aws omics get-read-set-activation-job \
    --sequence-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "completionTime": "2022-12-06T22:33:42.828Z",
    "creationTime": "2022-12-06T22:32:45.213Z",
    "id": "1234567890",
    "sequenceStoreId": "1234567890",
    "sources": [
        {
            "readSetId": "1234567890",
            "status": "FINISHED",
            "statusMessage": "No activation needed as read set is already in
 ACTIVATING or ACTIVE state."
        }
    ],
    "status": "COMPLETED",
    "statusMessage": "The job completed successfully."
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReadSetActivationJob。

get-read-set-export-job

以下代码示例演示了如何使用 get-read-set-export-job。

Amazon CLI

查看读取集导出作业

以下 get-read-set-export-job 示例获取有关读取集导出作业的详细信息。

```
aws omics get-read-set-export-job \
    --sequence-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "completionTime": "2022-12-06T22:39:14.491Z",
    "creationTime": "2022-12-06T22:37:18.612Z",
    "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",
    "id": "1234567890",
    "sequenceStoreId": "1234567890",
    "status": "COMPLETED",
    "statusMessage": "The job is submitted and will start soon."
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReadSetExportJob。

get-read-set-import-job

以下代码示例演示了如何使用 get-read-set-import-job。

Amazon CLI

查看读取集导入作业

以下 get-read-set-import-job 示例获取有关读取集导入作业的详细信息。

```
aws omics get-read-set-import-job \
    --sequence-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "creationTime": "2022-11-23T01:36:38.158Z",
    "id": "1234567890",
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
    "sequenceStoreId": "1234567890",
    "sources": [
        {
            "name": "HG00100",
            "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
            "sampleId": "bam-sample",
            "sourceFileType": "BAM",
            "sourceFiles": {
                "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00100.chrom20.ILLUMINA.bwa.GBR.low_coverage.20101123.bam",
                "source2": ""
            },
            "status": "IN_PROGRESS",
            "statusMessage": "The source job is currently in progress.",
            "subjectId": "bam-subject",
            "tags": {
                "aws:omics:sampleId": "bam-sample",
                "aws:omics:subjectId": "bam-subject"
            }
        },
        {
            "name": "HG00146",
            "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
            "sampleId": "fastq-sample",
            "sourceFileType": "FASTQ",
            "sourceFiles": {
                "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
SRR233106_1.filt.fastq.gz",
                "source2": "s3://omics-artifacts-01d6xmpl4e72dd32/
SRR233106_2.filt.fastq.gz"
```

```
            },
            "status": "IN_PROGRESS",
            "statusMessage": "The source job is currently in progress.",
            "subjectId": "fastq-subject",
            "tags": {
                "aws:omics:sampleId": "fastq-sample",
                "aws:omics:subjectId": "fastq-subject"
            }
        },
        {
            "name": "HG00096",
            "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
            "sampleId": "cram-sample",
            "sourceFileType": "CRAM",
            "sourceFiles": {
                "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00096.alt_bwamem_GRCh38DH.20150718.GBR.low_coverage.cram",
                "source2": ""
            },
            "status": "IN_PROGRESS",
            "statusMessage": "The source job is currently in progress.",
            "subjectId": "cram-subject",
            "tags": {
                "aws:omics:sampleId": "cram-sample",
                "aws:omics:subjectId": "cram-subject"
            }
        }
    ],
    "status": "IN_PROGRESS",
    "statusMessage": "The job is currently in progress."
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReadSetImportJob。

get-read-set-metadata

以下代码示例演示了如何使用 get-read-set-metadata。

Amazon CLI

查看读取集

以下 get-read-set-metadata 示例获取有关读取集文件的详细信息。

```
aws omics get-read-set-metadata \
    --sequence-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890/
readSet/1234567890",
    "creationTime": "2022-11-23T21:55:00.515Z",
    "fileType": "FASTQ",
    "files": {
        "source1": {
            "contentLength": 310054739,
            "partSize": 104857600,
            "totalParts": 3
        },
        "source2": {
            "contentLength": 307846621,
            "partSize": 104857600,
            "totalParts": 3
        }
    },
    "id": "1234567890",
    "name": "HG00146",
    "referenceArn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
    "sampleId": "fastq-sample",
    "sequenceInformation": {
        "alignment": "UNALIGNED",
        "totalBaseCount": 677717384,
        "totalReadCount": 8917334
    },
    "sequenceStoreId": "1234567890",
    "status": "ACTIVE",
    "subjectId": "fastq-subject"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReadSetMetadata。

get-read-set

以下代码示例演示了如何使用 get-read-set。

Amazon CLI

下载读取集

以下 get-read-set 示例将读取集的第 3 部分下载为 1234567890.3.bam。

```
aws omics get-read-set \
    --sequence-store-id 1234567890 \
    --id 1234567890 \
    --part-number 3  1234567890.3.bam
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReadSet。

get-reference-import-job

以下代码示例演示了如何使用 get-reference-import-job。

Amazon CLI

查看参考导入作业

以下 get-reference-import-job 示例获取有关参考导入任务的详细信息。

```
aws omics get-reference-import-job \
    --reference-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "creationTime": "2022-11-22T22:25:41.124Z",
    "id": "1234567890",
    "referenceStoreId": "1234567890",
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
```

```
    "sources": [
        {
            "name": "assembly-38",
            "sourceFile": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta",
            "status": "IN_PROGRESS",
            "statusMessage": "The source job is currently in progress."
        }
    ],
    "status": "IN_PROGRESS",
    "statusMessage": "The job is currently in progress."
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReferenceImportJob。

get-reference-metadata

以下代码示例演示了如何使用 get-reference-metadata。

Amazon CLI

查看参考

以下 get-reference-metadata 示例获取有关参考的详细信息。

```
aws omics get-reference-metadata \
    --reference-store-id 1234567890 \
    --id 1234567890
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
    "creationTime": "2022-11-22T22:27:09.033Z",
    "files": {
        "index": {
            "contentLength": 160928,
            "partSize": 104857600,
            "totalParts": 1
```

```
        },
        "source": {
            "contentLength": 3249912778,
            "partSize": 104857600,
            "totalParts": 31
        }
    },
    "id": "1234567890",
    "md5": "7ff134953dcca8c8997453bbb80b6b5e",
    "name": "assembly-38",
    "referenceStoreId": "1234567890",
    "status": "ACTIVE",
    "updateTime": "2022-11-22T22:27:09.033Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReferenceMetadata。

get-reference-store

以下代码示例演示了如何使用 get-reference-store。

Amazon CLI

查看参考存储

以下 get-reference-store 示例获取有关参考存储的详细信息。

```
aws omics get-reference-store \
    --id 1234567890
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",
    "creationTime": "2022-09-23T23:27:20.364Z",
    "id": "1234567890",
    "name": "my-rstore-0"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReferenceStore。

get-reference

以下代码示例演示了如何使用 get-reference。

Amazon CLI

下载基因组参考

以下 get-reference 示例将基因组的第 1 部分下载为 hg38.1.fa。

```
aws omics get-reference \
    --reference-store-id 1234567890 \
    --id 1234567890 \
    --part-number 1 hg38.1.fa
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReference。

get-run-group

以下代码示例演示了如何使用 get-run-group。

Amazon CLI

查看运行组

以下 get-run-group 示例获取有关运行组的详细信息。

```
aws omics get-run-group \
    --id 1234567
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
    "creationTime": "2022-12-01T00:58:42.915219Z",
    "id": "1234567",
    "maxCpus": 20,
```

```
    "maxDuration": 600,
    "name": "cram-convert",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating run groups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRunGroup。

get-run-task

以下代码示例演示了如何使用 get-run-task。

Amazon CLI

查看任务

以下 get-run-task 示例获取有关工作流任务的详细信息。

```
aws omics get-run-task \
    --id 1234567 \
    --task-id 1234567
```

输出：

```
{
    "cpus": 1,
    "creationTime": "2022-11-30T23:13:00.718651Z",
    "logStream": "arn:aws:logs:us-west-2:123456789012:log-group:/aws/omics/
WorkflowLog:log-stream:run/1234567/task/1234567",
    "memory": 15,
    "name": "CramToBamTask",
    "startTime": "2022-11-30T23:17:47.016Z",
    "status": "COMPLETED",
    "stopTime": "2022-11-30T23:18:21.503Z",
    "taskId": "1234567"
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Task lifecycle in a HealthOmics run。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRunTask。

get-run

以下代码示例演示了如何使用 get-run。

Amazon CLI

查看工作流运行

以下 get-run 示例获取有关工作流运行的详细信息。

```
aws omics get-run \
    --id 1234567
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
    "creationTime": "2022-11-30T22:58:22.615865Z",
    "digest":
 "sha256:c54bxmpl742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",
    "id": "1234567",
    "name": "cram-to-bam",
    "outputUri": "s3://omics-artifacts-01d6xmpl4e72dd32/workflow-output/",
    "parameters": {
        "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.dict",
        "ref_fasta_index": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta.fai",
        "ref_fasta": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta",
        "sample_name": "NA12878",
        "input_cram": "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram"
    },
    "resourceDigests": {
        "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.fasta.fai":
 "etag:f76371b113734a56cde236bc0372de0a",
        "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.dict":
 "etag:3884c62eb0e53fa92459ed9bff133ae6",
        "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.fasta":
 "etag:e307d81c605fb91b7720a08f00276842-388",
        "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram":
 "etag:a9f52976381286c6143b5cc681671ec6"
    },
```

```
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
    "startedBy": "arn:aws:iam::123456789012:user/laptop-2020",
    "status": "STARTING",
    "tags": {},
    "workflowId": "1234567",
    "workflowType": "PRIVATE"
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Run lifecycle in a workflow。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRun。

get-sequence-store

以下代码示例演示了如何使用 get-sequence-store。

Amazon CLI

查看序列存储

以下 get-sequence-store 示例获取有关 ID 为 1234567890 的序列存储的详细信息。

```
aws omics get-sequence-store \
    --id 1234567890
```

输出：

```
{
    "arn": "arn:aws:omics:us-east-1:123456789012:sequenceStore/1234567890",
    "creationTime": "2022-11-23T19:55:48.376Z",
    "id": "1234567890",
    "name": "my-seq-store"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSequenceStore。

get-share

以下代码示例演示了如何使用 get-share。

Amazon CLI

检索有关 HealthOmics 分析数据共享的元数据

以下 get-share 示例检索分析数据跨账户共享的元数据。

```
aws omics get-share \
    --share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

输出：

```
{
    "share": {
        "shareId": "495c21bedc889d07d0ab69d710a6841e-
dd75ab7a1a9c384fa848b5bd8e5a7e0a",
        "name": "my_Share-123",
        "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/
omics_dev_var_store",
        "principalSubscriber": "123456789012",
        "ownerId": "555555555555",
        "status": "PENDING"
    }
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的跨账户共享。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetShare。

get-variant-import-job

以下代码示例演示了如何使用 get-variant-import-job。

Amazon CLI

查看变体导入作业

以下 get-variant-import-job 示例获取有关变体导入作业的详细信息。

```
aws omics get-variant-import-job \
    --job-id edd7b8ce-xmpl-47e2-bc99-258cac95a508
```

输出：

```
{
    "creationTime": "2022-11-23T22:42:50.037812Z",
    "destinationName": "my_var_store",
    "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",
    "items": [
        {
            "jobStatus": "IN_PROGRESS",
            "source": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.known_indels.vcf.gz"
        }
    ],
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
    "runLeftNormalization": false,
    "status": "IN_PROGRESS",
    "updateTime": "2022-11-23T22:43:05.898309Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVariantImportJob。

get-variant-store

以下代码示例演示了如何使用 get-variant-store。

Amazon CLI

查看变体存储

以下 get-variant-store 示例获取有关变体存储的详细信息。

```
aws omics get-variant-store \
    --name my_var_store
```

输出：

```
{
    "creationTime": "2022-11-23T22:09:07.534499Z",
    "id": "02dexmplcfdd",
    "name": "my_var_store",
    "reference": {
```

```
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "CREATING",
    "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/my_var_store",
    "storeSizeBytes": 0,
    "tags": {},
    "updateTime": "2022-11-23T22:09:24.931711Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVariantStore。

get-workflow

以下代码示例演示了如何使用 get-workflow。

Amazon CLI

查看工作流

以下 get-workflow 示例获取有关 ID 为 1234567 的工作流的详细信息。

```
aws omics get-workflow \
    --id 1234567
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
    "creationTime": "2022-11-30T22:33:16.225368Z",
    "digest":
 "sha256:c54bxmpl742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",
    "engine": "WDL",
    "id": "1234567",
    "main": "workflow-crambam.wdl",
    "name": "cram-converter",
    "parameterTemplate": {
        "ref_dict": {
            "description": "dictionary file for 'ref_fasta'"
        },
        "ref_fasta_index": {
```

```
            "description": "Index of the reference genome fasta file"
        },
        "ref_fasta": {
            "description": "Reference genome fasta file"
        },
        "input_cram": {
            "description": "The Cram file to convert to BAM"
        },
        "sample_name": {
            "description": "The name of the input sample, used to name the output
 BAM"
        }
    },
    "status": "ACTIVE",
    "statusMessage": "workflow-crambam.wdl\n    workflow CramToBamFlow\n
 call CramToBamTask\n        call ValidateSamFile\n    task CramToBamTask\n    task
 ValidateSamFile\n",
    "tags": {},
    "type": "PRIVATE"
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating private workflows。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWorkflow。

list-annotation-import-jobs

以下代码示例演示了如何使用 list-annotation-import-jobs。

Amazon CLI

获取注释导入作业的列表

以下 list-annotation-import-jobs 获取注释导入作业的列表。

```
aws omics list-annotation-import-jobs
```

输出：

```
{
    "annotationImportJobs": [
        {
            "creationTime": "2022-11-30T01:39:41.478294Z",
```

```
            "destinationName": "gff_ann_store",
            "id": "18a9e792-xmpl-4869-a105-e5b602900444",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "runLeftNormalization": false,
            "status": "COMPLETED",
            "updateTime": "2022-11-30T01:47:09.145178Z"
        },
        {
            "creationTime": "2022-11-30T00:45:58.007838Z",
            "destinationName": "my_ann_store",
            "id": "4e9eafc8-xmpl-431e-a0b2-3bda27cb600a",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "runLeftNormalization": false,
            "status": "FAILED",
            "updateTime": "2022-11-30T00:47:01.706325Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAnnotationImportJobs。

`list-annotation-store-versions`

以下代码示例演示了如何使用 `list-annotation-store-versions`。

Amazon CLI

列出注释存储的所有版本。

以下 `list-annotation-store-versions` 示例列出了注释存储的所有版本。

```
aws omics list-annotation-store-versions \
    --name my_annotation_store
```

输出：

```
{
    "annotationStoreVersions": [
        {
```

```
        "storeId": "4934045d1c6d",
        "id": "2a3f4a44aa7b",
        "status": "CREATING",
        "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version_2",
        "name": "my_annotation_store",
        "versionName": "my_version_2",
        "creation Time": "2023-07-21T17:20:59.380043+00:00",
        "versionSizeBytes": 0
},
{
     "storeId": "4934045d1c6d",
     "id": "4934045d1c6d",
     "status": "ACTIVE",
     "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version_1",
     "name": "my_annotation_store",
     "versionName": "my_version_1",
     "creationTime": "2023-07-21T17:15:49.251040+00:00",
     "updateTime": "2023-07-21T17:15:56.434223+00:00",
     "statusMessage": "",
     "versionSizeBytes": 0
     }

}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的创建注释存储的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAnnotationStoreVersions。

list-annotation-stores

以下代码示例演示了如何使用 list-annotation-stores。

Amazon CLI

获取注释存储列表

以下 list-annotation-stores 示例获取注释存储列表。

```
aws omics list-annotation-stores
```

输出：

```
{
    "annotationStores": [
        {
            "creationTime": "2022-11-23T22:48:39.226492Z",
            "id": "0a91xmplc71f",
            "name": "my_ann_store",
            "reference": {
                "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
            },
            "status": "ACTIVE",
            "statusMessage": "",
            "storeArn": "arn:aws:omics:us-west-2:123456789012:annotationStore/
my_ann_store",
            "storeFormat": "VCF",
            "storeSizeBytes": 0,
            "updateTime": "2022-11-23T22:53:27.372840Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAnnotationStores。

list-multipart-read-set-uploads

以下代码示例演示了如何使用 list-multipart-read-set-uploads。

Amazon CLI

列出所有分段读取设置上传及其状态。

以下 list-multipart-read-set-uploads 示例列出了所有分段读取集上传及其状态。

```
aws omics list-multipart-read-set-uploads \
    --sequence-store-id 0123456789
```

输出：

```
{
"uploads":
```

```
    [
        {
           "sequenceStoreId": "0123456789",
           "uploadId": "8749584421",
           "sourceFileType": "FASTQ",
            "subjectId": "mySubject",
            "sampleId": "mySample",
            "generatedFrom": "1000 Genomes",
            "name": "HG00146",
            "description": "FASTQ for HG00146",
            "creationTime": "2023-11-29T19:22:51.349298+00:00"
        },
        {
            "sequenceStoreId": "0123456789",
            "uploadId": "5290538638",
            "sourceFileType": "BAM",
            "subjectId": "mySubject",
            "sampleId": "mySample",
            "generatedFrom": "1000 Genomes",
            "referenceArn": "arn:aws:omics:us-
west-2:845448930428:referenceStore/8168613728/reference/2190697383",
            "name": "HG00146",
            "description": "BAM for HG00146",
            "creationTime": "2023-11-29T19:23:33.116516+00:00"
        },
        {
            "sequenceStoreId": "0123456789",
            "uploadId": "4174220862",
            "sourceFileType": "BAM",
            "subjectId": "mySubject",
            "sampleId": "mySample",
            "generatedFrom": "1000 Genomes",
            "referenceArn": "arn:aws:omics:us-
west-2:845448930428:referenceStore/8168613728/reference/2190697383",
            "name": "HG00147",
            "description": "BAM for HG00147",
            "creationTime": "2023-11-29T19:23:47.007866+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMultipartReadSetUploads。

list-read-set-activation-jobs

以下代码示例演示了如何使用 list-read-set-activation-jobs。

Amazon CLI

获取读取集激活作业列表

以下 list-read-set-activation-jobs 示例获取 ID 为 1234567890 的序列存储的激活作业列表。

```
aws omics list-read-set-activation-jobs \
    --sequence-store-id 1234567890
```

输出：

```
{
    "activationJobs": [
        {
            "completionTime": "2022-12-06T22:33:42.828Z",
            "creationTime": "2022-12-06T22:32:45.213Z",
            "id": "1234567890",
            "sequenceStoreId": "1234567890",
            "status": "COMPLETED"
        },
        {
            "creationTime": "2022-12-06T22:35:10.100Z",
            "id": "1234567890",
            "sequenceStoreId": "1234567890",
            "status": "IN_PROGRESS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReadSetActivationJobs。

list-read-set-export-jobs

以下代码示例演示了如何使用 list-read-set-export-jobs。

Amazon CLI

获取读取集导出作业列表

以下 `list-read-set-export-jobs` 示例获取 ID 为 1234567890 的序列存储的导出作业列表。

```
aws omics list-read-set-export-jobs \
    --sequence-store-id 1234567890
```

输出：

```
{
    "exportJobs": [
        {
            "completionTime": "2022-12-06T22:39:14.491Z",
            "creationTime": "2022-12-06T22:37:18.612Z",
            "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",
            "id": "1234567890",
            "sequenceStoreId": "1234567890",
            "status": "COMPLETED"
        },
        {
            "creationTime": "2022-12-06T22:38:04.871Z",
            "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",
            "id": "1234567890",
            "sequenceStoreId": "1234567890",
            "status": "IN_PROGRESS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReadSetExportJobs。

list-read-set-import-jobs

以下代码示例演示了如何使用 `list-read-set-import-jobs`。

Amazon CLI

获取读取集导入作业列表

以下 list-read-set-import-jobs 示例获取 ID 为 1234567890 的序列存储的导入作业列表。

```
aws omics list-read-set-import-jobs \
    --sequence-store-id 1234567890
```

输出：

```
{
    "importJobs": [
        {
            "completionTime": "2022-11-29T18:17:49.244Z",
            "creationTime": "2022-11-29T17:32:47.700Z",
            "id": "1234567890",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "sequenceStoreId": "1234567890",
            "status": "COMPLETED"
        },
        {
            "completionTime": "2022-11-23T22:01:34.090Z",
            "creationTime": "2022-11-23T21:52:43.289Z",
            "id": "1234567890",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "sequenceStoreId": "1234567890",
            "status": "COMPLETED_WITH_FAILURES"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReadSetImportJobs。

list-read-set-upload-parts

以下代码示例演示了如何使用 list-read-set-upload-parts。

Amazon CLI

列出序列存储请求的分段上传中的所有分段。

以下 `list-read-set-upload-parts` 示例列出了序列存储请求的分段上传中的所有分段。

```
aws omics list-read-set-upload-parts \
    --sequence-store-id 0123456789 \
    --upload-id 1122334455 \
    --part-source SOURCE1
```

输出：

```
{
    "parts": [
        {
            "partNumber": 1,
            "partSize": 94371840,
            "file": "SOURCE1",
            "checksum":
 "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635",
            "lastUpdatedTime": "2023-02-02T20:14:47.533000+00:00"
        }
        {
            "partNumber": 2,
            "partSize": 10471840,
            "file": "SOURCE1",
            "checksum":
 "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635",
            "lastUpdatedTime": "2023-02-02T20:14:47.533000+00:00"
        }
      ]

}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReadSetUploadParts。

list-read-sets

以下代码示例演示了如何使用 `list-read-sets`。

Amazon CLI

获取读取集列表

以下 list-read-sets 示例获取 ID 为 1234567890 的序列存储的读取集列表。

```
aws omics list-read-sets \
    --sequence-store-id 1234567890
```

输出：

```
{
    "readSets": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890/
readSet/1234567890",
            "creationTime": "2022-11-23T21:55:00.515Z",
            "fileType": "FASTQ",
            "id": "1234567890",
            "name": "HG00146",
            "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
            "sampleId": "fastq-sample",
            "sequenceStoreId": "1234567890",
            "status": "ACTIVE",
            "subjectId": "fastq-subject"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReadSets。

list-reference-import-jobs

以下代码示例演示了如何使用 list-reference-import-jobs。

Amazon CLI

获取参考导入作业列表

以下 list-reference-import-jobs 示例获取 ID 为 1234567890 的参考存储的参考导入作业列表。

```
aws omics list-reference-import-jobs \
```

```
    --reference-store-id 1234567890
```

输出：

```
{
    "importJobs": [
        {
            "completionTime": "2022-11-23T19:54:58.204Z",
            "creationTime": "2022-11-23T19:53:20.729Z",
            "id": "1234567890",
            "referenceStoreId": "1234567890",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "status": "COMPLETED"
        },
        {
            "creationTime": "2022-11-23T20:34:03.250Z",
            "id": "1234567890",
            "referenceStoreId": "1234567890",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "status": "IN_PROGRESS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReferenceImportJobs。

list-reference-stores

以下代码示例演示了如何使用 list-reference-stores。

Amazon CLI

获取参考存储列表

以下 list-reference-stores 示例获取参考存储列表。

```
aws omics list-reference-stores
```

输出：

```
{
    "referenceStores": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",
            "creationTime": "2022-11-22T22:13:25.947Z",
            "id": "1234567890",
            "name": "my-ref-store"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReferenceStores。

`list-references`

以下代码示例演示了如何使用 `list-references`。

Amazon CLI

获取参考列表

以下 `list-references` 示例获取 ID 为 1234567890 的参考存储的基因组参考列表。

```
aws omics list-references \
    --reference-store-id 1234567890
```

输出：

```
{
    "references": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
            "creationTime": "2022-11-22T22:27:09.033Z",
            "id": "1234567890",
            "md5": "7ff134953dcca8c8997453bbb80b6b5e",
            "name": "assembly-38",
            "referenceStoreId": "1234567890",
            "status": "ACTIVE",
            "updateTime": "2022-11-22T22:27:09.033Z"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReferences。

list-run-groups

以下代码示例演示了如何使用 list-run-groups。

Amazon CLI

获取运行组列表

以下 list-run-groups 示例获取运行组列表。

```
aws omics list-run-groups
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
            "creationTime": "2022-12-01T00:58:42.915219Z",
            "id": "1234567",
            "maxCpus": 20,
            "maxDuration": 600,
            "name": "cram-convert"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating run groups。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRunGroups。

list-run-tasks

以下代码示例演示了如何使用 list-run-tasks。

Amazon CLI

获取任务列表

以下 `list-run-tasks` 示例获取工作流运行的任务列表。

```
aws omics list-run-tasks \
    --id 1234567
```

输出：

```
{
    "items": [
        {
            "cpus": 1,
            "creationTime": "2022-11-30T23:13:00.718651Z",
            "memory": 15,
            "name": "CramToBamTask",
            "startTime": "2022-11-30T23:17:47.016Z",
            "status": "COMPLETED",
            "stopTime": "2022-11-30T23:18:21.503Z",
            "taskId": "1234567"
        },
        {
            "cpus": 1,
            "creationTime": "2022-11-30T23:18:32.315606Z",
            "memory": 4,
            "name": "ValidateSamFile",
            "startTime": "2022-11-30T23:23:40.165Z",
            "status": "COMPLETED",
            "stopTime": "2022-11-30T23:24:14.766Z",
            "taskId": "1234567"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Task lifecycle in a HealthOmics run。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRunTasks。

list-runs

以下代码示例演示了如何使用 list-runs。

Amazon CLI

获取工作流运行列表

以下 list-runs 示例获取工作流运行列表。

```
aws omics list-runs
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
            "creationTime": "2022-12-02T23:20:01.202074Z",
            "id": "1234567",
            "name": "cram-to-bam",
            "priority": 1,
            "startTime": "2022-12-02T23:29:18.115Z",
            "status": "COMPLETED",
            "stopTime": "2022-12-02T23:57:54.428812Z",
            "storageCapacity": 10,
            "workflowId": "1234567"
        },
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
            "creationTime": "2022-12-03T00:16:57.180066Z",
            "id": "1234567",
            "name": "cram-to-bam",
            "priority": 1,
            "startTime": "2022-12-03T00:26:50.233Z",
            "status": "FAILED",
            "stopTime": "2022-12-03T00:37:21.451340Z",
            "storageCapacity": 10,
            "workflowId": "1234567"
        },
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
            "creationTime": "2022-12-05T17:57:08.444817Z",
            "id": "1234567",
```

```
            "name": "cram-to-bam",
            "status": "STARTING",
            "workflowId": "1234567"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Run lifecycle in a workflow。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRuns。

list-sequence-stores

以下代码示例演示了如何使用 list-sequence-stores。

Amazon CLI

获取序列存储列表

以下 list-sequence-stores 示例获取序列存储列表。

```
aws omics list-sequence-stores
```

输出：

```
{
    "sequenceStores": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890",
            "creationTime": "2022-11-23T01:24:33.629Z",
            "id": "1234567890",
            "name": "my-seq-store"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSequenceStores。

list-shares

以下代码示例演示了如何使用 list-shares。

Amazon CLI

列出 HealthOmics 分析数据的可用共享

以下 `list-shares` 示例列出了为资源所有者创建的所有共享。

```
aws omics list-shares \
    --resource-owner SELF
```

输出：

```
{
    "shares": [
        {
            "shareId": "595c1cbd-a008-4eca-a887-954d30c91c6e",
            "name": "myShare",
            "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/
store_1",
            "principalSubscriber": "123456789012",
            "ownerId": "555555555555",
            "status": "PENDING"
        }
        {
            "shareId": "39b65d0d-4368-4a19-9814-b0e31d73c10a",
            "name": "myShare3456",
            "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/
store_2",
            "principalSubscriber": "123456789012",
            "ownerId": "555555555555",
            "status": "ACTIVE"
        },
        {
            "shareId": "203152f5-eef9-459d-a4e0-a691668d44ef",
            "name": "myShare4",
            "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/
store_3",
            "principalSubscriber": "123456789012",
            "ownerId": "555555555555",
            "status": "ACTIVE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的跨账户共享。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListShares。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

获取标签列表

以下 list-tags-for-resource 示例获取 ID 为 1234567 的工作流运行的标签列表。

```
aws omics list-tags-for-resource \
    --resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567
```

输出：

```
{
    "tags": {
        "department": "analytics"
    }
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的在 Amazon Omics 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-variant-import-jobs

以下代码示例演示了如何使用 list-variant-import-jobs。

Amazon CLI

获取变体导入作业列表

以下 list-variant-import-jobs 示例获取变体导入作业的列表。

```
aws omics list-variant-import-jobs
```

输出：

```
{
    "variantImportJobs": [
        {
            "creationTime": "2022-11-23T22:47:02.514002Z",
            "destinationName": "my_var_store",
            "id": "69cb65d6-xmpl-4a4a-9025-4565794b684e",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "runLeftNormalization": false,
            "status": "COMPLETED",
            "updateTime": "2022-11-23T22:49:17.976597Z"
        },
        {
            "creationTime": "2022-11-23T22:42:50.037812Z",
            "destinationName": "my_var_store",
            "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",
            "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W8O1XMPL7QZ",
            "runLeftNormalization": false,
            "status": "COMPLETED",
            "updateTime": "2022-11-23T22:45:26.009880Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVariantImportJobs。

list-variant-stores

以下代码示例演示了如何使用 `list-variant-stores`。

Amazon CLI

获取变体存储列表

以下 `list-variant-stores` 示例获取变体存储列表。

```
aws omics list-variant-stores
```

输出：

```
{
    "variantStores": [
        {
            "creationTime": "2022-11-23T22:09:07.534499Z",
            "id": "02dexmplcfdd",
            "name": "my_var_store",
            "reference": {
                "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
            },
            "status": "CREATING",
            "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/
my_var_store",
            "storeSizeBytes": 0,
            "updateTime": "2022-11-23T22:09:24.931711Z"
        },
        {
            "creationTime": "2022-09-23T23:00:09.140265Z",
            "id": "8777xmpl1a24",
            "name": "myvstore0",
            "status": "ACTIVE",
            "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/
myvstore0",
            "storeSizeBytes": 0,
            "updateTime": "2022-09-23T23:03:26.013220Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVariantStores。

`list-workflows`

以下代码示例演示了如何使用 `list-workflows`。

Amazon CLI

获取工作流列表

以下 `list-workflows` 示例获取工作流列表。

```
aws omics list-workflows
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
            "creationTime": "2022-09-23T23:08:22.041227Z",
            "digest": "nSCNo/qMWFxmplXpUdokXJnwgneOaxyyc2YOxVxrJTE=",
            "id": "1234567",
            "name": "my-wkflow-0",
            "status": "ACTIVE",
            "type": "PRIVATE"
        },
        {
            "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
            "creationTime": "2022-11-30T22:33:16.225368Z",
            "digest":
 "sha256:c54bxmpl742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",
            "id": "1234567",
            "name": "cram-converter",
            "status": "ACTIVE",
            "type": "PRIVATE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating private workflows。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWorkflows。

start-annotation-import-job

以下代码示例演示了如何使用 start-annotation-import-job。

Amazon CLI

导入注释

以下 start-annotation-import-job 示例从 Amazon S3 导入注释。

```
aws omics start-annotation-import-job \
```

```
    --destination-name tsv_ann_store \
    --no-run-left-normalization \
    --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ \
    --items source=s3://omics-artifacts-01d6xmpl4e72dd32/targetedregions.bed.gz
```

输出：

```
{
    "jobId": "984162c7-xmpl-4d23-ab47-286f7950bfbf"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartAnnotationImportJob。

start-read-set-activation-job

以下代码示例演示了如何使用 start-read-set-activation-job。

Amazon CLI

激活已存档的读取集

以下 start-read-set-activation-job 示例激活了两个读取集。

```
aws omics start-read-set-activation-job \
    --sequence-store-id 1234567890 \
    --sources readSetId=1234567890 readSetId=1234567890
```

输出：

```
{
    "creationTime": "2022-12-06T22:35:10.100Z",
    "id": "1234567890",
    "sequenceStoreId": "1234567890",
    "status": "SUBMITTED"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReadSetActivationJob。

start-read-set-export-job

以下代码示例演示了如何使用 start-read-set-export-job。

Amazon CLI

导出读取集

以下 start-read-set-export-job 示例将两个读取集导出到 Amazon S3。

```
   aws omics start-read-set-export-job \
       --sequence-store-id 1234567890 \
       --sources readSetId=1234567890 readSetId=1234567890 \
       --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ
\
       --destination s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/
```

输出：

```
{
    "creationTime": "2022-12-06T22:37:18.612Z",
    "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",
    "id": "1234567890",
    "sequenceStoreId": "1234567890",
    "status": "SUBMITTED"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReadSetExportJob。

start-read-set-import-job

以下代码示例演示了如何使用 start-read-set-import-job。

Amazon CLI

导入读取集

以下 start-read-set-import-job 示例导入读取集。

```
aws omics start-read-set-import-job \
    --sequence-store-id 1234567890 \
    --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ \
    --sources file://readset-sources.json
```

readset-sources.json 是一个 JSON 文档，内容如下。

```
[
    {
        "sourceFiles":
        {
            "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00100.chrom20.ILLUMINA.bwa.GBR.low_coverage.20101123.bam"
        },
        "sourceFileType": "BAM",
        "subjectId": "bam-subject",
        "sampleId": "bam-sample",
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
        "name": "HG00100"
    }
]
```

输出：

```
{
    "creationTime": "2022-11-23T01:36:38.158Z",
    "id": "1234567890",
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W8O1XMPL7QZ",
    "sequenceStoreId": "1234567890",
    "status": "SUBMITTED"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReadSetImportJob。

start-reference-import-job

以下代码示例演示了如何使用 start-reference-import-job。

Amazon CLI

导入参考基因组

以下 start-reference-import-job 示例从 Amazon S3 导入参考基因组。

```
aws omics start-reference-import-job \
    --reference-store-id 1234567890 \
    --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-W8O1XMPL7QZ \
    --sources sourceFile=s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.fasta,name=assembly-38
```

输出：

```
{
    "creationTime": "2022-11-22T22:25:41.124Z",
    "id": "1234567890",
    "referenceStoreId": "1234567890",
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-W8O1XMPL7QZ",
    "status": "SUBMITTED"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartReferenceImportJob。

start-run

以下代码示例演示了如何使用 start-run。

Amazon CLI

运行工作流

以下 start-run 示例运行 ID 为 1234567 的工作流。

```
aws omics start-run \
    --workflow-id 1234567 \
    --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-W8O1XMPL7QZ \
```

```
    --name 'cram-to-bam' \
    --output-uri s3://omics-artifacts-01d6xmpl4e72dd32/workflow-output/ \
    --run-group-id 1234567 \
    --priority 1 \
    --storage-capacity 10 \
    --log-level ALL \
    --parameters file://workflow-inputs.json
```

workflow-inputs.json 是一个 JSON 文档，内容如下。

```
{
    "sample_name": "NA12878",
    "input_cram": "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram",
    "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.dict",
    "ref_fasta": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta",
    "ref_fasta_index": "omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta.fai"
}
```

输出：

```
{
    "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
    "id": "1234567",
    "status": "PENDING",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Starting a run。

从 Amazon Omics 加载源文件

您还可以使用特定于服务的 URI 从 Amazon Omics 存储服务加载源文件。以下示例 workflow-inputs.json 文件使用 Amazon Omics URI 作为读取集和参考基因组源。

```
{
    "sample_name": "NA12878",
    "input_cram": "omics://123456789012.storage.us-west-2.amazonaws.com/1234567890/
readSet/1234567890/source1",
```

```
    "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.dict",
    "ref_fasta": "omics://123456789012.storage.us-west-2.amazonaws.com/1234567890/
reference/1234567890",
    "ref_fasta_index": "omics://123456789012.storage.us-
west-2.amazonaws.com/1234567890/reference/1234567890/index"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartRun。

start-variant-import-job

以下代码示例演示了如何使用 start-variant-import-job。

Amazon CLI

导入变体文件

以下 start-variant-import-job 示例导入 VCF 格式的变体文件。

```
aws omics start-variant-import-job \
    --destination-name my_var_store \
    --no-run-left-normalization  \
    --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ \
    --items source=s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.known_indels.vcf.gz
```

输出：

```
{
    "jobId": "edd7b8ce-xmpl-47e2-bc99-258cac95a508"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartVariantImportJob。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例向 ID 为 1234567 的工作流添加了 department 标签。

```
aws omics tag-resource \
    --resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567 \
    --tags department=analytics
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的在 Amazon Omics 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从工作流中删除了 department 标签。

```
aws omics untag-resource \
    --resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567 \
    --tag-keys department
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学存储功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-annotation-store

以下代码示例演示了如何使用 update-annotation-store。

Amazon CLI

更新注释存储

以下 update-annotation-store 示例更新了名为 my_vcf_store 的注释存储的描述。

```
aws omics update-annotation-store \
    --name my_vcf_store \
    --description "VCF annotation store"
```

输出：

```
{
    "creationTime": "2022-12-05T18:00:56.101860Z",
    "description": "VCF annotation store",
    "id": "bd6axmpl2444",
    "name": "my_vcf_store",
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "ACTIVE",
    "storeFormat": "VCF",
    "updateTime": "2022-12-05T18:13:16.100051Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAnnotationStore。

update-run-group

以下代码示例演示了如何使用 update-run-group。

Amazon CLI

更新运行组

以下 update-run-group 示例更新了 ID 为 1234567 的运行组的设置。

```
aws omics update-run-group \
    --id 1234567 \
    --max-cpus 10
```

输出：

```
{
```

```
    "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
    "creationTime": "2022-12-01T00:58:42.915219Z",
    "id": "1234567",
    "maxCpus": 10,
    "maxDuration": 600,
    "name": "cram-convert",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学流程预置扩缩功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRunGroup。

update-variant-store

以下代码示例演示了如何使用 update-variant-store。

Amazon CLI

更新变体存储

以下 update-variant-store 示例更新了名为 my_var_store 的变体存储的描述。

```
aws omics update-variant-store \
    --name my_var_store \
    --description "variant store"
```

输出：

```
{
    "creationTime": "2022-11-23T22:09:07.534499Z",
    "description": "variant store",
    "id": "02dexmplcfdd",
    "name": "my_var_store",
    "reference": {
        "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
    },
    "status": "ACTIVE",
    "updateTime": "2022-12-05T18:23:37.686402Z"
}
```

有关更多信息，请参阅《Amazon Omics 开发人员指南》中的组学分析功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVariantStore。

update-workflow

以下代码示例演示了如何使用 update-workflow。

Amazon CLI

更新工作流

以下 update-workflow 示例更新了 ID 为 1234567 的工作流的描述。

```
aws omics update-workflow \
    --id 1234567 \
    --description "copy workflow"
```

有关更多信息，请参阅《Amazon HealthOmics User Guide》中的 Creating or updating a workflow。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWorkflow。

upload-read-set-part

以下代码示例演示了如何使用 upload-read-set-part。

Amazon CLI

上传读取集部分。

以下 upload-read-set-part 示例上传了读取集的指定部分。

```
aws omics upload-read-set-part \
    --sequence-store-id 0123456789 \
    --upload-id 1122334455 \
    --part-source SOURCE1 \
    --part-number 1 \
    --payload /path/to/file/read_1_part_1.fastq.gz
```

输出：

```
{
    "checksum": "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635"
}
```

有关更多信息，请参阅《Amazon HealthOmics 用户指南》中的直接上传到序列存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadReadSetPart。

使用 Amazon CLI 的 IAM 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 IAM 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-client-id-to-open-id-connect-provider`

以下代码示例演示了如何使用 `add-client-id-to-open-id-connect-provider`。

Amazon CLI

将客户端 ID（受众）添加到 Open-ID Connect（OIDC）提供者

以下 `add-client-id-to-open-id-connect-provider` 命令将客户端 ID `my-application-ID` 添加到名为 `server.example.com` 的 OIDC 提供者。

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

此命令不生成任何输出。

要创建 OIDC 提供者，请使用 `create-open-id-connect-provider` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddClientIdToOpenIdConnectProvider。

add-role-to-instance-profile

以下代码示例演示了如何使用 `add-role-to-instance-profile`。

Amazon CLI

为实例配置文件添加角色

以下 `add-role-to-instance-profile` 命令可将名为 `S3Access` 的角色添加到名为 `Webserver` 的实例配置文件。

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

此命令不生成任何输出。

要创建实例配置文件，请使用 `create-instance-profile` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM 角色为 Amazon EC2 实例上运行的应用程序授予权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddRoleToInstanceProfile。

add-user-to-group

以下代码示例演示了如何使用 `add-user-to-group`。

Amazon CLI

将用户添加到 IAM 组

以下 add-user-to-group 命令可将名为 Bob 的 IAM 用户添加到名为 Admins 的 IAM 组。

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 用户组中添加和删除用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddUserToGroup。

attach-group-policy

以下代码示例演示了如何使用 attach-group-policy。

Amazon CLI

将托管策略附加到 IAM 组

以下 attach-group-policy 命令可将名为 ReadOnlyAccess 的 Amazon 托管策略附加到名为 Finance 的 IAM 组。

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的托管策略和内联策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachGroupPolicy。

attach-role-policy

以下代码示例演示了如何使用 attach-role-policy。

Amazon CLI

将托管策略附加到 IAM 角色

以下 attach-role-policy 命令可将名为 ReadOnlyAccess 的 Amazon 托管策略附加到名为 ReadOnlyRole 的 IAM 角色。

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的托管策略和内联策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachRolePolicy。

attach-user-policy

以下代码示例演示了如何使用 attach-user-policy。

Amazon CLI

将托管策略附加到 IAM 用户

以下 attach-user-policy 命令可将名为 AdministratorAccess 的 Amazon 托管策略附加到名为 Alice 的 IAM 用户。

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的托管策略和内联策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachUserPolicy。

change-password

以下代码示例演示了如何使用 change-password。

Amazon CLI

更改 IAM 用户的密码

要更改 IAM 用户的密码，建议使用 `--cli-input-json` 参数传递包含新旧密码的 JSON 文件。通过此方法，您可以使用含有非字母数字字符的强密码。将密码作为命令行参数传递时，可能难以使用含有非字母数字字符的密码。要使用 `--cli-input-json` 参数，请先使用带 `--generate-cli-skeleton` 参数的 `change-password` 命令，如以下示例所示。

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

前述命令创建一个名为 change-password.json 的 JSON 文件，您可以用该文件来填写旧密码和新密码。例如，该文件可能如下所示。

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

接下来，要更改密码，请再次使用 `change-password` 命令，这次是传递 `--cli-input-json` 参数以指定您的 JSON 文件。以下 `change-password` 命令将 `--cli-input-json` 参数与名为 change-password.json 的 JSON 文件一起使用。

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

此命令不生成任何输出。

此命令只能由 IAM 用户调用。如果使用 Amazon 账户（根）凭证调用此命令，则命令将返回 `InvalidUserType` 错误。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 用户如何更改自己的密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangePassword。

`create-access-key`

以下代码示例演示了如何使用 `create-access-key`。

Amazon CLI

为 IAM 用户创建访问密钥

以下 create-access-key 命令创建名为 Bob IAM 用户的访问密钥（访问密钥 ID 和秘密访问密钥）。

```
aws iam create-access-key \
    --user-name Bob
```

输出：

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

将秘密访问密钥存储在安全位置。如果访问密钥丢失，将无法恢复，必须创建一个新的访问密钥。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccessKey。

create-account-alias

以下代码示例演示了如何使用 create-account-alias。

Amazon CLI

创建账户别名

以下 create-account-alias 命令可为 Amazon 账户创建别名 examplecorp。

```
aws iam create-account-alias \
    --account-alias examplecorp
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的您的 Amazon 账户 ID 及其别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccountAlias。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

创建 IAM 组

以下 create-group 命令创建名为 Admins 的 IAM 组。

```
aws iam create-group \
    --group-name Admins
```

输出：

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-instance-profile

以下代码示例演示了如何使用 create-instance-profile。

Amazon CLI

创建实例配置文件

以下 create-instance-profile 命令创建名为 Webserver 的实例配置文件。

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

输出：

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

要将角色添加到实例配置文件，请使用 `add-role-to-instance-profile` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM 角色为 Amazon EC2 实例上运行的应用程序授予权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstanceProfile。

`create-login-profile`

以下代码示例演示了如何使用 `create-login-profile`。

Amazon CLI

为 IAM 用户创建密码

要为 IAM 用户创建密码，建议使用 `--cli-input-json` 参数传递包含密码的 JSON 文件。通过此方法，您可以创建含有非字母数字字符的强密码。将密码作为命令行参数传递时，可能难以创建含有非字母数字字符的密码。

要使用 `--cli-input-json` 参数，请先使用带 `--generate-cli-skeleton` 参数的 `create-login-profile` 命令，如以下示例所示。

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

前述命令创建了一个名为 create-login-profile.json 的 JSON 文件，您可以用该文件来填写后续 `create-login-profile` 命令的信息。例如：

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

接下来，要为 IAM 用户创建密码，请再次使用 `create-login-profile` 命令，这次是传递 `--cli-input-json` 参数以指定您的 JSON 文件。以下 `create-login-profile` 命令将 `--cli-input-json` 参数与名为 create-login-profile.json 的 JSON 文件一起使用。

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

输出：

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

如果新密码违反了账户密码策略，则该命令将返回 `PasswordPolicyViolation` 错误。

要更改已有密码用户的密码，请使用 `update-login-profile`。要为账户设置密码策略，请使用 `update-account-password-policy` 命令。

如果账户密码策略允许，则 IAM 用户可以使用 `change-password` 命令更改自己的密码。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoginProfile。

`create-open-id-connect-provider`

以下代码示例演示了如何使用 `create-open-id-connect-provider`。

Amazon CLI

创建 OpenID Connect（OIDC）提供者

要创建 OpenID Connect（OIDC）提供者，建议使用 `--cli-input-json` 参数来传递包含所需参数的 JSON 文件。创建 OIDC 提供者时，必须传递提供者的 URL，而且 URL 必须以 `https://` 开头。将 URL 作为命令行参数传递可能很困难，因为冒号（:）和正斜杠（/）字符在某些命令行环境中具有特殊含义。使用 `--cli-input-json` 参数可以绕过这一限制。

要使用 `--cli-input-json` 参数，请先使用带 `--generate-cli-skeleton` 参数的 `create-open-id-connect-provider` 命令，如以下示例所示。

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

前述命令创建一个名为 create-open-id-connect-provider.json 的 JSON 文件，您可以用该文件来填写后续 `create-open-id-connect-provider` 命令的信息。例如：

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

接下来，要创建 OpenID Connect（OIDC）提供者，请再次使用 `create-open-id-connect-provider` 命令，这次是传递 `--cli-input-json` 参数以指定您的 JSON 文件。以下 `create-open-id-connect-provider` 命令将 `--cli-input-json` 参数与名为 create-open-id-connect-provider.json 的 JSON 文件一起使用。

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

输出：

```
{
```

```
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

有关 OIDC 提供者的更多信息，请参阅《Amazon IAM 用户指南》中的创建 OpenID Connect（OIDC）身份提供者。

有关获取 OIDC 提供者指纹的更多信息，请参阅《Amazon IAM 用户指南》中的获取 OpenID Connect 身份提供者的指纹。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOpenIdConnectProvider。

`create-policy-version`

以下代码示例演示了如何使用 `create-policy-version`。

Amazon CLI

创建新版本的托管策略

此示例创建了一个新的 v2 版本的 IAM policy，其 ARN 为 `arn:aws:iam::123456789012:policy/MyPolicy`，并设为默认版本。

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

输出：

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM policy 版本控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicyVersion。

create-policy

以下代码示例演示了如何使用 create-policy。

Amazon CLI

示例 1：创建客户托管策略

以下命令创建名为 my-policy 的客户托管策略。policy.json 文件是当前文件夹中的一个 JSON 文档，它授予对名为 amzn-s3-demo-bucket 的 Amazon S3 存储桶中 shared 文件夹的只读权限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json
```

policy.json 的内容：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/shared/*"
            ]
        }
    ]
}
```

输出：

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
```

```
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

有关使用文件作为字符串参数输入的更多信息，请参阅《Amazon CLI 用户指南》中的为 Amazon CLI 指定参数值。

示例 2：创建带有描述的客户管理型策略

以下命令创建名为 my-policy 且带有不可变描述的客户管理型策略。

policy.json 文件是当前文件夹中的一个 JSON 文档，它授予对名为 amzn-s3-demo-bucket 的 Amazon S3 存储桶的所有 Put、List 和 Get 操作的访问权限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions for
 amzn-s3-demo-bucket"
```

policy.json 的内容：

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
```

```
}
```

输出：

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

有关基于身份的策略的更多信息，请参阅《Amazon IAM 用户指南》中的基于身份的策略和基于资源的策略。

示例 3：创建带有标签的客户管理型策略

以下命令创建名为 my-policy 且带有标签的客户托管策略。此示例使用带有以下 JSON 格式标签的 --tags 参数：'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'。或者，--tags 参数可与简写格式的标签一起使用：'Key=Department,Value=Accounting Key=Location,Value=Seattle'。

policy.json 文件是当前文件夹中的一个 JSON 文档，它授予对名为 amzn-s3-demo-bucket 的 Amazon S3 存储桶的所有 Put、List 和 Get 操作的访问权限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

policy.json 的内容：

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

输出：

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {
        ]
    }
}
```

有关标记策略的更多信息，请参阅《Amazon IAM 用户指南》中的标记客户托管策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicy。

`create-role`

以下代码示例演示了如何使用 `create-role`。

Amazon CLI

示例 1：创建 IAM 角色

以下 `create-role` 命令创建一个名为 `Test-Role` 的角色并对其附加信任策略。

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

输出：

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

信任策略在 Test-Role-Trust-Policy.json 文件中定义为 JSON 文档。（文件名和扩展名没有意义。）信任策略必须指定主体。

要将权限策略附加到角色，请使用 `put-role-policy` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

示例 2：创建具有指定最长会话持续时间的 IAM 角色

以下 `create-role` 命令创建一个名为 `Test-Role` 的角色，并将最长会话持续时间设置为 7200 秒（2 小时）。

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

输出：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色最长会话持续时间（Amazon API）。

示例 3：创建带有标签的 IAM 角色

以下命令创建带有标签的 IAM 角色。Test-Role此示例使用带有以下 JSON 格式标签的 --tags 参数标志：'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'。或者，--tags 标志可与简写格式的标签一起使用：'Key=Department,Value=Accounting Key=Location,Value=Seattle'。

```
aws iam create-role \
```

```
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

输出：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRole。

create-saml-provider

以下代码示例演示了如何使用 create-saml-provider。

Amazon CLI

创建 SAML 提供者

此示例在 IAM 中创建了一个名为 MySAMLProvider 的新 SAML 提供者。SAMLMetaData.xml 文件中的 SAML 元数据文档对其进行了描述。

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

输出：

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM SAML 身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSAMLProvider。

create-service-linked-role

以下代码示例演示了如何使用 create-service-linked-role。

Amazon CLI

创建服务相关角色

以下 create-service-linked-role 示例为指定的 Amazon 服务创建了服务相关角色，并附加了指定的描述。

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

输出：

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用服务相关角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceLinkedRole。

create-service-specific-credential

以下代码示例演示了如何使用 create-service-specific-credential。

Amazon CLI

为用户创建一组特定于服务的凭证

以下 create-service-specific-credential 示例创建一个用户名和密码，只能用它们访问已配置的服务。

```
aws iam create-service-specific-credential \
```

```
    --user-name sofia \
    --service-name codecommit.amazonaws.com
```

输出：

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-04-18T20:45:36+00:00",
        "ServiceName": "codecommit.amazonaws.com",
        "ServiceUserName": "sofia-at-123456789012",
        "ServicePassword": "k1zPZM6uVxMQ3oxqgoYlNuJPyRTZ1vREs76zTQE3eJk=",
        "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
        "UserName": "sofia",
        "Status": "Active"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceSpecificCredential。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

示例 1：创建 IAM 用户

以下 create-user 命令可在当前账户中创建一个名为 Bob 的 IAM 用户。

```
aws iam create-user \
    --user-name Bob
```

输出：

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
```

```
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 账户中创建 IAM 用户。

示例 2：在指定路径创建 IAM 用户

以下 `create-user` 命令可在指定路径创建一个名为 Bob 的 IAM 用户。

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

输出：

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 标识符。

示例 3：创建带有标签的 IAM 用户

以下 `create-user` 命令创建一个名为 Bob 且带有标签的 IAM 用户。此示例使用带有以下 JSON 格式标签的 `--tags` 参数标志：`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`。或者，`--tags` 标志可与简写格式的标签一起使用：`'Key=Department,Value=Accounting Key=Location,Value=Seattle'`。

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

输出：

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 用户。

示例 3：创建具有设定权限边界的 IAM 用户

以下 `create-user` 命令创建一个名为 Bob 且权限边界为 AmazonS3FullAccess 的 IAM 用户。

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

输出：

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
```

```
        }
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 实体的权限边界。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

create-virtual-mfa-device

以下代码示例演示了如何使用 create-virtual-mfa-device。

Amazon CLI

创建虚拟 MFA 设备

此示例创建名为 BobsMFADevice 的新虚拟 MFA 设备。它会创建一个包含引导信息的、名为 QRCode.png 的文件，并将其放在 C:/ 目录中。本示例中使用的引导方法为 QRCodePNG。

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

输出：

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 中使用多重身份验证（MFA）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVirtualMfaDevice。

deactivate-mfa-device

以下代码示例演示了如何使用 deactivate-mfa-device。

Amazon CLI

停用 MFA 设备

此命令使用与用户 Bob 关联的 ARN arn:aws:iam::210987654321:mfa/BobsMFADevice 停用虚拟 MFA 设备。

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 中使用多重身份验证（MFA）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeactivateMfaDevice。

decode-authorization-message

以下代码示例演示了如何使用 decode-authorization-message。

Amazon CLI

解码授权失败消息

以下 decode-authorization-message 示例解码在没有所需权限的情况下尝试启动实例时 EC2 控制台返回的消息。

```
aws sts decode-authorization-message \
    --encoded-message lxzA8VEjEvu-s0TTt3PgYCXik9YakOqsrFJGRZR98xNcyWAxwRq14xIvd-
npzbgTevuufCTbjeBAaDARg9cbTK1rJbg3awM33o-Vy3ebPErE2-
mWR9hVYdvX-0zKgVOWF9pWjZaJSMqxB-aLXo-I_8TTvBq88x8IFPbMArNdpu0IjxDjzf22PF3SOE3XvIQ-
_PEO0aUqHCCcsSrFtvxm6yQD1nbm6VTIVrfa0Bzy8lsoMo7SjIaJ2r5vph6SY5vCCwg6o2JKe3hIHTa8zRrDbZSFMkcX
Xx9AYAAIr6bhcis7C__bZh4dlAAWooHFGKgfoJcWGwgdzgbu9hWyVvKTpeot5hsb8qANYjJRCPXTKpi6PZfdijIkwb6g
```

输出格式为单行 JSON 文本字符串，您可以使用任何 JSON 文本处理器对其进行解析。

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":false,\"matchedStatements
\":{\"items\":[]},\"failures\":{\"items\":[]},\"context\":{\"principal
\":{\"id\":\"AIDAV3ZUEFP6J7GY7O6LO\",\"name\":\"chain-user\",\"arn\":
\"arn:aws:iam::403299380220:user/chain-user\"},\"action\":\"ec2:RunInstances\",
\"resource\":\"arn:aws:ec2:us-east-2:403299380220:instance/*\",\"conditions\":
{\"items\":[{\"key\":\"ec2:InstanceMarketType\",\"values\":{\"items\":[{\"value
\":\"on-demand\"}]}},{\"key\":\"aws:Resource\",\"values\":{\"items\":[{\"value
```

```
\":\"instance/*\"}]}},{\"key\":\"aws:Account\",\"values\":{\"items\":[{\"value
\":\"403299380220\"}]}},{\"key\":\"ec2:AvailabilityZone\",\"values\":{\"items\":
[{\"value\":\"us-east-2b\"}]}},{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items
\":[{\"value\":\"false\"}]}},{\"key\":\"ec2:IsLaunchTemplateResource\",\"values
\":{\"items\":[{\"value\":\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values
\":{\"items\":[{\"value\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",
\"values\":{\"items\":[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values
\":{\"items\":[{\"value\":\"us-east-2\"}]}},{\"key\":\"aws:Service\",\"values
\":{\"items\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":
{\"items\":[{\"value\":\"*\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items\":
[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items\":
[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-2\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":[{\"value\":
\"arn:aws:ec2:us-east-2:403299380220:instance/*\"}]}}]}}}"
}
```

有关更多消息，请参阅《Amazon re:Post》中的在 EC2 实例启动期间收到“UnauthorizedOperation”错误后，如何解码授权失败消息？

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecodeAuthorizationMessage。

delete-access-key

以下代码示例演示了如何使用 delete-access-key。

Amazon CLI

删除 IAM 用户的访问密钥

以下 delete-access-key 命令将删除名为 Bob IAM 用户的指定访问密钥（访问密钥 ID 和秘密访问密钥）。

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

此命令不生成任何输出。

要列出为 IAM 用户定义的访问密钥，请使用 list-access-keys 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessKey。

delete-account-alias

以下代码示例演示了如何使用 delete-account-alias。

Amazon CLI

删除账户别名

以下 delete-account-alias 命令将删除当前账户的别名 mycompany。

```
aws iam delete-account-alias \
    --account-alias mycompany
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的您的 Amazon 账户 ID 及其别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccountAlias。

delete-account-password-policy

以下代码示例演示了如何使用 delete-account-password-policy。

Amazon CLI

删除当前账户密码策略

以下 delete-account-password-policy 命令将删除当前账户的密码策略。

```
aws iam delete-account-password-policy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的为 IAM 用户设置账户密码策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccountPasswordPolicy。

delete-group-policy

以下代码示例演示了如何使用 delete-group-policy。

Amazon CLI

从 IAM 组中删除策略

以下 delete-group-policy 命令可将名为 ExamplePolicy 的策略从名为 Admins 的组中删除。

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

此命令不生成任何输出。

要查看附加到组的策略，请使用 list-group-policies 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM policy。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroupPolicy。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除 IAM 组

以下 delete-group 命令将删除名为 MyTestGroup 的 IAM 组。

```
aws iam delete-group \
    --group-name MyTestGroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的删除 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-instance-profile

以下代码示例演示了如何使用 delete-instance-profile。

Amazon CLI

删除实例配置文件

以下 delete-instance-profile 命令将删除名为 ExampleInstanceProfile 的实例配置文件。

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInstanceProfile。

delete-login-profile

以下代码示例演示了如何使用 delete-login-profile。

Amazon CLI

删除 IAM 用户的密码

以下 delete-login-profile 命令删除名为 Bob 的 IAM 用户的密码。

```
aws iam delete-login-profile \
    --user-name Bob
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoginProfile。

delete-open-id-connect-provider

以下代码示例演示了如何使用 delete-open-id-connect-provider。

Amazon CLI

删除 IAM OpenID Connect 身份提供者

此示例删除了连接到提供者 example.oidcprovider.com 的 IAM OIDC 提供者。

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOpenIdConnectProvider。

delete-policy-version

以下代码示例演示了如何使用 delete-policy-version。

Amazon CLI

删除托管策略的某个版本

此示例从 ARN 为 arn:aws:iam::123456789012:policy/MySamplePolicy 的策略中删除标识为 v2 的版本。

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicyVersion。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除 IAM 策略

此示例删除了 ARN 为 arn:aws:iam::123456789012:policy/MySamplePolicy 的策略。

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

delete-role-permissions-boundary

以下代码示例演示了如何使用 delete-role-permissions-boundary。

Amazon CLI

删除 IAM 角色的权限边界

以下 delete-role-permissions-boundary 示例删除指定 IAM 角色的权限边界。要对角色应用权限边界，请使用 put-role-permissions-boundary 命令。

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRolePermissionsBoundary。

delete-role-policy

以下代码示例演示了如何使用 delete-role-policy。

Amazon CLI

从 IAM 角色中移除策略

以下 delete-role-policy 命令可将名为 ExamplePolicy 的策略从名为 Test-Role 的角色中移除。

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRolePolicy。

delete-role

以下代码示例演示了如何使用 delete-role。

Amazon CLI

删除 IAM 角色

以下 delete-role 命令将删除名为 Test-Role 的角色。

```
aws iam delete-role \
    --role-name Test-Role
```

此命令不生成任何输出。

在删除角色之前，必须从所有实例配置文件中删除该角色（remove-role-from-instance-profile），分离所有托管策略（detach-role-policy），删除附加到该角色的所有内联策略（delete-role-policy）。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色和使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRole。

delete-saml-provider

以下代码示例演示了如何使用 delete-saml-provider。

Amazon CLI

删除 SAML 提供者

此示例删除了 ARN 为 arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider 的 IAM SAML 2.0 提供者。

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM SAML 身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSAMLProvider。

delete-server-certificate

以下代码示例演示了如何使用 delete-server-certificate。

Amazon CLI

从 Amazon 账户中删除服务器证书

以下 delete-server-certificate 命令可从 Amazon 账户中删除指定的服务器证书。

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

此命令不生成任何输出。

要列出 Amazon 账户中可用的服务器证书，请使用 list-server-certificates 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 中管理服务器证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServerCertificate。

delete-service-linked-role

以下代码示例演示了如何使用 delete-service-linked-role。

Amazon CLI

删除服务相关角色

以下 delete-service-linked-role 示例将删除您不再需要的指定服务相关角色。删除操作异步进行。您也可以使用 get-service-linked-role-deletion-status 命令查看删除状态并确认何时删除。

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

输出：

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用服务相关角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceLinkedRole。

delete-service-specific-credential

以下代码示例演示了如何使用 delete-service-specific-credential。

Amazon CLI

示例 1：删除请求用户的特定服务凭证

以下 delete-service-specific-credential 示例删除发出请求的用户的指定特定服务凭证。service-specific-credential-id 是您在创建凭证时提供的，您可以使用 list-service-specific-credentials 命令检索它。

```
aws iam delete-service-specific-credential \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

此命令不生成任何输出。

示例 2：删除指定用户的特定服务凭证

以下 delete-service-specific-credential 示例删除指定用户的指定特定服务凭证。service-specific-credential-id 是您在创建凭证时提供的，您可以使用 list-service-specific-credentials 命令检索它。

```
aws iam delete-service-specific-credential \
    --user-name sofia \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceSpecificCredential。

delete-signing-certificate

以下代码示例演示了如何使用 delete-signing-certificate。

Amazon CLI

删除 IAM 用户的签名证书

以下 delete-signing-certificate 命令删除名为 Bob 的 IAM 用户的指定签名证书。

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

此命令不生成任何输出。

要获取签名证书的 ID，请使用 list-signing-certificates 命令。

有关更多信息，请参阅《Amazon EC2 用户指南》中的管理签名证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSigningCertificate。

delete-ssh-public-key

以下代码示例演示了如何使用 delete-ssh-public-key。

Amazon CLI

删除附加到 IAM 用户的 SSH 公钥

以下 delete-ssh-public-key 命令删除附加到 IAM 用户 sofia 的指定 SSH 公钥。

```
aws iam delete-ssh-public-key \
    --user-name sofia \
    --ssh-public-key-id APKA123456789EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的将 SSH 密钥和 SSH 用于 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSshPublicKey。

delete-user-permissions-boundary

以下代码示例演示了如何使用 delete-user-permissions-boundary。

Amazon CLI

删除 IAM 用户的权限边界

以下 delete-user-permissions-boundary 示例删除了附加到名为 intern 的 IAM 用户的权限边界。要对用户应用权限边界，请使用 put-user-permissions-boundary 命令。

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserPermissionsBoundary。

delete-user-policy

以下代码示例演示了如何使用 delete-user-policy。

Amazon CLI

从 IAM 用户中移除策略

以下 delete-user-policy 命令可将指定策略从名为 Bob 的 IAM 用户中移除。

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

此命令不生成任何输出。

要获取 IAM 用户的策略列表，请使用 `list-user-policies` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 账户中创建 IAM 用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUserPolicy。

delete-user

以下代码示例演示了如何使用 `delete-user`。

Amazon CLI

删除 IAM 用户

以下 `delete-user` 命令可将名为 Bob 的 IAM 用户从当前账户中删除。

```
aws iam delete-user \
    --user-name Bob
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的删除 IAM 用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

delete-virtual-mfa-device

以下代码示例演示了如何使用 `delete-virtual-mfa-device`。

Amazon CLI

删除虚拟 MFA 设备

以下 `delete-virtual-mfa-device` 命令从当前账户中删除指定的 MFA 设备。

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的停用 MFA 设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVirtualMfaDevice。

detach-group-policy

以下代码示例演示了如何使用 detach-group-policy。

Amazon CLI

从组中分离策略

此示例将从名为 Testers 的组重删除 ARN 为 arn:aws:iam::123456789012:policy/TesterAccessPolicy 的托管策略。

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachGroupPolicy。

detach-role-policy

以下代码示例演示了如何使用 detach-role-policy。

Amazon CLI

从角色分离策略

此示例将从名为 FedTesterRole 的角色删除具有 ARN arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy 的托管策略。

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachRolePolicy。

detach-user-policy

以下代码示例演示了如何使用 detach-user-policy。

Amazon CLI

从用户分离策略

此示例将从用户 Bob 删除具有 ARN arn:aws:iam::123456789012:policy/TesterPolicy 的托管策略。

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的更改 IAM 用户的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachUserPolicy。

disable-organizations-root-credentials-management

以下代码示例演示了如何使用 disable-organizations-root-credentials-management。

Amazon CLI

在组织中禁用 RootCredentialsManagement 功能

以下 disable-organizations-root-credentials-management 命令禁止在组织中跨成员账户管理特权根用户凭证。

```
aws iam disable-organizations-root-credentials-management
```

输出：

```
{
    "EnabledFeatures": [
        "RootSessions"
    ]
    "OrganizationId": "o-aa111bb222"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的集中成员账户的根访问。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableOrganizationsRootCredentialsManagement。

disable-organizations-root-sessions

以下代码示例演示了如何使用 disable-organizations-root-sessions。

Amazon CLI

在组织中禁用 RootSessions 功能

以下 disable-organizations-root-sessions 命令在组织中禁用跨成员账户的特权任务的根用户会话。

```
aws iam disable-organizations-root-sessions
```

输出：

```
{
    "EnabledFeatures": [
        "RootCredentialsManagement"
    ]
    "OrganizationId": "o-aa111bb222"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的集中成员账户的根访问。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisableOrganizationsRootSessions。

enable-mfa-device

以下代码示例演示了如何使用 enable-mfa-device。

Amazon CLI

启用 MFA 设备

使用 create-virtual-mfa-device 命令创建新的虚拟 MFA 设备后，您可以将 MFA 设备分配给用户。以下 enable-mfa-device 示例将序列号为 arn:aws:iam::210987654321:mfa/BobsMFADevice 的 MFA 设备分配给用户 Bob。该命令还通过按顺序包含虚拟 MFA 设备中的前两个代码，将设备与 Amazon 同步。

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的启用虚拟多重身份验证（MFA）设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableMfaDevice。

enable-organizations-root-credentials-management

以下代码示例演示了如何使用 enable-organizations-root-credentials-management。

Amazon CLI

在组织中启用 RootCredentialsManagement 功能

以下 enable-organizations-root-credentials-management 命令允许在组织中跨成员账户管理特权根用户凭证。

```
aws iam enable-organizations-root-credentials-management
```

输出：

```
{
    "EnabledFeatures": [
        "RootCredentialsManagement"
    ]
    "OrganizationId": "o-aa111bb222"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的集中成员账户的根访问。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 EnableOrganizationsRootCredentialsManagement。

enable-organizations-root-sessions

以下代码示例演示了如何使用 enable-organizations-root-sessions。

Amazon CLI

在组织中启用 RootSessions 功能

以下 enable-organizations-root-sessions 命令允许管理账户或委托管理员对组织中的成员账户执行特权任务。

```
aws iam enable-organizations-root-sessions
```

输出：

```
{
    "EnabledFeatures": [
        "RootSessions"
    ]
    "OrganizationId": "o-aa111bb222"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的集中成员账户的根访问。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 EnableOrganizationsRootSessions。

generate-credential-report

以下代码示例演示了如何使用 generate-credential-report。

Amazon CLI

生成凭证报告

以下示例尝试为 Amazon 账户生成凭证报告。

```
aws iam generate-credential-report
```

输出：

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的获取 Amazon 账户的凭证报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateCredentialReport。

generate-organizations-access-report

以下代码示例演示了如何使用 generate-organizations-access-report。

Amazon CLI

示例 1：为组织中的根账户生成访问报告

以下 generate-organizations-access-report 示例启动一个后台作业来为组织中的指定根账户创建访问报告。创建报告后，您可以通过运行 get-organizations-access-report 命令来显示该报告。

```
aws iam generate-organizations-access-report \
    --entity-path o-4fxmplt198/r-c3xb
```

输出：

```
{
    "JobId": "a8b6c06f-aaa4-8xmp-28bc-81da71836359"
```

```
}
```

示例 2：为组织中的账户生成访问报告

以下 generate-organizations-access-report 示例启动一个后台作业来为组织 o-4fxmplt198 中的账户 ID 123456789012 创建访问报告。创建报告后，您可以通过运行 get-organizations-access-report 命令来显示该报告。

```
aws iam generate-organizations-access-report \
    --entity-path o-4fxmplt198/r-c3xb/123456789012
```

输出：

```
{
    "JobId": "14b6c071-75f6-2xmp-fb77-faf6fb4201d2"
}
```

示例 3：为组织中某个组织单位的某个账户生成访问报告

以下 generate-organizations-access-report 示例启动一个后台作业来为组织 o-4fxmplt198 的组织单位 ou-c3xb-lmu7j2yg 中的账户 ID 234567890123 创建访问报告。创建报告后，您可以通过运行 get-organizations-access-report 命令来显示该报告。

```
aws iam generate-organizations-access-report \
    --entity-path o-4fxmplt198/r-c3xb/ou-c3xb-lmu7j2yg/234567890123
```

输出：

```
{
    "JobId": "2eb6c2e6-0xmp-ec04-1425-c937916a64af"
}
```

要获取有关组织中根账户和组织单位的详细信息，请使用 organizations list-roots 和 organizations list-organizational-units-for-parent 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用上次访问的信息优化 Amazon 中的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateOrganizationsAccessReport。

generate-service-last-accessed-details

以下代码示例演示了如何使用 generate-service-last-accessed-details。

Amazon CLI

示例 1：为自定义策略生成服务访问报告

以下 generate-service-last-accessed-details 示例启动后台作业以生成一份报告，其中列出使用名为 intern-boundary 的自定义策略的 IAM 用户和其他实体所访问的服务。创建报告后，您可以通过运行 get-service-last-accessed-details 命令来显示该报告。

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

输出：

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

示例 2：为 Amazon 托管 AdministratorAccess 策略生成服务访问报告

以下 generate-service-last-accessed-details 示例启动后台作业以生成一份报告，其中列出使用 Amazon 托管 AdministratorAccess 策略的 IAM 用户和其他实体所访问的服务。创建报告后，您可以通过运行 get-service-last-accessed-details 命令来显示该报告。

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

输出：

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用上次访问的信息优化 Amazon 中的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateServiceLastAccessedDetails。

get-access-key-last-used

以下代码示例演示了如何使用 get-access-key-last-used。

Amazon CLI

检索上次使用指定访问密钥的时间信息

以下示例将检索上次使用访问密钥 ABCDEXAMPLE 的时间信息。

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

输出：

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessKeyLastUsed。

get-account-authorization-details

以下代码示例演示了如何使用 get-account-authorization-details。

Amazon CLI

列出 Amazon 账户 IAM 用户、组、角色和策略

以下 get-account-authorization-details 命令将返回 Amazon 账户中所有 IAM 用户、组、角色和策略的信息。

```
aws iam get-account-authorization-details
```

输出：

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "RoleId": "AROA1234567890EXAMPLE",
            "CreateDate": "2014-07-30T17:09:20Z",
            "InstanceProfileList": [
                {
                    "InstanceProfileId": "AIPA1234567890EXAMPLE",
                    "Roles": [
                        {
                            "AssumeRolePolicyDocument": {
                                "Version":"2012-10-17",
                                "Statement": [
                                    {
                                        "Sid": "",
                                        "Effect": "Allow",
                                        "Principal": {
                                            "Service": "ec2.amazonaws.com"
                                        },
                                        "Action": "sts:AssumeRole"
                                    }
                                ]
                            },
                            "RoleId": "AROA1234567890EXAMPLE",
                            "CreateDate": "2014-07-30T17:09:20Z",
                            "RoleName": "EC2role",
                            "Path": "/",
                            "Arn": "arn:aws:iam::123456789012:role/EC2role"
                        }
```

```
                ],
                "CreateDate": "2014-07-30T17:09:20Z",
                "InstanceProfileName": "EC2role",
                "Path": "/",
                "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
            }
        ],
        "RoleName": "EC2role",
        "Path": "/",
        "AttachedManagedPolicies": [
            {
                "PolicyName": "AmazonS3FullAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
            },
            {
                "PolicyName": "AmazonDynamoDBFullAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess"
            }
        ],
        "RoleLastUsed": {
            "Region": "us-west-2",
            "LastUsedDate": "2019-11-13T17:30:00Z"
        },
        "RolePolicyList": [],
        "Arn": "arn:aws:iam::123456789012:role/EC2role"
    }
],
"GroupDetailList": [
    {
        "GroupId": "AIDA1234567890EXAMPLE",
        "AttachedManagedPolicies": {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        "GroupName": "Admins",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "CreateDate": "2013-10-14T18:32:24Z",
        "GroupPolicyList": []
    },
    {
        "GroupId": "AIDA1234567890EXAMPLE",
        "AttachedManagedPolicies": {
            "PolicyName": "PowerUserAccess",
```

```
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
```

```
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
```

```
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
```

```
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::amzn-s3-demo-bucket",
                                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
```

```
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 Amazon 安全审计指南。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccountAuthorizationDetails。

get-account-password-policy

以下代码示例演示了如何使用 get-account-password-policy。

Amazon CLI

查看当前账户密码策略

以下 get-account-password-policy 命令将显示有关当前账户密码策略的详细信息。

```
aws iam get-account-password-policy
```

输出：

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

如果没有为账户定义密码策略，命令将返回 NoSuchEntity 错误。

有关更多信息，请参阅《Amazon IAM 用户指南》中的为 IAM 用户设置账户密码策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccountPasswordPolicy。

get-account-summary

以下代码示例演示了如何使用 get-account-summary。

Amazon CLI

获取当前账户中 IAM 实体使用情况和 IAM 配额的信息

以下 get-account-summary 命令将返回账户中当前 IAM 实体使用情况和当前 IAM 实体配额的信息。

```
aws iam get-account-summary
```

输出：

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
```

```
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

有关实体限制的更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 和 Amazon STS 配额。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccountSummary。

get-context-keys-for-custom-policy

以下代码示例演示了如何使用 get-context-keys-for-custom-policy。

Amazon CLI

示例 1：列出作为命令行参数提供的一个或多个自定义 JSON 策略所引用的上下文键

以下 get-context-keys-for-custom-policy 命令解析每个提供的策略，并列出这些策略使用的上下文键。使用此命令来确定必须提供哪些上下文键值才能成功使用策略模拟器命令 simulate-custom-policy 和 simulate-custom-policy。您还可以使用 get-context-keys-for-custom-policy 命令检索与 IAM 用户或角色关联的所有策略使用的上下文键列表。以 file:// 开头的参数值指示命令读取文件的内容，然后使用内容而不是文件名本身作为参数的值。

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

输出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

示例 2：列出作为文件输入提供的一个或多个自定义 JSON 策略所引用的上下文键

以下 `get-context-keys-for-custom-policy` 命令与前面的示例相同，只是策略是在文件中提供而不是作为参数提供。由于该命令需要 JSON 字符串列表而不是 JSON 结构列表，因此尽管您可以将其折叠成一个，但该文件的结构必须如下所示。

```
[
    "Policy1",
    "Policy2"
]
```

例如，包含上一个示例中策略的文件必须如下所示。您必须在策略字符串中每个嵌入的双引号前面加上反斜杠 " 来对其进行转义。

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow\", \"Action
\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-west-2:128716708097:table/
${aws:username}\", \"Condition\": {\"DateGreaterThan\": {\"aws:CurrentTime\":
 \"2015-08-16T12:00:00Z\"}}}}" ]
```

然后，可以将此文件提交给以下命令。

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

输出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Policy Simulator（Amazon CLI 和 Amazon API）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContextKeysForCustomPolicy。

get-context-keys-for-principal-policy

以下代码示例演示了如何使用 get-context-keys-for-principal-policy。

Amazon CLI

列出与 IAM 主体关联的所有策略引用的上下文键

以下 get-context-keys-for-principal-policy 命令检索附加到用户 saanvi 及其所属任何组的所有策略。然后，该命令会解析每个策略并列出这些策略使用的上下文键。使用此命令来确定必须提供哪些上下文键值才能成功使用 simulate-custom-policy 和 simulate-principal-policy 命令。您还可以使用 get-context-keys-for-custom-policy 命令检索任意 JSON 策略使用的上下文键列表。

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

输出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Policy Simulator（Amazon CLI 和 Amazon API）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContextKeysForPrincipalPolicy。

get-credential-report

以下代码示例演示了如何使用 get-credential-report。

Amazon CLI

获取凭证报告

此示例打开返回的报告，并将其作为文本行数组输出到管道。

```
aws iam get-credential-report
```

输出：

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的获取 Amazon 账户的凭证报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCredentialReport。

get-group-policy

以下代码示例演示了如何使用 get-group-policy。

Amazon CLI

获取有关附加到 IAM 组的策略的信息

以下 get-group-policy 命令获取有关附加到名为 Test-Group 的组的指定策略的信息。

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

输出：

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupPolicy。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

获取 IAM 组

此示例返回有关 IAM 组 Admins 的详细信息。

```
aws iam get-group \
    --group-name Admins
```

输出：

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
```

```
        "GroupName": "Admins"
    },
    "Users": []
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 身份（用户、用户组和角色）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

get-instance-profile

以下代码示例演示了如何使用 get-instance-profile。

Amazon CLI

获取有关实例配置文件的信息

以下 get-instance-profile 命令可获取名为 ExampleInstanceProfile 的实例配置文件的信息。

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

输出：

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/ExampleInstanceProfile"
    }
```

```
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceProfile。

get-login-profile

以下代码示例演示了如何使用 get-login-profile。

Amazon CLI

获取 IAM 用户的密码信息

以下 get-login-profile 命令可获取名为 Bob 的 IAM 用户的密码相关信息。

```
aws iam get-login-profile \
    --user-name Bob
```

输出：

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

get-login-profile 命令可用于验证 IAM 用户是否有密码。如果没有为用户定义密码，则该命令将返回 NoSuchEntity 错误。

您无法使用此命令查看密码。如果密码丢失，则可以为用户重置密码（update-login-profile）。或者，您可以删除用户的登录配置文件（delete-login-profile），然后创建新的登录配置文件（create-login-profile）。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoginProfile。

get-mfa-device

以下代码示例演示了如何使用 get-mfa-device。

Amazon CLI

检索有关 FIDO 安全密钥的信息

以下 get-mfa-device 命令示例检索有关指定 FIDO 安全密钥的信息。

```
aws iam get-mfa-device \
    --serial-number arn:aws:iam::123456789012:u2f/user/alice/fidokeyname-
EXAMPLEBN5FHTECLFG7EXAMPLE
```

输出：

```
{
    "UserName": "alice",
    "SerialNumber": "arn:aws:iam::123456789012:u2f/user/alice/fidokeyname-
EXAMPLEBN5FHTECLFG7EXAMPLE",
    "EnableDate": "2023-09-19T01:49:18+00:00",
    "Certifications": {
        "FIDO": "L1"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 中使用多重身份验证（MFA）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMfaDevice。

get-open-id-connect-provider

以下代码示例演示了如何使用 get-open-id-connect-provider。

Amazon CLI

返回有关指定 OpenID Connect 提供者的信息

此示例返回有关 ARN 为 arn:aws:iam::123456789012:oidc-provider/server.example.com 的 OpenID Connect 提供者的详细信息。

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

输出：

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOpenIdConnectProvider。

`get-organizations-access-report`

以下代码示例演示了如何使用 `get-organizations-access-report`。

Amazon CLI

检索访问报告

以下 `get-organizations-access-report` 示例显示了以前为 Amazon Organizations 实体生成的访问报告。要生成报告，请使用 `generate-organizations-access-report` 命令。

```
aws iam get-organizations-access-report \
    --job-id a8b6c06f-aaa4-8xmp-28bc-81da71836359
```

输出：

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-09-30T06:53:36.187Z",
    "JobCompletionDate": "2019-09-30T06:53:37.547Z",
    "NumberOfServicesAccessible": 188,
    "NumberOfServicesNotAccessed": 171,
    "AccessDetails": [
        {
```

```
            "ServiceName": "Alexa for Business",
            "ServiceNamespace": "a4b",
            "TotalAuthenticatedEntities": 0
        },
        ...
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用上次访问的信息优化 Amazon 中的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOrganizationsAccessReport。

get-policy-version

以下代码示例演示了如何使用 get-policy-version。

Amazon CLI

检索有关指定托管策略的指定版本的信息

此示例将返回 ARN 为 arn:aws:iam::123456789012:policy/MyManagedPolicy 的策略 v2 版本的策略文档。

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

输出：

```
{
    "PolicyVersion": {
        "Document": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
```

```
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicyVersion。

get-policy

以下代码示例演示了如何使用 get-policy。

Amazon CLI

检索有关指定托管策略的信息

此示例将返回 ARN 为 arn:aws:iam::123456789012:policy/MySamplePolicy 的托管策略的详细信息。

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

输出：

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicy。

get-role-policy

以下代码示例演示了如何使用 get-role-policy。

Amazon CLI

获取有关附加到 IAM 角色的策略的信息

以下 get-role-policy 命令获取有关附加到名为 Test-Role 的角色的指定策略的信息。

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

输出：

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRolePolicy。

get-role

以下代码示例演示了如何使用 get-role。

Amazon CLI

获取有关 IAM 角色的信息

以下 `get-role` 命令可获取名为 `Test-Role` 的角色的信息。

```
aws iam get-role \
    --role-name Test-Role
```

输出：

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

该命令会显示附加到角色的信任策略。要列出附加到角色的权限策略，请使用 `list-role-policies` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRole。

get-saml-provider

以下代码示例演示了如何使用 `get-saml-provider`。

Amazon CLI

检索 SAML 提供者元文档

此示例检索有关 ARN 为 arn:aws:iam::123456789012:saml-provider/SAMLADFS 的 SAML 2.0 提供者的详细信息。响应包括您从身份提供者那里获得的、用于创建 Amazon SAML 提供者实体的元数据文档，以及创建日期和到期日期。

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

输出：

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM SAML 身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSamlProvider。

get-server-certificate

以下代码示例演示了如何使用 get-server-certificate。

Amazon CLI

获取有关 Amazon 账户中服务器证书的详细信息

以下 get-server-certificate 命令将检索 Amazon 账户中指定服务器证书的所有详细信息。

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

输出：

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
```

```
}
```

要列出 Amazon 账户中可用的服务器证书，请使用 `list-server-certificates` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 中管理服务器证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServerCertificate。

`get-service-last-accessed-details-with-entities`

以下代码示例演示了如何使用 `get-service-last-accessed-details-with-entities`。

Amazon CLI

检索包含服务详细信息的服务访问报告

以下 `get-service-last-accessed-details-with-entities` 示例检索一份报告，其中包含有关访问指定服务的 IAM 用户和其他实体的详细信息。要生成报告，请使用 `generate-service-last-accessed-details` 命令。要获取使用命名空间访问的服务列表，请使用 `get-service-last-accessed-details`。

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

输出：

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
```

```
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用上次访问的信息优化 Amazon 中的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceLastAccessedDetailsWithEntities。

get-service-last-accessed-details

以下代码示例演示了如何使用 get-service-last-accessed-details。

Amazon CLI

检索服务访问报告

以下 get-service-last-accessed-details 示例检索之前生成的报告，其中列出了 IAM 实体所访问的服务。要生成报告，请使用 generate-service-last-accessed-details 命令。

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

输出：

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
```

```
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用上次访问的信息优化 Amazon 中的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceLastAccessedDetails。

get-service-linked-role-deletion-status

以下代码示例演示了如何使用 get-service-linked-role-deletion-status。

Amazon CLI

查看删除服务相关角色的请求状态

以下 get-service-linked-role-deletion-status 示例演示了先前请求删除服务相关角色的状态。删除操作异步进行。当您发出请求时，您将得到一个 DeletionTaskId 值，该值将作为此命令的参数提供。

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

输出：

```
{
"Status": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用服务相关角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceLinkedRoleDeletionStatus。

get-ssh-public-key

以下代码示例演示了如何使用 get-ssh-public-key。

Amazon CLI

示例 1：检索以 SSH 编码形式附加到 IAM 用户的 SSH 公钥

以下 get-ssh-public-key 命令从 IAM 用户 sofia 那里检索指定的 SSH 公钥。输出采用 SSH 编码。

```
aws iam get-ssh-public-key \
    --user-name sofia \
    --ssh-public-key-id APKA123456789EXAMPLE \
    --encoding SSH
```

输出：

```
{
    "SSHPublicKey": {
        "UserName": "sofia",
        "SSHPublicKeyId": "APKA123456789EXAMPLE",
        "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",
        "SSHPublicKeyBody": "ssh-rsa <<long encoded SSH string>>",
        "Status": "Inactive",
        "UploadDate": "2019-04-18T17:04:49+00:00"
    }
}
```

示例 2：检索以 PEM 编码形式附加到 IAM 用户的 SSH 公钥

以下 get-ssh-public-key 命令从 IAM 用户 sofia 那里检索指定的 SSH 公钥。输出采用 PEM 编码。

```
aws iam get-ssh-public-key \
    --user-name sofia \
    --ssh-public-key-id APKA123456789EXAMPLE \
    --encoding PEM
```

输出：

```
{
```

```
    "SSHPublicKey": {
        "UserName": "sofia",
        "SSHPublicKeyId": "APKA123456789EXAMPLE",
        "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",
        "SSHPublicKeyBody": ""-----BEGIN PUBLIC KEY-----\n<<long encoded PEM
 string>>\n-----END PUBLIC KEY-----\n"",
        "Status": "Inactive",
        "UploadDate": "2019-04-18T17:04:49+00:00"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的将 SSH 密钥和 SSH 用于 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSshPublicKey。

get-user-policy

以下代码示例演示了如何使用 get-user-policy。

Amazon CLI

列出 IAM 用户的策略详细信息

以下 get-user-policy 命令将列出附加到名为 Bob 的 IAM 用户的指定策略的详细信息。

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

输出：

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
```

```
        ]
    }
}
```

要获取 IAM 用户的策略列表，请使用 `list-user-policies` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUserPolicy。

`get-user`

以下代码示例演示了如何使用 `get-user`。

Amazon CLI

获取有关 IAM 用户的信息

以下 `get-user` 命令可获取名为 `Paulo` 的 IAM 用户的信息。

```
aws iam get-user \
    --user-name Paulo
```

输出：

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUser。

`list-access-keys`

以下代码示例演示了如何使用 `list-access-keys`。

Amazon CLI

列出 IAM 用户的访问密钥 ID

以下 `list-access-keys` 命令将列出名为 Bob 的 IAM 用户的访问密钥 ID。

```
aws iam list-access-keys \
    --user-name Bob
```

输出：

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

无法列出 IAM 用户的秘密访问密钥。如果秘密访问密钥丢失，则必须使用 `create-access-keys` 命令创建新的访问密钥。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessKeys。

`list-account-aliases`

以下代码示例演示了如何使用 `list-account-aliases`。

Amazon CLI

列出账户别名

以下 list-account-aliases 命令将列出当前账户的别名。

```
aws iam list-account-aliases
```

输出：

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的您的 Amazon 账户 ID 及其别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccountAliases。

list-attached-group-policies

以下代码示例演示了如何使用 list-attached-group-policies。

Amazon CLI

列出附加到指定组的所有托管策略

此示例将返回附加到 Amazon 账户中名为 Admins 的 IAM 组的托管策略名称和 ARN。

```
aws iam list-attached-group-policies \
    --group-name Admins
```

输出：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
```

```
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttachedGroupPolicies。

`list-attached-role-policies`

以下代码示例演示了如何使用 `list-attached-role-policies`。

Amazon CLI

列出附加到指定角色的所有托管策略

此命令将返回附加到 Amazon 账户中名为 `SecurityAuditRole` 的 IAM 角色的托管策略名称和 ARN。

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

输出：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttachedRolePolicies。

`list-attached-user-policies`

以下代码示例演示了如何使用 `list-attached-user-policies`。

Amazon CLI

列出附加到指定用户的所有托管策略

此命令将返回附加到 Amazon 账户中名为 Bob 的 IAM 用户的托管策略名称和 ARN。

```
aws iam list-attached-user-policies \
    --user-name Bob
```

输出：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttachedUserPolicies。

list-entities-for-policy

以下代码示例演示了如何使用 list-entities-for-policy。

Amazon CLI

列出指定托管策略所附加到的所有用户、组和角色

此示例返回附加了策略 arn:aws:iam::123456789012:policy/TestPolicy 的 IAM 组、角色和用户的列表。

```
aws iam list-entities-for-policy \
```

```
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

输出：

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntitiesForPolicy。

list-group-policies

以下代码示例演示了如何使用 list-group-policies。

Amazon CLI

列出附加到指定组的所有内联策略

以下 list-group-policies 命令列出附加到当前文档中名为 Admins 的 IAM 组的内联策略名称。

```
aws iam list-group-policies \
```

```
    --group-name Admins
```

输出：

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupPolicies。

list-groups-for-user

以下代码示例演示了如何使用 list-groups-for-user。

Amazon CLI

列出 IAM 用户所属的组

以下 list-groups-for-user 命令显示名为 Bob 的 IAM 用户所属的组。

```
aws iam list-groups-for-user \
    --user-name Bob
```

输出：

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
```

```
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupsForUser。

`list-groups`

以下代码示例演示了如何使用 `list-groups`。

Amazon CLI

列出当前账户的 IAM 组

以下 `list-groups` 命令将列出当前账户中的 IAM 组。

```
aws iam list-groups
```

输出：

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

list-instance-profile-tags

以下代码示例演示了如何使用 list-instance-profile-tags。

Amazon CLI

列出附加到某个实例配置文件的标签

以下 list-instance-profile-tags 命令检索与指定实例配置文件关联的标签列表。

```
aws iam list-instance-profile-tags \
    --instance-profile-name deployment-role
```

输出：

```
{
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstanceProfileTags。

list-instance-profiles-for-role

以下代码示例演示了如何使用 list-instance-profiles-for-role。

Amazon CLI

列出 IAM 角色的实例配置文件

以下 `list-instance-profiles-for-role` 命令列出了中与角色 `Test-Role` 关联的实例配置文件。

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

输出：

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstanceProfilesForRole。

`list-instance-profiles`

以下代码示例演示了如何使用 `list-instance-profiles`。

Amazon CLI

列出账户的实例配置文件

以下 `list-instance-profiles` 命令列出与当前账户关联的实例配置文件。

```
aws iam list-instance-profiles
```

输出：

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
```

```
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstanceProfiles。

list-mfa-device-tags

以下代码示例演示了如何使用 list-mfa-device-tags。

Amazon CLI

列出附加到 MFA 设备的标签

以下 list-mfa-device-tags 命令检索与指定 MFA 设备关联的标签列表。

```
aws iam list-mfa-device-tags \
```

```
    --serial-number arn:aws:iam::123456789012:mfa/alice
```

输出：

```
{
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMfaDeviceTags。

list-mfa-devices

以下代码示例演示了如何使用 list-mfa-devices。

Amazon CLI

列出指定用户的所有 MFA 设备

此示例返回有关分配给 IAM 用户 Bob 的 MFA 设备的详细信息。

```
aws iam list-mfa-devices \
    --user-name Bob
```

输出：

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
```

```
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 中使用多重身份验证（MFA）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMfaDevices。

`list-open-id-connect-provider-tags`

以下代码示例演示了如何使用 `list-open-id-connect-provider-tags`。

Amazon CLI

列出附加到兼容 OpenID Connect（OIDC）的身份提供者的标签

以下 `list-open-id-connect-provider-tags` 命令检索与指定 OIDC 身份提供者关联的标签列表。

```
aws iam list-open-id-connect-provider-tags \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

输出：

```
{
```

```
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOpenIdConnectProviderTags。

`list-open-id-connect-providers`

以下代码示例演示了如何使用 `list-open-id-connect-providers`。

Amazon CLI

列出 Amazon 账户中 OpenID Connect 提供者的相关信息

此示例返回当前 Amazon 账户中定义的所有 OpenID Connect 提供者的 ARN 列表。

```
aws iam list-open-id-connect-providers
```

输出：

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOpenIdConnectProviders。

list-organizations-features

以下代码示例演示了如何使用 list-organizations-features。

Amazon CLI

列出为组织启用的集中根访问功能

以下 list-organizations-features 命令列出为组织启用的集中根访问功能。

```
aws iam list-organizations-features
```

输出：

```
{
    "EnabledFeatures": [
        "RootCredentialsManagement",
        "RootSessions"
    ]
    "OrganizationId": "o-aa111bb222"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的集中管理成员账户的根访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListOrganizationsFeatures。

list-policies-granting-service-access

以下代码示例演示了如何使用 list-policies-granting-service-access。

Amazon CLI

列出授予主体访问指定服务的权限的策略

以下 list-policies-granting-service-access 示例检索授予 IAM 用户 sofia 访问 Amazon CodeCommit 服务的权限的策略列表。

```
aws iam list-policies-granting-service-access \
    --arn arn:aws:iam::123456789012:user/sofia \
    --service-namespaces codecommit
```

输出：

```
{
    "PoliciesGrantingServiceAccess": [
        {
            "ServiceNamespace": "codecommit",
            "Policies": [
                {
                    "PolicyName": "Grant-Sofia-Access-To-CodeCommit",
                    "PolicyType": "INLINE",
                    "EntityType": "USER",
                    "EntityName": "sofia"
                }
            ]
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的将 IAM 用于 CodeCommit：Git 凭证、SSH 密钥和 Amazon 访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPoliciesGrantingServiceAccess。

`list-policies`

以下代码示例演示了如何使用 `list-policies`。

Amazon CLI

列出您的 Amazon 账户可用的托管策略

此示例将返回当前 Amazon 账户中可用的前两个托管策略的集合。

```
aws iam list-policies \
    --max-items 3
```

输出：

```
{
```

```
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicies。

list-policy-tags

以下代码示例演示了如何使用 list-policy-tags。

Amazon CLI

列出附加到某个托管策略的标签

以下 list-policy-tags 命令检索与指定托管策略关联的标签列表。

```
aws iam list-policy-tags \
    --policy-arn arn:aws:iam::123456789012:policy/billing-access
```

输出：

```
{
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyTags。

list-policy-versions

以下代码示例演示了如何使用 list-policy-versions。

Amazon CLI

列出有关指定托管策略版本的信息

此示例返回 ARN 为 arn:aws:iam::123456789012:policy/MySamplePolicy 的策略的可用版本列表。

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

输出：

```
{
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyVersions。

`list-role-policies`

以下代码示例演示了如何使用 `list-role-policies`。

Amazon CLI

列出附加到 IAM 角色的策略

以下 `list-role-policies` 命令将列出指定 IAM 角色的权限策略名称。

```
aws iam list-role-policies \
    --role-name Test-Role
```

输出：

```
{
```

```
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

要查看附加到角色的信任策略，请使用 `get-role` 命令。要查看权限策略的详细信息，请使用 `get-role-policy` 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRolePolicies。

`list-role-tags`

以下代码示例演示了如何使用 `list-role-tags`。

Amazon CLI

列出附加到角色的标签

以下 `list-role-tags` 命令检索与指定角色关联的标签列表。

```
aws iam list-role-tags \
    --role-name production-role
```

输出：

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoleTags。

list-roles

以下代码示例演示了如何使用 list-roles。

Amazon CLI

列出当前账户的 IAM 角色

以下 list-roles 命令将列出当前账户中的 IAM 角色。

```
aws iam list-roles
```

输出：

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
```

```
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoles。

list-saml-provider-tags

以下代码示例演示了如何使用 list-saml-provider-tags。

Amazon CLI

列出附加到 SAML 提供者的标签

以下 list-saml-provider-tags 命令检索与指定 SAML 提供者关联的标签列表。

```
aws iam list-saml-provider-tags \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS
```

输出：

```
{
    "Tags": [
```

```
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSamlProviderTags。

`list-saml-providers`

以下代码示例演示了如何使用 `list-saml-providers`。

Amazon CLI

列出 Amazon 账户中的 SAML 提供者

此示例将检索在当前 Amazon 账户中创建的 SAML 2.0 提供者列表。

```
aws iam list-saml-providers
```

输出：

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM SAML 身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSAMLProviders。

list-server-certificate-tags

以下代码示例演示了如何使用 list-server-certificate-tags。

Amazon CLI

列出附加到某个服务器证书的标签

以下 list-server-certificate-tags 命令检索与指定服务器证书关联的标签列表。

```
aws iam list-server-certificate-tags \
    --server-certificate-name ExampleCertificate
```

输出：

```
{
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServerCertificateTags。

list-server-certificates

以下代码示例演示了如何使用 list-server-certificates。

Amazon CLI

列出 Amazon 账户中的服务器证书

以下 list-server-certificates 命令将列出 Amazon 账户中存储并可供使用的所有服务器证书。

```
aws iam list-server-certificates
```

输出：

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 中管理服务器证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServerCertificates。

list-service-specific-credential

以下代码示例演示了如何使用 list-service-specific-credential。

Amazon CLI

示例 1：列出用户的特定服务凭证

以下 list-service-specific-credentials 示例显示了分配给指定用户的所有特定服务凭证。响应中不包含密码。

```
aws iam list-service-specific-credentials \
    --user-name sofia
```

输出：

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-04-18T20:45:36+00:00",
        "ServiceName": "codecommit.amazonaws.com",
        "ServiceUserName": "sofia-at-123456789012",
        "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
        "UserName": "sofia",
        "Status": "Active"
    }
}
```

示例 2：列出筛选到指定服务的用户的特定服务凭证

以下 `list-service-specific-credentials` 示例显示分配给发出请求的用户的特定服务凭证。此列表经过筛选，仅包括指定服务的凭证。响应中不包含密码。

```
aws iam list-service-specific-credentials \
    --service-name codecommit.amazonaws.com
```

输出：

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-04-18T20:45:36+00:00",
        "ServiceName": "codecommit.amazonaws.com",
        "ServiceUserName": "sofia-at-123456789012",
        "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
        "UserName": "sofia",
        "Status": "Active"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceSpecificCredential。

list-service-specific-credentials

以下代码示例演示了如何使用 list-service-specific-credentials。

Amazon CLI

检索凭证列表

以下 list-service-specific-credentials 示例列出为名为 developer 的用户生成的用来通过 HTTPS 访问 Amazon CodeCommit 存储库的凭证。

```
aws iam list-service-specific-credentials \
    --user-name developer \
    --service-name codecommit.amazonaws.com
```

输出：

```
{
    "ServiceSpecificCredentials": [
        {
            "UserName": "developer",
            "Status": "Inactive",
            "ServiceUserName": "developer-at-123456789012",
            "CreateDate": "2019-10-01T04:31:41Z",
            "ServiceSpecificCredentialId": "ACCAQFODXMPL4YFHP7DZE",
            "ServiceName": "codecommit.amazonaws.com"
        },
        {
            "UserName": "developer",
            "Status": "Active",
            "ServiceUserName": "developer+1-at-123456789012",
            "CreateDate": "2019-10-01T04:31:45Z",
            "ServiceSpecificCredentialId": "ACCAQFOXMPL6VW57M7AJP",
            "ServiceName": "codecommit.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceSpecificCredentials。

list-signing-certificates

以下代码示例演示了如何使用 list-signing-certificates。

Amazon CLI

列出 IAM 用户的签名证书

以下 list-signing-certificates 命令列出名为 Bob 的 IAM 用户的签名证书。

```
aws iam list-signing-certificates \
    --user-name Bob
```

输出：

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----
END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon EC2 用户指南》中的管理签名证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSigningCertificates。

list-ssh-public-keys

以下代码示例演示了如何使用 list-ssh-public-keys。

Amazon CLI

列出附加到 IAM 用户的 SSH 公钥

以下 list-ssh-public-keys 示例列出附加到 IAM 用户 sofia 的 SSH 公钥。

```
aws iam list-ssh-public-keys \
```

```
    --user-name sofia
```

输出：

```
{
    "SSHPublicKeys": [
        {
            "UserName": "sofia",
            "SSHPublicKeyId": "APKA1234567890EXAMPLE",
            "Status": "Inactive",
            "UploadDate": "2019-04-18T17:04:49+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的将 SSH 密钥和 SSH 用于 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSshPublicKeys。

`list-user-policies`

以下代码示例演示了如何使用 `list-user-policies`。

Amazon CLI

列出 IAM 用户的策略

以下 `list-user-policies` 命令列出附加到名为 Bob 的 IAM 用户的策略。

```
aws iam list-user-policies \
    --user-name Bob
```

输出：

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 账户中创建 IAM 用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUserPolicies。

list-user-tags

以下代码示例演示了如何使用 list-user-tags。

Amazon CLI

列出附加到用户的标签

以下 list-user-tags 命令检索与指定 IAM 用户关联的标签列表。

```
aws iam list-user-tags \
    --user-name alice
```

输出：

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUserTags。

list-users

以下代码示例演示了如何使用 list-users。

Amazon CLI

列出 IAM 用户

以下 `list-users` 命令将列出当前账户中的 IAM 用户。

```
aws iam list-users
```

输出：

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的列出 IAM 用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

`list-virtual-mfa-devices`

以下代码示例演示了如何使用 `list-virtual-mfa-devices`。

Amazon CLI

列出虚拟 MFA 设备

以下 `list-virtual-mfa-devices` 命令列出已为当前账户配置的虚拟 MFA 设备。

```
aws iam list-virtual-mfa-devices
```

输出：

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的启用虚拟多重身份验证（MFA）设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVirtualMfaDevices。

put-group-policy

以下代码示例演示了如何使用 put-group-policy。

Amazon CLI

为组添加策略

以下 put-group-policy 命令可将策略添加到名为 Admins 的 IAM 组。

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

此命令不生成任何输出。

该策略在 AdminPolicy.json 文件中定义为 JSON 文档。（文件名和扩展名没有意义。）

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM policy。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutGroupPolicy。

put-role-permissions-boundary

以下代码示例演示了如何使用 put-role-permissions-boundary。

Amazon CLI

示例 1：将基于自定义策略的权限边界应用于 IAM 角色

以下 put-role-permissions-boundary 示例应用名为 intern-boundary 的自定义策略作为指定 IAM 角色的权限边界。

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

此命令不生成任何输出。

示例 2：将基于 Amazon 托管策略的权限边界应用于 IAM 角色

以下 put-role-permissions-boundary 示例应用 Amazon 托管 PowerUserAccess 策略作为指定 IAM 角色的权限边界。

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRolePermissionsBoundary。

put-role-policy

以下代码示例演示了如何使用 put-role-policy。

Amazon CLI

将权限策略附加到 IAM 角色

以下 put-role-policy 命令可将权限策略附加到名为 Test-Role 的角色。

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
```

```
    --policy-document file://AdminPolicy.json
```

此命令不生成任何输出。

该策略在 AdminPolicy.json 文件中定义为 JSON 文档。（文件名和扩展名没有意义。）

要将信任策略附加到角色，请使用 update-assume-role-policy 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRolePolicy。

put-user-permissions-boundary

以下代码示例演示了如何使用 put-user-permissions-boundary。

Amazon CLI

示例 1：将基于自定义策略的权限边界应用于 IAM 用户

以下 put-user-permissions-boundary 示例应用名为 intern-boundary 的自定义策略作为指定 IAM 用户的权限边界。

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

此命令不生成任何输出。

示例 2：将基于 Amazon 托管策略的权限边界应用于 IAM 用户

以下 put-user-permissions-boundary 示例应用名为 PowerUserAccess 的 Amazon 托管策略作为指定 IAM 用户的权限边界。

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的添加和移除 IAM 身份权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutUserPermissionsBoundary。

put-user-policy

以下代码示例演示了如何使用 put-user-policy。

Amazon CLI

将策略附加到 IAM 用户

以下 put-user-policy 命令可将策略附加到名为 Bob 的 IAM 用户。

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

此命令不生成任何输出。

该策略在 AdminPolicy.json 文件中定义为 JSON 文档。（文件名和扩展名没有意义。）

有关更多信息，请参阅《Amazon IAM 用户指南》中的添加和删除 IAM 身份权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutUserPolicy。

remove-client-id-from-open-id-connect-provider

以下代码示例演示了如何使用 remove-client-id-from-open-id-connect-provider。

Amazon CLI

从为指定的 IAM OpenID Connect 提供者注册的客户端 ID 列表中删除指定的客户端 ID

此示例从与 ARN 为 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com 的 IAM OIDC 提供者关联的客户端 ID 列表中删除客户端 ID My-TestApp-3。

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveClientIdFromOpenIdConnectProvider。

remove-role-from-instance-profile

以下代码示例演示了如何使用 remove-role-from-instance-profile。

Amazon CLI

从实例配置文件中删除角色

以下 remove-role-from-instance-profile 命令可将名为 Test-Role 的角色从名为 ExampleInstanceProfile 的实例配置文件中删除。

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveRoleFromInstanceProfile。

remove-user-from-group

以下代码示例演示了如何使用 remove-user-from-group。

Amazon CLI

从 IAM 组中删除用户

以下 remove-user-from-group 命令可将名为 Bob 的用户从名为 Admins 的 IAM 组中删除。

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 用户组中添加和删除用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveUserFromGroup。

reset-service-specific-credential

以下代码示例演示了如何使用 reset-service-specific-credential。

Amazon CLI

示例 1：重置附加到发出请求的用户的特定服务凭证的密码

以下 reset-service-specific-credential 示例为附加到发出请求的用户的特定服务凭证生成一个新的加密强密码。

```
aws iam reset-service-specific-credential \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

输出：

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-04-18T20:45:36+00:00",
        "ServiceName": "codecommit.amazonaws.com",
        "ServiceUserName": "sofia-at-123456789012",
        "ServicePassword": "+oaFsNk7tLco+C/obP9GhhcOzGcKOayTmE3LnAmAmH4=",
        "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
        "UserName": "sofia",
        "Status": "Active"
    }
}
```

示例 2：重置附加到指定用户的特定服务凭证的密码

以下 reset-service-specific-credential 示例为附加到指定用户的特定服务凭证生成一个新的加密强密码。

```
aws iam reset-service-specific-credential \
    --user-name sofia \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

输出：

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-04-18T20:45:36+00:00",
        "ServiceName": "codecommit.amazonaws.com",
        "ServiceUserName": "sofia-at-123456789012",
        "ServicePassword": "+oaFsNk7tLco+C/obP9GhhcOzGcKOayTmE3LnAmAmH4=",
        "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
        "UserName": "sofia",
        "Status": "Active"
    }
}
```

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetServiceSpecificCredential。

`resync-mfa-device`

以下代码示例演示了如何使用 `resync-mfa-device`。

Amazon CLI

同步 MFA 设备

以下 `resync-mfa-device` 示例将与 IAM 用户 Bob 关联且 ARN 为 `arn:aws:iam::123456789012:mfa/BobsMFADevice` 的 MFA 设备与提供这两个身份验证码的身份验证器程序同步。

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 中使用多重身份验证（MFA）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResyncMfaDevice。

set-default-policy-version

以下代码示例演示了如何使用 set-default-policy-version。

Amazon CLI

将指定策略的指定版本设置为策略的默认版本

此示例将 ARN 为 arn:aws:iam::123456789012:policy/MyPolicy 的策略的 v2 版本设置为默认活动版本。

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM 中的策略和权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetDefaultPolicyVersion。

set-security-token-service-preferences

以下代码示例演示了如何使用 set-security-token-service-preferences。

Amazon CLI

设置全局端点令牌版本

以下 set-security-token-service-preferences 示例将 Amazon STS 配置为在您针对全局端点进行身份验证时使用版本 2 令牌。

```
aws iam set-security-token-service-preferences \
    --global-endpoint-token-version v2Token
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon 区域中管理 Amazon STS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetSecurityTokenServicePreferences。

simulate-custom-policy

以下代码示例演示了如何使用 simulate-custom-policy。

Amazon CLI

示例 1：模拟与 IAM 用户或角色关联的所有 IAM 策略的影响

以下 simulate-custom-policy 演示了如何提供策略并定义变量值，然后模拟一个 API 调用来看看它被允许还是被拒绝。以下示例演示了一个策略，该策略仅在指定日期和时间后才允许访问数据库。该模拟之所以成功，是因为模拟的操作和指定的 aws:CurrentTime 变量都符合策略的要求。

```
aws iam simulate-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"*","Condition":
{"DateGreaterThan":{"aws:CurrentTime":"2018-08-16T12:00:00Z"}}}}' \
    --action-names dynamodb:CreateBackup \
    --context-
entries "ContextKeyName='aws:CurrentTime',ContextKeyValues='2019-04-25T11:00:00Z',ContextKey
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvalActionName": "dynamodb:CreateBackup",
            "EvalResourceName": "*",
            "EvalDecision": "allowed",
            "MatchedStatements": [
                {
                    "SourcePolicyId": "PolicyInputList.1",
                    "StartPosition": {
                        "Line": 1,
                        "Column": 38
                    },
                    "EndPosition": {
                        "Line": 1,
                        "Column": 167
                    }
                }
            ],
            "MissingContextValues": []
        }
    ]
}
```

示例 2：模拟策略禁止的命令

以下 simulate-custom-policy 示例演示了模拟被策略禁止的命令的结果。在此示例中，提供的日期早于策略条件所要求的日期。

```
aws iam simulate-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"*","Condition":
{"DateGreaterThan":{"aws:CurrentTime":"2018-08-16T12:00:00Z"}}}}' \
    --action-names dynamodb:CreateBackup \
    --context-
entries "ContextKeyName='aws:CurrentTime',ContextKeyValues='2014-04-25T11:00:00Z',ContextKey
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvalActionName": "dynamodb:CreateBackup",
            "EvalResourceName": "*",
            "EvalDecision": "implicitDeny",
            "MatchedStatements": [],
            "MissingContextValues": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM policy simulator 测试 IAM 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SimulateCustomPolicy。

simulate-principal-policy

以下代码示例演示了如何使用 simulate-principal-policy。

Amazon CLI

示例 1：模拟任意 IAM 策略的效果

以下 simulate-principal-policy 演示了如何模拟用户调用 API 操作并确定与该用户关联的策略是允许还是拒绝该操作。在以下示例中，用户有一个策略是仅允许 codecommit:ListRepositories 操作。

```
aws iam simulate-principal-policy \
    --policy-source-arn arn:aws:iam::123456789012:user/alejandro \
    --action-names codecommit:ListRepositories
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvalActionName": "codecommit:ListRepositories",
            "EvalResourceName": "*",
            "EvalDecision": "allowed",
            "MatchedStatements": [
                {
                    "SourcePolicyId": "Grant-Access-To-CodeCommit-ListRepo",
                    "StartPosition": {
                        "Line": 3,
                        "Column": 19
                    },
                    "EndPosition": {
                        "Line": 9,
                        "Column": 10
                    }
                }
            ],
            "MissingContextValues": []
        }
    ]
}
```

示例 2：模拟被禁止命令的效果

以下 simulate-custom-policy 示例演示了模拟被用户的策略之一禁止的命令的结果。在以下示例中，用户有一条策略是仅允许在特定日期和时间之后访问 DynamoDB 数据库。在模拟中，用户试图使用早于策略条件允许的 aws:CurrentTime 值访问数据库。

```
aws iam simulate-principal-policy \
    --policy-source-arn arn:aws:iam::123456789012:user/alejandro \
    --action-names dynamodb:CreateBackup \
    --context-
entries "ContextKeyName='aws:CurrentTime',ContextKeyValues='2018-04-25T11:00:00Z',ContextKey
```

输出：

```
{
    "EvaluationResults": [
        {
            "EvalActionName": "dynamodb:CreateBackup",
            "EvalResourceName": "*",
            "EvalDecision": "implicitDeny",
            "MatchedStatements": [],
            "MissingContextValues": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM policy simulator 测试 IAM 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SimulatePrincipalPolicy。

tag-instance-profile

以下代码示例演示了如何使用 tag-instance-profile。

Amazon CLI

为实例配置文件添加标签

以下 tag-instance-profile 命令向指定实例配置文件添加带有部门名称的标签。

```
aws iam tag-instance-profile \
    --instance-profile-name deployment-role \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagInstanceProfile。

tag-mfa-device

以下代码示例演示了如何使用 tag-mfa-device。

Amazon CLI

为 MFA 设备添加标签

以下 tag-mfa-device 命令为指定 MFA 设备添加带有部门名称的标签。

```
aws iam tag-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/alice \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagMfaDevice。

tag-open-id-connect-provider

以下代码示例演示了如何使用 tag-open-id-connect-provider。

Amazon CLI

为兼容 OpenID Connect（OIDC）的身份提供者添加标签

以下 tag-open-id-connect-provider 命令为指定 OIDC 身份提供者添加带有部门名称的标签。

```
aws iam tag-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagOpenIdConnectProvider。

tag-policy

以下代码示例演示了如何使用 tag-policy。

Amazon CLI

为客户托管策略添加标签

以下 tag-policy 命令为指定客户托管策略添加带有部门名称的标签。

```
aws iam tag-policy \
    --policy-arn arn:aws:iam::123456789012:policy/billing-access \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagPolicy。

tag-role

以下代码示例演示了如何使用 tag-role。

Amazon CLI

为角色添加标签

以下 tag-role 命令为指定角色添加带有部门名称的标签。

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagRole。

tag-saml-provider

以下代码示例演示了如何使用 tag-saml-provider。

Amazon CLI

为 SAML 提供者添加标签

以下 tag-saml-provider 命令为指定 SAML 提供者添加带有部门名称的标签。

```
aws iam tag-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagSamlProvider。

tag-server-certificate

以下代码示例演示了如何使用 tag-server-certificate。

Amazon CLI

为服务器证书添加标签

以下 tag-saml-provider 命令为指定服务器证书添加带有部门名称的标签。

```
aws iam tag-server-certificate \
    --server-certificate-name ExampleCertificate \
    --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagServerCertificate。

tag-user

以下代码示例演示了如何使用 tag-user。

Amazon CLI

为用户添加标签

以下 tag-user 命令为指定用户添加带有相关部门的标签。

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagUser。

untag-instance-profile

以下代码示例演示了如何使用 untag-instance-profile。

Amazon CLI

从实例配置文件中移除标签

以下 untag-instance-profile 命令从指定实例配置文件中移除带有键名称“Department”的任何标签。

```
aws iam untag-instance-profile \
    --instance-profile-name deployment-role \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagInstanceProfile。

untag-mfa-device

以下代码示例演示了如何使用 untag-mfa-device。

Amazon CLI

从 MFA 设备中移除标签

以下 untag-mfa-device 命令从指定 MFA 设备移除带有键名称“Department”的任何标签。

```
aws iam untag-mfa-device \
```

```
    --serial-number arn:aws:iam::123456789012:mfa/alice \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagMfaDevice。

untag-open-id-connect-provider

以下代码示例演示了如何使用 untag-open-id-connect-provider。

Amazon CLI

从 OIDC 身份提供者移除标签

以下 untag-open-id-connect-provider 命令从指定 OIDC 身份提供者中移除带有键名称“Department”的任何标签。

```
aws iam untag-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagOpenIdConnectProvider。

untag-policy

以下代码示例演示了如何使用 untag-policy。

Amazon CLI

从客户托管策略中移除标签

以下 untag-policy 命令从指定客户托管策略中移除带有键名称“Department”的任何标签。

```
aws iam untag-policy \
    --policy-arn arn:aws:iam::452925170507:policy/billing-access \
```

```
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagPolicy。

untag-role

以下代码示例演示了如何使用 untag-role。

Amazon CLI

从角色移除标签

以下 untag-role 命令从指定角色中移除带有键名称“Department”的任何标签。

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagRole。

untag-saml-provider

以下代码示例演示了如何使用 untag-saml-provider。

Amazon CLI

从 SAML 提供者中移除标签

以下 untag-saml-provider 命令从指定实例配置文件中移除带有键名称“Department”的任何标签。

```
aws iam untag-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagSamlProvider。

untag-server-certificate

以下代码示例演示了如何使用 untag-server-certificate。

Amazon CLI

从服务器证书中移除标签

以下 untag-server-certificate 命令从指定服务器证书中移除带有键名称“Department”的任何标签。

```
aws iam untag-server-certificate \
    --server-certificate-name ExampleCertificate \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagServerCertificate。

untag-user

以下代码示例演示了如何使用 untag-user。

Amazon CLI

从用户中移除标签

以下 untag-user 命令从指定用户中移除带有键名称“Department”的任何标签。

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的标记 IAM 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagUser。

update-access-key

以下代码示例演示了如何使用 update-access-key。

Amazon CLI

激活或停用 IAM 用户的访问密钥

以下 update-access-key 命令停用名为 Bob 的 IAM 用户的指定访问密钥（访问密钥 ID 和秘密访问密钥）。

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

此命令不生成任何输出。

停用密钥意味着它不能用于以编程方式访问 Amazon。但密钥仍然可用且可以重新激活。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的访问密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccessKey。

update-account-password-policy

以下代码示例演示了如何使用 update-account-password-policy。

Amazon CLI

设置或更改当前账户密码策略

以下 update-account-password-policy 命令将密码策略设置为要求长度最少为八个字符，并要求在密码中包含一个或多个数字。

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

此命令不生成任何输出。

对账户密码策略的更改会影响为账户中的 IAM 用户创建的所有新密码。密码策略更改不会影响现有的密码。

有关更多信息，请参阅《Amazon IAM 用户指南》中的为 IAM 用户设置账户密码策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccountPasswordPolicy。

update-assume-role-policy

以下代码示例演示了如何使用 update-assume-role-policy。

Amazon CLI

更新 IAM 角色的信任策略

以下 update-assume-role-policy 命令更新名为 Test-Role 的角色的信任策略。

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

此命令不生成任何输出。

信任策略在 Test-Role-Trust-Policy.json 文件中定义为 JSON 文档。（文件名和扩展名没有意义。）信任策略必须指定主体。

要更新角色的权限策略，请使用 put-role-policy 命令。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAssumeRolePolicy。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

重命名 IAM 组

以下 update-group 命令可将 IAM 组 Test 的名称更改为 Test-1。

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的重命名 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

update-login-profile

以下代码示例演示了如何使用 update-login-profile。

Amazon CLI

更新 IAM 用户的密码

以下 update-login-profile 命令为名为 Bob 的 IAM 用户创建新密码。

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

此命令不生成任何输出。

要为账户设置密码策略，请使用 update-account-password-policy 命令。如果新密码违反了账户密码策略，则该命令将返回 PasswordPolicyViolation 错误。

如果账户密码策略允许，则 IAM 用户可以使用 change-password 命令更改自己的密码。

将密码保存在安全位置。如果密码丢失，则将无法恢复，必须使用 create-login-profile 命令创建新密码。

有关更多信息，请参阅《Amazon IAM 用户指南》中的管理 IAM 用户的密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLoginProfile。

update-open-id-connect-provider-thumbprint

以下代码示例演示了如何使用 update-open-id-connect-provider-thumbprint。

Amazon CLI

将现有服务器证书指纹列表替换为新列表

此示例更新了其 ARN 为 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com 的 OIDC 提供者的证书指纹列表以使用新指纹。

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 OpenID Connect（OIDC）身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOpenIdConnectProviderThumbprint。

update-role-description

以下代码示例演示了如何使用 update-role-description。

Amazon CLI

更改 IAM 角色的描述

以下 update-role 命令可将 IAM 角色 production-role 的描述更改为 Main production role。

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

输出：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
```

```
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoleDescription。

update-role

以下代码示例演示了如何使用 update-role。

Amazon CLI

更改 IAM 角色的描述或会话持续时间

以下 update-role 命令将 IAM 角色 production-role 的描述更改为 Main production role，并将最长会话持续时间设置为 12 小时。

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的修改角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRole。

update-saml-provider

以下代码示例演示了如何使用 update-saml-provider。

Amazon CLI

更新现有 SAML 提供者的元数据文档

此示例使用文件 SAMLMetaData.xml 中的新 SAML 元数据文档更新 ARN 为 arn:aws:iam::123456789012:saml-provider/SAMLADFS 的 IAM 中的 SAML 提供者。

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

输出：

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的创建 IAM SAML 身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSamlProvider。

update-server-certificate

以下代码示例演示了如何使用 update-server-certificate。

Amazon CLI

更改 Amazon 账户中服务器证书的路径或名称

以下 update-server-certificate 命令可将证书名称从 myServerCertificate 更改为 myUpdatedServerCertificate，还会将路径更改为 /cloudfront/，以便 Amazon CloudFront 服务可以访问它。此命令不生成任何输出。运行 list-server-certificates 命令即可查看更新结果。

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的在 IAM 中管理服务器证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServerCertificate。

update-service-specific-credential

以下代码示例演示了如何使用 update-service-specific-credential。

Amazon CLI

示例 1：更新请求用户的特定服务凭证的状态

以下 update-service-specific-credential 示例更改了向 Inactive 发出请求的用户的指定凭证的状态。

```
aws iam update-service-specific-credential \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE \
    --status Inactive
```

此命令不生成任何输出。

示例 2：更新指定用户的特定服务凭证的状态

以下 update-service-specific-credential 示例将指定用户的凭证状态更改为“Inactive”。

```
aws iam update-service-specific-credential \
    --user-name sofia \
    --service-specific-credential-id ACCAEXAMPLE123EXAMPLE \
    --status Inactive
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CodeCommit 用户指南》中的创建 Git 凭证以用于通过 HTTPS 连接 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServiceSpecificCredential。

update-signing-certificate

以下代码示例演示了如何使用 update-signing-certificate。

Amazon CLI

激活或停用 IAM 用户的签名证书

以下 update-signing-certificate 命令停用名为 Bob 的 IAM 用户的指定签名证书。

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

要获取签名证书的 ID，请使用 list-signing-certificates 命令。

有关更多信息，请参阅《Amazon EC2 用户指南》中的管理签名证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSigningCertificate。

update-ssh-public-key

以下代码示例演示了如何使用 update-ssh-public-key。

Amazon CLI

更改 SSH 公钥的状态

以下 update-ssh-public-key 命令将指定公钥的状态更改为 Inactive。

```
aws iam update-ssh-public-key \
    --user-name sofia \
    --ssh-public-key-id APKA1234567890EXAMPLE \
    --status Inactive
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的将 SSH 密钥和 SSH 用于 CodeCommit。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSshPublicKey。

update-user

以下代码示例演示了如何使用 update-user。

Amazon CLI

更改 IAM 用户的名称

以下 update-user 命令将 IAM 用户 Bob 的名称更改为 Robert。

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的重命名 IAM 用户组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUser。

upload-server-certificate

以下代码示例演示了如何使用 upload-server-certificate。

Amazon CLI

将服务器证书上传到 Amazon 账户

使用以下 upload-server-certificate 命令可将服务器证书上传到 Amazon 账户。在此示例中，证书位于 public_key_cert_file.pem 文件中，关联的私有密钥位于 my_private_key.pem 文件中，而证书颁发机构（CA）提供的证书链位于 my_certificate_chain_file.pem 文件中。文件上传完成后，就位于 myServerCertificate 名称下。以 file:// 开头的参数让命令读取文件的内容，将其用作参数值，而不是文件名本身。

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

输出：

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
```

```
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

有关更多信息，请参阅《使用 IAM》指南中的“创建、上传和删除服务器证书”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadServerCertificate。

upload-signing-certificate

以下代码示例演示了如何使用 upload-signing-certificate。

Amazon CLI

上传 IAM 用户的签名证书

以下 upload-signing-certificate 命令上传名为 Bob 的 IAM 用户的签名证书。

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

输出：

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

证书位于名为 certificate.pem 的文件中，采用 PEM 格式。

有关更多信息，请参阅《使用 IAM》指南中的“创建和上传用户签名证书”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadSigningCertificate。

upload-ssh-public-key

以下代码示例演示了如何使用 upload-ssh-public-key。

Amazon CLI

上传 SSH 公钥并将其与用户关联

以下 upload-ssh-public-key 命令上传在文件 sshkey.pub 中找到的公钥并将其附加到用户 sofia。

```
aws iam upload-ssh-public-key \
    --user-name sofia \
    --ssh-public-key-body file://sshkey.pub
```

输出：

```
{
    "SSHPublicKey": {
        "UserName": "sofia",
        "SSHPublicKeyId": "APKA1234567890EXAMPLE",
        "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",
        "SSHPublicKeyBody": "ssh-rsa <<long string generated by ssh-keygen
 command>>",
        "Status": "Active",
        "UploadDate": "2019-04-18T17:04:49+00:00"
    }
}
```

有关如何以适合此命令的格式生成密钥的更多信息，请参阅《Amazon CodeCommit 用户指南》中的 SSH 和 Linux、macOS 或 Unix：为 Git 和 CodeCommit 设置公钥和私钥，或 SSH 和 Windows：为 Git 和 CodeCommit 设置公钥和私钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadSshPublicKey。

使用 Amazon CLI 的 IAM Access Analyzer 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 IAM Access Analyzer 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

apply-archive-rule

以下代码示例演示了如何使用 apply-archive-rule。

Amazon CLI

将存档规则应用于符合存档规则条件的现有调查发现

以下 apply-archive-rule 示例将存档规则应用于符合存档规则条件的现有调查发现。

```
aws accessanalyzer apply-archive-rule \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
UnusedAccess-ConsoleAnalyzer-organization \
    --rule-name MyArchiveRule
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ApplyArchiveRule。

cancel-policy-generation

以下代码示例演示了如何使用 cancel-policy-generation。

Amazon CLI

取消请求的策略生成

以下 cancel-policy-generation 示例取消请求的策略生成作业 ID。

```
aws accessanalyzer cancel-policy-generation \
    --job-id 923a56b0-ebb8-4e80-8a3c-a11ccfbcd6f2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelPolicyGeneration。

check-access-not-granted

以下代码示例演示了如何使用 check-access-not-granted。

Amazon CLI

检查策略是否不允许指定访问

以下 check-access-not-granted 示例检查策略是否不允许指定访问。

```
aws accessanalyzer check-access-not-granted \
    --policy-document file://myfile.json \
    --access actions="s3:DeleteBucket","s3:GetBucketLocation" \
    --policy-type IDENTITY_POLICY
```

的内容myfile.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
```

```
}
```

输出：

```
{
    "result": "PASS",
    "message": "The policy document does not grant access to perform one or more of
 the listed actions."
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckAccessNotGranted。

check-no-new-access

以下代码示例演示了如何使用 check-no-new-access。

Amazon CLI

检查与现有策略相比，更新后的策略是否允许新的访问

以下 check-no-new-access 示例检查与现有策略相比，更新后的策略是否允许新的访问。

```
aws accessanalyzer check-no-new-access \
    --existing-policy-document file://existing-policy.json \
    --new-policy-document file://new-policy.json \
    --policy-type IDENTITY_POLICY
```

的内容existing-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

的内容new-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

输出：

```
{
    "result": "FAIL",
    "message": "The modified permissions grant new access compared to your existing
 policy.",
    "reasons": [
        {
            "description": "New access in the statement with index: 0.",
            "statementIndex": 0
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckNoNewAccess。

check-no-public-access

以下代码示例演示了如何使用 check-no-public-access。

Amazon CLI

检查资源策略是否可以授予对指定资源类型的公共访问权限

以下 check-no-public-access 示例检查资源策略是否可以授予对指定资源类型的公共访问权限。

```
aws accessanalyzer check-no-public-access \
    --policy-document file://check-no-public-access-myfile.json \
    --resource-type AWS::S3::Bucket
```

的内容myfile.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CheckNoPublicAccess",
            "Effect": "Allow",
            "Principal": { "AWS": "arn:aws:iam::111122223333:user/JohnDoe" },
            "Action": [
                "s3:GetObject"
            ]
        }
    ]
}
```

输出：

```
{
    "result": "PASS",
    "message": "The resource policy does not grant public access for the given
 resource type."
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckNoPublicAccess。

create-access-preview

以下代码示例演示了如何使用 create-access-preview。

Amazon CLI

创建访问预览，以便允许您在部署资源权限之前预览资源的 IAM Access Analyzer 调查发现

以下 create-access-preview 示例创建访问预览，以便在向您的 Amazon 账户中部署资源权限之前允许您预览资源的 IAM Access Analyzer 调查发现。

```
aws accessanalyzer create-access-preview \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --configurations file://myfile.json
```

的内容myfile.json：

```
{
    "arn:aws:s3:::amzn-s3-demo-bucket": {
        "s3Bucket": {
            "bucketPolicy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam::111122223333:root\"]},\"Action
\":[\"s3:PutObject\",\"s3:PutObjectAcl\"],\"Resource\":\"arn:aws:s3:::amzn-s3-demo-
bucket/*\"}]}",
            "bucketPublicAccessBlock": {
                "ignorePublicAcls": true,
                "restrictPublicBuckets": true
            },
            "bucketAclGrants": [
                {
                    "grantee": {
                        "id":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
                    },
                    "permission": "READ"
                }
            ]
        }
    }
}
```

输出：

```
{
    "id": "3c65eb13-6ef9-4629-8919-a32043619e6b"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccessPreview。

create-analyzer

以下代码示例演示了如何使用 create-analyzer。

Amazon CLI

创建分析器

以下 create-analyzer 示例在您的 Amazon 账户中创建分析器。

```
aws accessanalyzer create-analyzer \
    --analyzer-name example \
    --type ACCOUNT
```

输出：

```
{
    "arn": "arn:aws:access-analyzer:us-east-2:111122223333:analyzer/example"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的开始使用 Amazon Identity and Access Management Access Analyzer 调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAnalyzer。

create-archive-rule

以下代码示例演示了如何使用 create-archive-rule。

Amazon CLI

为指定分析器创建存档规则

以下 create-archive-rule 示例在您的 Amazon 账户中为指定分析器创建存档规则。

```
aws accessanalyzer create-archive-rule \
    --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \
    --rule-name MyRule \
    --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
 ["AWS::IAM::Role"]}}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateArchiveRule。

delete-analyzer

以下代码示例演示了如何使用 delete-analyzer。

Amazon CLI

删除指定分析器

以下 delete-analyzer 示例删除您 Amazon 账户中的指定分析器。

```
aws accessanalyzer delete-analyzer \
    --analyzer-name example
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAnalyzer。

delete-archive-rule

以下代码示例演示了如何使用 delete-archive-rule。

Amazon CLI

删除指定存档规则

以下 delete-archive-rule 示例删除您 Amazon 账户中的指定存档规则。

```
aws accessanalyzer delete-archive-rule \
    --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \
    --rule-name MyRule
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteArchiveRule。

get-access-preview

以下代码示例演示了如何使用 get-access-preview。

Amazon CLI

检索有关指定分析器访问预览的信息

以下 get-access-preview 示例检索有关您 Amazon 账户中指定分析器的访问预览的信息。

```
aws accessanalyzer get-access-preview \
    --access-preview-id 3c65eb13-6ef9-4629-8919-a32043619e6b \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account
```

输出：

```
{
    "accessPreview": {
        "id": "3c65eb13-6ef9-4629-8919-a32043619e6b",
        "analyzerArn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
        "configurations": {
            "arn:aws:s3:::amzn-s3-demo-bucket": {
                "s3Bucket": {
                    "bucketPolicy": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam::111122223333:root\"]},
\"Action\":[\"s3:PutObject\",\"s3:PutObjectAcl\"],\"Resource\":\"arn:aws:s3:::amzn-
s3-demo-bucket/*\"}]}",
                    "bucketAclGrants": [
                        {
                            "permission": "READ",
```

```
                            "grantee": {
                                "id":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
                            }
                        }
                    ],
                    "bucketPublicAccessBlock": {
                        "ignorePublicAcls": true,
                        "restrictPublicBuckets": true
                    }
                }
            }
        },
        "createdAt": "2024-02-17T00:18:44+00:00",
        "status": "COMPLETED"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessPreview。

get-analyzed-resource

以下代码示例演示了如何使用 get-analyzed-resource。

Amazon CLI

检索有关已分析资源的信息

以下 get-analyzed-resource 示例检索有关您 Amazon 账户中已分析资源的信息。

```
aws accessanalyzer get-analyzed-resource \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --resource-arn arn:aws:s3:::amzn-s3-demo-bucket
```

输出：

```
{
    "resource": {
        "analyzedAt": "2024-02-15T18:01:53.002000+00:00",
```

```
        "isPublic": false,
        "resourceArn": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceOwnerAccount": "111122223333",
        "resourceType": "AWS::S3::Bucket"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAnalyzedResource。

get-analyzer

以下代码示例演示了如何使用 get-analyzer。

Amazon CLI

检索有关指定分析器的信息

以下 get-analyzer 示例检索有关您 Amazon 账户中指定分析器的信息。

```
aws accessanalyzer get-analyzer \
    --analyzer-name ConsoleAnalyzer-account
```

输出：

```
{
    "analyzer": {
        "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
        "createdAt": "2019-12-03T07:28:17+00:00",
        "lastResourceAnalyzed": "arn:aws:sns:us-west-2:111122223333:config-topic",
        "lastResourceAnalyzedAt": "2024-02-15T18:01:53.003000+00:00",
        "name": "ConsoleAnalyzer-account",
        "status": "ACTIVE",
        "tags": {
            "auto-delete": "no"
        },
        "type": "ACCOUNT"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAnalyzer。

get-archive-rule

以下代码示例演示了如何使用 get-archive-rule。

Amazon CLI

检索有关存档规则的信息

以下 get-archive-rule 示例检索有关您 Amazon 账户中存档规则的信息。

```
aws accessanalyzer get-archive-rule \
    --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \
    --rule-name MyArchiveRule
```

输出：

```
{
    "archiveRule": {
        "createdAt": "2024-02-15T00:49:27+00:00",
        "filter": {
            "resource": {
                "contains": [
                    "Cognito"
                ]
            },
            "resourceType": {
                "eq": [
                    "AWS::IAM::Role"
                ]
            }
        },
        "ruleName": "MyArchiveRule",
        "updatedAt": "2024-02-15T00:49:27+00:00"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetArchiveRule。

get-finding-v2

以下代码示例演示了如何使用 get-finding-v2。

Amazon CLI

检索有关指定调查发现的信息

以下 get-finding-v2 示例返回有关您 Amazon 账户中指定调查发现的信息。

```
aws accessanalyzer get-finding-v2 \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-organization \
    --id 0910eedb-381e-4e95-adda-0d25c19e6e90
```

输出：

```
{
    "findingDetails": [
        {
            "externalAccessDetails": {
                "action": [
                    "sts:AssumeRoleWithWebIdentity"
                ],
                "condition": {
                    "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
                },
                "isPublic": false,
                "principal": {
                    "Federated": "cognito-identity.amazonaws.com"
                }
            }
        }
    ],
    "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
    "status": "ACTIVE",
    "error": null,
    "createdAt": "2021-02-26T21:17:50.905000+00:00",
    "resourceType": "AWS::IAM::Role",
    "findingType": "ExternalAccess",
```

```
    "resourceOwnerAccount": "111122223333",
    "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
    "id": "0910eedb-381e-4e95-adda-0d25c19e6e90",
    "updatedAt": "2021-02-26T21:17:50.905000+00:00"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的查看调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFindingV2。

get-finding

以下代码示例演示了如何使用 get-finding。

Amazon CLI

检索有关指定调查发现的信息

以下 get-finding 示例返回有关您 Amazon 账户中指定调查发现的信息。

```
aws accessanalyzer get-finding \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-organization \
    --id 0910eedb-381e-4e95-adda-0d25c19e6e90
```

输出：

```
{
    "finding": {
        "id": "0910eedb-381e-4e95-adda-0d25c19e6e90",
        "principal": {
            "Federated": "cognito-identity.amazonaws.com"
        },
        "action": [
            "sts:AssumeRoleWithWebIdentity"
        ],
        "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
        "isPublic": false,
        "resourceType": "AWS::IAM::Role",
        "condition": {
            "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
        },
```

```
        "createdAt": "2021-02-26T21:17:50.905000+00:00",
        "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
        "updatedAt": "2021-02-26T21:17:50.905000+00:00",
        "status": "ACTIVE",
        "resourceOwnerAccount": "111122223333"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的查看调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFinding。

get-generated-policy

以下代码示例演示了如何使用 get-generated-policy。

Amazon CLI

检索使用 `StartPolicyGeneration` API 生成的策略

以下 get-generated-policy 示例检索在您 Amazon 账户中使用 StartPolicyGeneration API 生成的策略。

```
aws accessanalyzer get-generated-policy \
    --job-id c557dc4a-0338-4489-95dd-739014860ff9
```

输出：

```
{
    "generatedPolicyResult": {
        "generatedPolicies": [
            {
                "policy": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\"Sid\":\"SupportedServiceSid0\",\"Effect\":\"Allow\",\"Action\":
[\"access-analyzer:GetAnalyzer\",\"access-analyzer:ListAnalyzers\",
\"access-analyzer:ListArchiveRules\",\"access-analyzer:ListFindings
\",\"cloudtrail:DescribeTrails\",\"cloudtrail:GetEventDataStore\",
\"cloudtrail:GetEventSelectors\",\"cloudtrail:GetInsightSelectors
\",\"cloudtrail:GetTrailStatus\",\"cloudtrail:ListChannels\",
\"cloudtrail:ListEventDataStores\",\"cloudtrail:ListQueries\",\"cloudtrail:ListTags
\",\"cloudtrail:LookupEvents\",\"ec2:DescribeRegions\",\"iam:GetAccountSummary
\",\"iam:GetOpenIDConnectProvider\",\"iam:GetRole\",\"iam:ListAccessKeys\",
\"iam:ListAccountAliases\",\"iam:ListOpenIDConnectProviders\",\"iam:ListRoles
```

```
\",\"iam:ListSAMLProviders\",\"kms:ListAliases\",\"s3:GetBucketLocation\",
\"s3:ListAllMyBuckets\"],\"Resource\":\"*\"}]}"
            }
        ],
        "properties": {
            "cloudTrailProperties": {
                "endTime": "2024-02-14T22:44:40+00:00",
                "startTime": "2024-02-13T00:30:00+00:00",
                "trailProperties": [
                    {
                        "allRegions": true,
                        "cloudTrailArn": "arn:aws:cloudtrail:us-
west-2:111122223333:trail/my-trail",
                        "regions": []
                    }
                ]
            },
            "isComplete": false,
            "principalArn": "arn:aws:iam::111122223333:role/Admin"
        }
    },
    "jobDetails": {
        "completedOn": "2024-02-14T22:47:01+00:00",
        "jobId": "c557dc4a-0338-4489-95dd-739014860ff9",
        "startedOn": "2024-02-14T22:44:41+00:00",
        "status": "SUCCEEDED"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGeneratedPolicy。

`list-access-preview-findings`

以下代码示例演示了如何使用 `list-access-preview-findings`。

Amazon CLI

检索由指定访问预览生成的访问预览调查发现列表

以下 `list-access-preview-findings` 示例检索由您 Amazon 账户中指定访问预览生成的访问预览调查发现列表。

```
aws accessanalyzer list-access-preview-findings \
    --access-preview-id 3c65eb13-6ef9-4629-8919-a32043619e6b \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account
```

输出：

```
{
    "findings": [
        {
            "id": "e22fc158-1c87-4c32-9464-e7f405ce8d74",
            "principal": {
                "AWS": "111122223333"
            },
            "action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "condition": {},
            "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "isPublic": false,
            "resourceType": "AWS::S3::Bucket",
            "createdAt": "2024-02-17T00:18:46+00:00",
            "changeType": "NEW",
            "status": "ACTIVE",
            "resourceOwnerAccount": "111122223333",
            "sources": [
                {
                    "type": "POLICY"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessPreviewFindings。

`list-access-previews`

以下代码示例演示了如何使用 `list-access-previews`。

Amazon CLI

检索指定分析器的访问预览列表

以下 `list-access-previews` 示例检索有关您 Amazon 账户中指定分析器的访问预览列表。

```
aws accessanalyzer list-access-previews \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account
```

输出：

```
{
    "accessPreviews": [
        {
            "id": "3c65eb13-6ef9-4629-8919-a32043619e6b",
            "analyzerArn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
            "createdAt": "2024-02-17T00:18:44+00:00",
            "status": "COMPLETED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 IAM Access Analyzer API 预览访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessPreviews。

`list-analyzed-resources`

以下代码示例演示了如何使用 `list-analyzed-resources`。

Amazon CLI

列出可用的小部件

以下 `list-analyzed-resources` 示例列出您的 Amazon 账户中可用的小组件。

```
aws accessanalyzer list-analyzed-resources \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
```

```
    --resource-type AWS::IAM::Role
```

输出：

```
{
    "analyzedResources": [
        {
            "resourceArn": "arn:aws:sns:us-west-2:111122223333:Validation-Email",
            "resourceOwnerAccount": "111122223333",
            "resourceType": "AWS::SNS::Topic"
        },
        {
            "resourceArn": "arn:aws:sns:us-west-2:111122223333:admin-alerts",
            "resourceOwnerAccount": "111122223333",
            "resourceType": "AWS::SNS::Topic"
        },
        {
            "resourceArn": "arn:aws:sns:us-west-2:111122223333:config-topic",
            "resourceOwnerAccount": "111122223333",
            "resourceType": "AWS::SNS::Topic"
        },
        {
            "resourceArn": "arn:aws:sns:us-west-2:111122223333:inspector-topic",
            "resourceOwnerAccount": "111122223333",
            "resourceType": "AWS::SNS::Topic"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAnalyzedResources。

list-analyzers

以下代码示例演示了如何使用 list-analyzers。

Amazon CLI

检索分析器列表

以下 list-analyzers 示例检索您 Amazon 账户中分析器的列表。

```
aws accessanalyzer list-analyzers
```

输出：

```
{
    "analyzers": [
        {
            "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
UnusedAccess-ConsoleAnalyzer-organization",
            "createdAt": "2024-02-15T00:46:40+00:00",
            "name": "UnusedAccess-ConsoleAnalyzer-organization",
            "status": "ACTIVE",
            "tags": {
                "auto-delete": "no"
            },
            "type": "ORGANIZATION_UNUSED_ACCESS"
        },
        {
            "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-organization",
            "createdAt": "2020-04-25T07:43:28+00:00",
            "lastResourceAnalyzed": "arn:aws:s3:::amzn-s3-demo-bucket",
            "lastResourceAnalyzedAt": "2024-02-15T21:51:56.517000+00:00",
            "name": "ConsoleAnalyzer-organization",
            "status": "ACTIVE",
            "tags": {
                "auto-delete": "no"
            },
            "type": "ORGANIZATION"
        },
        {
            "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
            "createdAt": "2019-12-03T07:28:17+00:00",
            "lastResourceAnalyzed": "arn:aws:sns:us-west-2:111122223333:config-
topic",
            "lastResourceAnalyzedAt": "2024-02-15T18:01:53.003000+00:00",
            "name": "ConsoleAnalyzer-account",
            "status": "ACTIVE",
            "tags": {
                "auto-delete": "no"
            },
            "type": "ACCOUNT"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAnalyzers。

list-archive-rules

以下代码示例演示了如何使用 list-archive-rules。

Amazon CLI

检索为指定分析器创建的存档规则列表

以下 list-archive-rules 示例检索您 Amazon 账户中为指定分析器创建的存档规则列表。

```
aws accessanalyzer list-archive-rules \
    --analyzer-name UnusedAccess-ConsoleAnalyzer-organization
```

输出：

```
{
    "archiveRules": [
        {
            "createdAt": "2024-02-15T00:49:27+00:00",
            "filter": {
                "resource": {
                    "contains": [
                        "Cognito"
                    ]
                },
                "resourceType": {
                    "eq": [
                        "AWS::IAM::Role"
                    ]
                }
            },
            "ruleName": "MyArchiveRule",
            "updatedAt": "2024-02-15T00:49:27+00:00"
```

```
        },
        {
            "createdAt": "2024-02-15T23:27:45+00:00",
            "filter": {
                "findingType": {
                    "eq": [
                        "UnusedIAMUserAccessKey"
                    ]
                }
            },
            "ruleName": "ArchiveRule-56125a39-e517-4ff8-afb1-ef06f58db612",
            "updatedAt": "2024-02-15T23:27:45+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListArchiveRules。

`list-findings-v2`

以下代码示例演示了如何使用 `list-findings-v2`。

Amazon CLI

检索由指定分析器生成的调查发现列表

以下 `list-findings-v2` 示例检索由您 Amazon 账户中指定分析器生成的调查发现列表。此示例筛选结果以便仅包括其名称包含 Cognito 的 IAM 角色。

```
aws accessanalyzer list-findings-v2 \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
 ["AWS::IAM::Role"]}}'
```

输出：

```
{
    "findings": [
```

```
        {
            "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
            "createdAt": "2021-02-26T21:17:24.710000+00:00",
            "id": "597f3bc2-3adc-4c18-9879-5c4b23485e46",
            "resource": "arn:aws:iam::111122223333:role/
Cognito_testpoolUnauth_Role",
            "resourceType": "AWS::IAM::Role",
            "resourceOwnerAccount": "111122223333",
            "status": "ACTIVE",
            "updatedAt": "2021-02-26T21:17:24.710000+00:00",
            "findingType": "ExternalAccess"
        },
        {
            "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
            "createdAt": "2021-02-26T21:17:50.905000+00:00",
            "id": "ce0e221a-85b9-4d52-91ff-d7678075442f",
            "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
            "resourceType": "AWS::IAM::Role",
            "resourceOwnerAccount": "111122223333",
            "status": "ACTIVE",
            "updatedAt": "2021-02-26T21:17:50.905000+00:00",
            "findingType": "ExternalAccess"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFindingsV2。

`list-findings`

以下代码示例演示了如何使用 `list-findings`。

Amazon CLI

检索由指定分析器生成的调查发现列表

以下 `list-findings` 示例检索由您 Amazon 账户中指定分析器生成的调查发现列表。此示例筛选结果以便仅包括其名称包含 `Cognito` 的 IAM 角色。

```
aws accessanalyzer list-findings \
```

```
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
 ["AWS::IAM::Role"]}}'
```

输出：

```
{
    "findings": [
        {
            "id": "597f3bc2-3adc-4c18-9879-5c4b23485e46",
            "principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "action": [
                "sts:AssumeRoleWithWebIdentity"
            ],
            "resource": "arn:aws:iam::111122223333:role/
Cognito_testpoolUnauth_Role",
            "isPublic": false,
            "resourceType": "AWS::IAM::Role",
            "condition": {
                "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
            },
            "createdAt": "2021-02-26T21:17:24.710000+00:00",
            "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
            "updatedAt": "2021-02-26T21:17:24.710000+00:00",
            "status": "ACTIVE",
            "resourceOwnerAccount": "111122223333"
        },
        {
            "id": "ce0e221a-85b9-4d52-91ff-d7678075442f",
            "principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "action": [
                "sts:AssumeRoleWithWebIdentity"
            ],
            "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
            "isPublic": false,
            "resourceType": "AWS::IAM::Role",
            "condition": {
```

```
                "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
            },
            "createdAt": "2021-02-26T21:17:50.905000+00:00",
            "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
            "updatedAt": "2021-02-26T21:17:50.905000+00:00",
            "status": "ACTIVE",
            "resourceOwnerAccount": "111122223333"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFindings。

list-policy-generations

以下代码示例演示了如何使用 list-policy-generations。

Amazon CLI

列出最近七天请求的所有策略生成

以下 list-policy-generations 示例列出了您 Amazon 账户中最近七天请求的所有策略生成。

```
aws accessanalyzer list-policy-generations
```

输出：

```
{
    "policyGenerations": [
        {
            "completedOn": "2024-02-14T23:43:38+00:00",
            "jobId": "923a56b0-ebb8-4e80-8a3c-a11ccfbcd6f2",
            "principalArn": "arn:aws:iam::111122223333:role/Admin",
            "startedOn": "2024-02-14T23:43:02+00:00",
            "status": "CANCELED"
        },
        {
            "completedOn": "2024-02-14T22:47:01+00:00",
```

```
            "jobId": "c557dc4a-0338-4489-95dd-739014860ff9",
            "principalArn": "arn:aws:iam::111122223333:role/Admin",
            "startedOn": "2024-02-14T22:44:41+00:00",
            "status": "SUCCEEDED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyGenerations。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

检索应用于指定资源的标签列表

以下 `list-tags-for-resource` 示例检索您 Amazon 账户中应用于指定资源的标签列表。

```
aws accessanalyzer list-tags-for-resource \
    --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account
```

输出：

```
{
    "tags": {
        "Zone-of-trust": "Account",
        "Name": "ConsoleAnalyzer"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`start-policy-generation`

以下代码示例演示了如何使用 `start-policy-generation`。

Amazon CLI

启动策略生成请求

以下 start-policy-generation 示例在您 Amazon 账户中启动策略生成请求。

```
aws accessanalyzer start-policy-generation \
    --policy-generation-details '{"principalArn":"arn:aws:iam::111122223333:role/Admin"}' \
    --cloud-trail-details file://myfile.json
```

的内容myfile.json：

```
{
    "accessRole": "arn:aws:iam::111122223333:role/service-role/AccessAnalyzerMonitorServiceRole",
    "startTime": "2024-02-13T00:30:00Z",
    "trails": [
        {
            "allRegions": true,
            "cloudTrailArn": "arn:aws:cloudtrail:us-west-2:111122223333:trail/my-trail"
        }
    ]
}
```

输出：

```
{
    "jobId": "c557dc4a-0338-4489-95dd-739014860ff9"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPolicyGeneration。

start-resource-scan

以下代码示例演示了如何使用 start-resource-scan。

Amazon CLI

立即开始扫描应用于指定资源的策略

以下 start-resource-scan 示例立即开始扫描应用于您 Amazon 账户中指定资源的策略。

```
aws accessanalyzer start-resource-scan \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --resource-arn arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的 IAM Access Analyzer 策略生成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartResourceScan。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为指定资源添加标签

以下 tag-resource 示例为您 Amazon 账户中的指定资源添加标签。

```
aws accessanalyzer tag-resource \
    --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --tags Environment=dev,Purpose=testing
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除指定资源的标签

以下 untag-resource 示例移除您 Amazon 账户中指定资源的标签。

```
aws accessanalyzer untag-resource \
    --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
    --tag-keys Environment Purpose
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-archive-rule

以下代码示例演示了如何使用 update-archive-rule。

Amazon CLI

更新指定存档规则的条件和值

以下 update-archive-rule 示例更新您 Amazon 账户中指定存档规则的条件和值。

```
aws accessanalyzer update-archive-rule \
    --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \
    --rule-name MyArchiveRule \
    --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
 ["AWS::IAM::Role"]}}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的存档规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateArchiveRule。

update-findings

以下代码示例演示了如何使用 update-findings。

Amazon CLI

更新指定调查发现的状态

以下 update-findings 示例更新您 Amazon 账户中指定调查发现的状态。

```
aws accessanalyzer update-findings \
    --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
UnusedAccess-ConsoleAnalyzer-organization \
    --ids 4f319ac3-2e0c-4dc4-bf51-7013a086b6ae 780d586a-2cce-4f72-aff6-359d450e7500
 \
    --status ARCHIVED
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IAM 用户指南》中的使用 Amazon Identity and Access Management Access Analyzer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFindings。

validate-policy

以下代码示例演示了如何使用 validate-policy。

Amazon CLI

请求验证策略并返回调查发现列表

以下 validate-policy 示例请求验证策略并返回调查发现列表。此示例中的策略是用于网络身份联合验证的 Amazon Cognito 角色的角色信任策略。信任策略生成的结果与空 Sid 元素值和不匹配的策略主体有关，原因是使用了不正确的假设角色操作 sts:AssumeRole。与 Cognito 一起使用的正确假设角色操作是 sts:AssumeRoleWithWebIdentity。

```
aws accessanalyzer validate-policy \
    --policy-document file://myfile.json \
    --policy-type RESOURCE_POLICY
```

的内容myfile.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
```

```
            "Principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ],
            "Condition": {
                "StringEquals": {
                    "cognito-identity.amazonaws.com:aud": "us-west-2_EXAMPLE"
                }
            }
        }
    ]
}
```

输出：

```
{
    "findings": [
        {
            "findingDetails": "Add a value to the empty string in the Sid element.",
            "findingType": "SUGGESTION",
            "issueCode": "EMPTY_SID_VALUE",
            "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/
access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-
checks-suggestion-empty-sid-value",
            "locations": [
                {
                    "path": [
                        {
                            "value": "Statement"
                        },
                        {
                            "index": 0
                        },
                        {
                            "value": "Sid"
                        }
                    ],
                    "span": {
                        "end": {
                            "column": 21,
```

```
                            "line": 5,
                            "offset": 81
                        },
                        "start": {
                            "column": 19,
                            "line": 5,
                            "offset": 79
                        }
                    }
                }
            ]
        },
        {
            "findingDetails": "The sts:AssumeRole action is invalid with the
 following principal(s): cognito-identity.amazonaws.com. Use a SAML provider
 principal with the sts:AssumeRoleWithSAML action or use an OIDC provider principal
 with the sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if
 you use either of the two options.",
            "findingType": "ERROR",
            "issueCode": "MISMATCHED_ACTION_FOR_PRINCIPAL",
            "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/
access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-
checks-error-mismatched-action-for-principal",
            "locations": [
                {
                    "path": [
                        {
                            "value": "Statement"
                        },
                        {
                            "index": 0
                        },
                        {
                            "value": "Action"
                        },
                        {
                            "index": 0
                        }
                    ],
                    "span": {
                        "end": {
                            "column": 32,
                            "line": 11,
                            "offset": 274
```

```
                        },
                        "start": {
                            "column": 16,
                            "line": 11,
                            "offset": 258
                        }
                    }
                },
                {
                    "path": [
                        {
                            "value": "Statement"
                        },
                        {
                            "index": 0
                        },
                        {
                            "value": "Principal"
                        },
                        {
                            "value": "Federated"
                        }
                    ],
                    "span": {
                        "end": {
                            "column": 61,
                            "line": 8,
                            "offset": 202
                        },
                        "start": {
                            "column": 29,
                            "line": 8,
                            "offset": 170
                        }
                    }
                }
            ]
        },
        {
            "findingDetails": "The following actions: sts:TagSession are not
supported by the condition key cognito-identity.amazonaws.com:aud. The condition
will not be evaluated for these actions. We recommend that you move these actions
to a different statement without this condition key.",
            "findingType": "ERROR",
```

```
            "issueCode": "UNSUPPORTED_ACTION_FOR_CONDITION_KEY",
            "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/
access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-
checks-error-unsupported-action-for-condition-key",
            "locations": [
                {
                    "path": [
                        {
                            "value": "Statement"
                        },
                        {
                            "index": 0
                        },
                        {
                            "value": "Action"
                        },
                        {
                            "index": 1
                        }
                    ],
                    "span": {
                        "end": {
                            "column": 32,
                            "line": 12,
                            "offset": 308
                        },
                        "start": {
                            "column": 16,
                            "line": 12,
                            "offset": 292
                        }
                    }
                },
                {
                    "path": [
                        {
                            "value": "Statement"
                        },
                        {
                            "index": 0
                        },
                        {
                            "value": "Condition"
                        },
```

```
                        {
                            "value": "StringEquals"
                        },
                        {
                            "value": "cognito-identity.amazonaws.com:aud"
                        }
                    ],
                    "span": {
                        "end": {
                            "column": 79,
                            "line": 16,
                            "offset": 464
                        },
                        "start": {
                            "column": 58,
                            "line": 16,
                            "offset": 443
                        }
                    }
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的检查验证策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidatePolicy。

使用 Amazon CLI 的 Image Builder 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Image Builder 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-component`

以下代码示例演示了如何使用 `create-component`。

Amazon CLI

创建组件

以下 `create-component` 示例创建了一个组件，该组件使用 JSON 文档文件并引用上传到 Amazon S3 存储桶的 YAML 格式的组件文档。

```
aws imagebuilder create-component \
    --cli-input-json file://create-component.json
```

的内容`create-component.json`：

```
{
    "name": "MyExampleComponent",
    "semanticVersion": "2019.12.02",
    "description": "An example component that builds, validates and tests an image",
    "changeDescription": "Initial version.",
    "platform": "Windows",
    "uri": "s3://s3-bucket-name/s3-bucket-path/component.yaml"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "componentBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/examplecomponent/2019.12.02/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateComponent。

create-distribution-configuration

以下代码示例演示了如何使用 create-distribution-configuration。

Amazon CLI

创建分发配置

以下 create-distribution-configuration 示例使用 JSON 文件创建分发配置。

```
aws imagebuilder create-distribution-configuration \
    --cli-input-json file:/create-distribution-configuration.json
```

的内容create-distribution-configuration.json：

```
{
    "name": "MyExampleDistribution",
    "description": "Copies AMI to eu-west-1",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter references",
                "amiTags": {
                    "KeyName": "{{ssm:parameter_name}}"
                },
                "launchPermission": {
                    "userIds": [
                        "123456789012"
                    ]
                }
            }
        },
        {
            "region": "eu-west-1",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "amiTags": {
                    "KeyName": "Value"
                },
                "launchPermission": {
```

```
                    "userIds": [
                        "123456789012"
                    ]
                }
            }
        }
    ]
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDistributionConfiguration。

create-image-pipeline

以下代码示例演示了如何使用 create-image-pipeline。

Amazon CLI

创建镜像管道

以下 create-image-pipeline 示例使用 JSON 文件创建了一个镜像管道。

```
aws imagebuilder create-image-pipeline \
    --cli-input-json file://create-image-pipeline.json
```

的内容create-image-pipeline.json：

```
{
    "name": "MyWindows2016Pipeline",
    "description": "Builds Windows 2016 Images",
```

```
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * SUN)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateImagePipeline。

`create-image-recipe`

以下代码示例演示了如何使用 `create-image-recipe`。

Amazon CLI

创建配方

以下 `create-image-recipe` 示例使用 JSON 文件创建了一个镜像配方。组件是按指定顺序安装的。

```
aws imagebuilder create-image-recipe \
    --cli-input-json file://create-image-recipe.json
```

的内容create-image-recipe.json：

```
{
    "name": "MyBasicRecipe",
    "description": "This example image recipe creates a Windows 2016 image.",
    "semanticVersion": "2019.12.03",
    "components":
    [
        {
            "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
myexamplecomponent/2019.12.02/1"
        },
        {
            "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
myimportedcomponent/1.0.0/1"
        }
    ],
    "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-
english-full-base-x86/xxxx.x.x"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateImageRecipe。

create-image

以下代码示例演示了如何使用 create-image。

Amazon CLI

创建镜像

以下 create-image 示例创建了一个镜像。

```
aws imagebuilder create-image \
    --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/mybasicrecipe/2019.12.03 \
    --infrastructure-configuration-arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/myexampleinfrastructure
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/mybasicrecipe/2019.12.03/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateImage。

create-infrastructure-configuration

以下代码示例演示了如何使用 create-infrastructure-configuration。

Amazon CLI

创建基础结构配置

以下 create-infrastructure-configuration 示例使用 JSON 文件创建了一个基础结构配置。

```
aws imagebuilder create-infrastructure-configuration \
    --cli-input-json file://create-infrastructure-configuration.json
```

的内容create-infrastructure-configuration.json：

```
{
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
        "m5.large", "m5.xlarge"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "securityGroupIds": [
        "sg-a1b2c3d4"
    ],
    "subnetId": "subnet-a1b2c3d4",
    "logging": {
        "s3Logs": {
            "s3BucketName": "bucket-name",
            "s3KeyPrefix": "bucket-path"
        }
    },
    "keyPair": "key-pair-name",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-topic-name"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInfrastructureConfiguration。

`delete-component`

以下代码示例演示了如何使用 `delete-component`。

Amazon CLI

删除组件

以下 delete-component 示例通过指定组件的 ARN 来删除其生成版本。

```
aws imagebuilder delete-component \
    --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/myexamplecomponent/2019.12.02/1
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myexamplecomponent/2019.12.02/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteComponent。

delete-image-pipeline

以下代码示例演示了如何使用 delete-image-pipeline。

Amazon CLI

删除镜像管道

以下 delete-image-pipeline 示例通过指定镜像管道的 ARN 来将其删除。

```
aws imagebuilder delete-image-pipeline \
    --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
my-example-pipeline
```

输出：

```
{
```

```
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteImagePipeline。

delete-image-recipe

以下代码示例演示了如何使用 delete-image-recipe。

Amazon CLI

删除镜像配方

以下 delete-image-recipe 示例通过指定镜像配方的 ARN 来将其删除。

```
aws imagebuilder delete-image-recipe \
    --image-recipe-arn arn:aws:imagebuilder:us-east-1:123456789012:image-recipe/
mybasicrecipe/2019.12.03
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteImageRecipe。

delete-image

以下代码示例演示了如何使用 delete-image。

Amazon CLI

删除镜像

以下 delete-image 示例通过指定镜像生成版本的 ARN 来将其删除。

```
aws imagebuilder delete-image \
    --image-build-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-image/2019.12.02/1
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/mybasicrecipe/2019.12.03/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteImage。

delete-infrastructure-configuration

以下代码示例演示了如何使用 delete-infrastructure-configuration。

Amazon CLI

删除基础结构配置

以下 delete-infrastructure-configuration 示例通过指定镜像管道的 ARN 来将其删除。

```
aws imagebuilder delete-infrastructure-configuration \
    --infrastructure-configuration-arn arn:aws:imagebuilder:us-east-1:123456789012:infrastructure-configuration/myexampleinfrastructure
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInfrastructureConfiguration。

get-component-policy

以下代码示例演示了如何使用 get-component-policy。

Amazon CLI

获取组件策略详细信息

以下 get-component-policy 示例通过指定组件策略的 ARN 来列出其详细信息。

```
aws imagebuilder get-component-policy \
    --component-arn arn:aws:imagebuilder:us-west-2:123456789012:component/my-
example-component/2019.12.03/1
```

输出：

```
{
    "Policy": "{ "Version":"2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetComponent", "imagebuilder:ListComponents" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.03/1" ] } ] }"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的“使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道”<https://docs.aws.amazon.com/imagebuilder/latest/userguide/managing-image-builder-cli.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComponentPolicy。

get-component

以下代码示例演示了如何使用 get-component。

Amazon CLI

获取组件详细信息

以下 get-component 示例通过指定组件的 ARN 来列出其详细信息。

```
aws imagebuilder get-component \
    --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/component-name/1.0.0/1
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "component": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/component-
name/1.0.0/1",
        "name": "component-name",
        "version": "1.0.0",
        "type": "TEST",
        "platform": "Linux",
        "owner": "123456789012",
        "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
 testing document.\nschemaVersion: 1.0\n\nphases:\n  - name: test\n    steps:\n
   - name: HelloWorldStep\n        action: ExecuteBash\n        inputs:\n
 commands:\n            - echo \"Hello World! Test.\"\n",
        "encrypted": true,
        "dateCreated": "2020-01-27T20:43:30.306Z",
        "tags": {}
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComponent。

get-distribution-configuration

以下代码示例演示了如何使用 get-distribution-configuration。

Amazon CLI

获取分发配置的详细信息

以下 get-distribution-configuration 示例通过指定分发配置的 ARN 来获取其详细信息。

```
aws imagebuilder get-distribution-configuration \
    --distribution-configuration-arn arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "distributionConfiguration": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/myexampledistribution",
        "name": "MyExampleDistribution",
        "description": "Copies AMI to eu-west-1 and exports to S3",
        "distributions": [
            {
                "region": "us-west-2",
                "amiDistributionConfiguration": {
                    "name": "Name {{imagebuilder:buildDate}}",
                    "description": "An example image name with parameter
 references",
                    "amiTags": {
                        "KeyName": "{{ssm:parameter_name}}"
                    },
                    "launchPermission": {
                        "userIds": [
                            "123456789012"
                        ]
                    }
                }
            },
            {
                "region": "eu-west-1",
                "amiDistributionConfiguration": {
                    "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                    "amiTags": {
                        "KeyName": "Value"
```

```
                    },
                    "launchPermission": {
                        "userIds": [
                            "123456789012"
                        ]
                    }
                }
            }
        ],
        "dateCreated": "2020-02-19T18:40:10.529Z",
        "tags": {}
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDistributionConfiguration。

get-image-pipeline

以下代码示例演示了如何使用 get-image-pipeline。

Amazon CLI

获取镜像管道详细信息

以下 get-image-pipeline 示例通过指定镜像管道的 ARN 来获取其详细信息。

```
aws imagebuilder get-image-pipeline \
    --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imagePipeline": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
        "name": "MyWindows2016Pipeline",
        "description": "Builds Windows 2016 Images",
```

```
        "platform": "Windows",
        "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
        "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
        "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
        "imageTestsConfiguration": {
            "imageTestsEnabled": true,
            "timeoutMinutes": 60
        },
        "schedule": {
            "scheduleExpression": "cron(0 0 * * SUN)",
            "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
        },
        "status": "ENABLED",
        "dateCreated": "2020-02-19T19:04:01.253Z",
        "dateUpdated": "2020-02-19T19:04:01.253Z",
        "tags": {}
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImagePipeline。

get-image-policy

以下代码示例演示了如何使用 get-image-policy。

Amazon CLI

获取镜像策略详细信息

以下 get-image-policy 示例通过指定镜像策略的 ARN 来列出其详细信息。

```
aws imagebuilder get-image-policy \
    --image-arn arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image/2019.12.03/1
```

输出：

```
{
    "Policy": "{ "Version":"2012-10-17",        "Statement": [ { "Effect": "Allow",
 "Principal": { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetImage",
 "imagebuilder:ListImages" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.03/1" ] } ] }"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImagePolicy。

get-image-recipe-policy

以下代码示例演示了如何使用 get-image-recipe-policy。

Amazon CLI

获取镜像配方策略详细信息

以下 get-image-recipe-policy 示例通过指定镜像配方策略的 ARN 来获取其详细信息。

```
aws imagebuilder get-image-recipe-policy \
    --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-recipe/2019.12.03/1
```

输出：

```
{
    "Policy": "{ "Version":"2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03/1" ] } ] }"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImageRecipePolicy。

get-image

以下代码示例演示了如何使用 get-image。

Amazon CLI

获取镜像详细信息

以下 get-image 示例通过指定镜像的 ARN 来获取其详细信息。

```
aws imagebuilder get-image \
    --image-build-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "image": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1",
        "name": "MyBasicRecipe",
        "version": "2019.12.03/1",
        "platform": "Windows",
        "state": {
            "status": "BUILDING"
        },
        "imageRecipe": {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
            "name": "MyBasicRecipe",
            "description": "This example image recipe creates a Windows 2016
 image.",
            "platform": "Windows",
            "version": "2019.12.03",
            "components": [
                {
                    "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myexamplecomponent/2019.12.02/1"
                },
                {
                    "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myimportedcomponent/1.0.0/1"
                }
```

```
            ],
            "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-
server-2016-english-full-base-x86/2019.12.17/1",
            "dateCreated": "2020-02-14T19:46:16.904Z",
            "tags": {}
        },
        "infrastructureConfiguration": {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/myexampleinfrastructure",
            "name": "MyExampleInfrastructure",
            "description": "An example that will retain instances of failed builds",
            "instanceTypes": [
                "m5.large",
                "m5.xlarge"
            ],
            "instanceProfileName": "EC2InstanceProfileForImageFactory",
            "securityGroupIds": [
                "sg-a1b2c3d4"
            ],
            "subnetId": "subnet-a1b2c3d4",
            "logging": {
                "s3Logs": {
                    "s3BucketName": "bucket-name",
                    "s3KeyPrefix": "bucket-path"
                }
            },
            "keyPair": "Sam",
            "terminateInstanceOnFailure": false,
            "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name",
            "dateCreated": "2020-02-14T21:21:05.098Z",
            "tags": {}
        },
        "imageTestsConfiguration": {
            "imageTestsEnabled": true,
            "timeoutMinutes": 720
        },
        "dateCreated": "2020-02-14T23:14:13.597Z",
        "outputResources": {
            "amis": []
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetImage。

get-infrastructure-configuration

以下代码示例演示了如何使用 get-infrastructure-configuration。

Amazon CLI

获取基础结构配置详细信息

以下 get-infrastructure-configuration 示例通过指定基础结构配置的 ARN 来获取其详细信息。

```
aws imagebuilder get-infrastructure-configuration \
    --infrastructure-configuration-arn arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "infrastructureConfiguration": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/myexampleinfrastructure",
        "name": "MyExampleInfrastructure",
        "description": "An example that will retain instances of failed builds",
        "instanceTypes": [
            "m5.large",
            "m5.xlarge"
        ],
        "instanceProfileName": "EC2InstanceProfileForImageBuilder",
        "securityGroupIds": [
            "sg-a48c95ef"
        ],
        "subnetId": "subnet-a48c95ef",
        "logging": {
            "s3Logs": {
                "s3BucketName": "bucket-name",
                "s3KeyPrefix": "bucket-path"
```

```
            }
        },
        "keyPair": "Name",
        "terminateInstanceOnFailure": false,
        "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name",
        "dateCreated": "2020-02-19T19:11:51.858Z",
        "tags": {}
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInfrastructureConfiguration。

`import-component`

以下代码示例演示了如何使用 `import-component`。

Amazon CLI

导入组件

以下 `import-component` 示例使用 JSON 文件导入先前存在的脚本。

```
aws imagebuilder import-component \
    --cli-input-json file://import-component.json
```

的内容`import-component.json`：

```
{
    "name": "MyImportedComponent",
    "semanticVersion": "1.0.0",
    "description": "An example of how to import a component",
    "changeDescription": "First commit message.",
    "format": "SHELL",
    "platform": "Windows",
    "type": "BUILD",
    "uri": "s3://s3-bucket-name/s3-bucket-path/component.yaml"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "componentBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myimportedcomponent/1.0.0/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportComponent。

`list-component-build-versions`

以下代码示例演示了如何使用 `list-component-build-versions`。

Amazon CLI

列出组件生成版本

以下 `list-component-build-versions` 示例列出具有特定语义版本的组件生成版本。

```
aws imagebuilder list-component-build-versions --component-
version-arn arn:aws:imagebuilder:us-west-2:123456789012:component/
myexamplecomponent/2019.12.02
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
myexamplecomponent/2019.12.02/1",
            "name": "MyExampleComponent",
            "version": "2019.12.02",
            "platform": "Windows",
            "type": "BUILD",
            "owner": "123456789012",
            "description": "An example component that builds, validates and tests an
 image",
```

```
            "changeDescription": "Initial version.",
            "dateCreated": "2020-02-19T18:53:45.940Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListComponentBuildVersions。

`list-components`

以下代码示例演示了如何使用 `list-components`。

Amazon CLI

列出所有组件语义版本

以下 `list-components` 示例列出您有权访问的所有组件语义版本。您可以选择筛选依据，如只列出您拥有的组件、Amazon 拥有的组件或其他账户与您共享的组件。

```
aws imagebuilder list-components
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/component-
name/1.0.0",
            "name": "component-name",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "TEST",
            "owner": "123456789012",
            "dateCreated": "2020-01-27T20:43:30.306Z"
        }
```

```
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListComponents。

`list-distribution-configurations`

以下代码示例演示了如何使用 `list-distribution-configurations`。

Amazon CLI

列出分发配置

以下 `list-distribution-configurations` 示例列出您的所有分发配置。

```
aws imagebuilder list-distribution-configurations
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "distributionConfigurationSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/myexampledistribution",
            "name": "MyExampleDistribution",
            "description": "Copies AMI to eu-west-1 and exports to S3",
            "dateCreated": "2020-02-19T18:40:10.529Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDistributionConfigurations。

list-image-build-versions

以下代码示例演示了如何使用 list-image-build-versions。

Amazon CLI

列出镜像生成版本

以下 list-image-build-versions 示例列出具有某个语义版本的所有镜像生成版本。

```
aws imagebuilder list-image-build-versions \
    --image-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/7",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/7",
            "platform": "Windows",
            "state": {
                "status": "FAILED",
                "reason": "Can't start SSM Automation for arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/mybasicrecipe/2019.12.03/7 during
 building. Parameter \"iamInstanceProfileName\" has a null value."
            },
            "owner": "123456789012",
            "dateCreated": "2020-02-19T18:56:11.511Z",
            "outputResources": {
                "amis": []
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/6",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/6",
            "platform": "Windows",
```

```
            "state": {
                "status": "FAILED",
                "reason": "An internal error has occurred."
            },
            "owner": "123456789012",
            "dateCreated": "2020-02-18T22:49:08.142Z",
            "outputResources": {
                "amis": [
                    {
                        "region": "us-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "MyBasicRecipe 2020-02-18T22-49-38.704Z",
                        "description": "This example image recipe creates a Windows
 2016 image."
                    },
                    {
                        "region": "us-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "Name 2020-02-18T22-49-08.131Z",
                        "description": "Copies AMI to eu-west-2 and exports to S3"
                    },
                    {
                        "region": "eu-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "My 6 image 2020-02-18T22-49-08.131Z",
                        "description": "Copies AMI to eu-west-2 and exports to S3"
                    }
                ]
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/5",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/5",
            "platform": "Windows",
            "state": {
                "status": "AVAILABLE"
            },
            "owner": "123456789012",
            "dateCreated": "2020-02-18T16:51:48.403Z",
            "outputResources": {
                "amis": [
```

```
                    {
                        "region": "us-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "MyBasicRecipe 2020-02-18T16-52-18.965Z",
                        "description": "This example image recipe creates a Windows
 2016 image."
                    }
                ]
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/4",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/4",
            "platform": "Windows",
            "state": {
                "status": "AVAILABLE"
            },
            "owner": "123456789012",
            "dateCreated": "2020-02-18T16:50:01.827Z",
            "outputResources": {
                "amis": [
                    {
                        "region": "us-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "MyBasicRecipe 2020-02-18T16-50-32.280Z",
                        "description": "This example image recipe creates a Windows
 2016 image."
                    }
                ]
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/3",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/3",
            "platform": "Windows",
            "state": {
                "status": "AVAILABLE"
            },
```

```
            "owner": "123456789012",
            "dateCreated": "2020-02-14T23:14:13.597Z",
            "outputResources": {
                "amis": [
                    {
                        "region": "us-west-2",
                        "image": "ami-a1b2c3d4567890ab",
                        "name": "MyBasicRecipe 2020-02-14T23-14-44.243Z",
                        "description": "This example image recipe creates a Windows
 2016 image."
                    }
                ]
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/2",
            "name": "MyBasicRecipe",
            "version": "2019.12.03/2",
            "platform": "Windows",
            "state": {
                "status": "FAILED",
                "reason": "SSM execution 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
 failed with status = 'Failed' and failure message = 'Step fails when it is
 verifying the command has completed. Command a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
 returns unexpected invocation result: \n{Status=[Failed], ResponseCode=[1],
 Output=[\n----------ERROR-------\nfailed to run commands: exit status 1],
 OutputPayload=[{\"Status\":\"Failed\",\"ResponseCode\":1,\"Output\":\"\
\n----------ERROR-------\\nfailed to run commands: exit status 1\",\"CommandId\":
\"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111\"}], CommandId=[a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111]}. Please refer to Automation Service Troubleshooting Guide for more
 diagnosis details.'"
            },
            "owner": "123456789012",
            "dateCreated": "2020-02-14T22:57:42.593Z",
            "outputResources": {
                "amis": []
            },
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImageBuildVersions。

`list-image-pipeline-images`

以下代码示例演示了如何使用 `list-image-pipeline-images`。

Amazon CLI

列出镜像管道创建的镜像

以下 `list-image-pipeline-images` 示例列出由某个特定镜像管道创建的所有镜像。

```
aws imagebuilder list-image-pipeline-images \
    --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imagePipelineList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
            "name": "MyWindows2016Pipeline",
            "description": "Builds Windows 2016 Images",
            "platform": "Windows",
            "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
recipe/mybasicrecipe/2019.12.03",
            "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
            "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
            "imageTestsConfiguration": {
                "imageTestsEnabled": true,
                "timeoutMinutes": 60
            },
            "schedule": {
                "scheduleExpression": "cron(0 0 * * SUN)",
```

```
                "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
            },
            "status": "ENABLED",
            "dateCreated": "2020-02-19T19:04:01.253Z",
            "dateUpdated": "2020-02-19T19:04:01.253Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/sam",
            "name": "PipelineName",
            "platform": "Linux",
            "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
recipe/recipe-name-a1b2c3d45678/1.0.0",
            "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/infrastructureconfiguration-name-
a1b2c3d45678",
            "imageTestsConfiguration": {
                "imageTestsEnabled": true,
                "timeoutMinutes": 720
            },
            "status": "ENABLED",
            "dateCreated": "2019-12-16T18:19:02.068Z",
            "dateUpdated": "2019-12-16T18:19:02.068Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImagePipelineImages。

list-image-recipes

以下代码示例演示了如何使用 `list-image-recipes`。

Amazon CLI

列出镜像配方

以下 `list-image-recipes` 示例列出您的所有镜像配方。

```
aws imagebuilder list-image-recipes
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageRecipeSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
            "name": "MyBasicRecipe",
            "platform": "Windows",
            "owner": "123456789012",
            "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-
server-2016-english-full-base-x86/2019.x.x",
            "dateCreated": "2020-02-19T18:54:25.975Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/recipe-
name-a1b2c3d45678/1.0.0",
            "name": "recipe-name-a1b2c3d45678",
            "platform": "Linux",
            "owner": "123456789012",
            "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2-
x86/2019.11.21",
            "dateCreated": "2019-12-16T18:19:00.120Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImageRecipes。

`list-images`

以下代码示例演示了如何使用 `list-images`。

Amazon CLI

列出镜像

以下 `list-images` 示例列出您有权访问的所有语义版本。

```
aws imagebuilder list-images
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03",
            "name": "MyBasicRecipe",
            "version": "2019.12.03",
            "platform": "Windows",
            "owner": "123456789012",
            "dateCreated": "2020-02-14T21:29:18.810Z"
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListImages。

`list-infrastructure-configurations`

以下代码示例演示了如何使用 `list-infrastructure-configurations`。

Amazon CLI

列出基础结构配置

以下 `list-infrastructure-configurations` 示例列出您的所有基础结构配置。

```
aws imagebuilder list-infrastructure-configurations
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "infrastructureConfigurationSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/myexampleinfrastructure",
            "name": "MyExampleInfrastructure",
            "description": "An example that will retain instances of failed builds",
            "dateCreated": "2020-02-19T19:11:51.858Z",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/infrastructureconfiguration-name-a1b2c3d45678",
            "name": "infrastructureConfiguration-name-a1b2c3d45678",
            "dateCreated": "2019-12-16T18:19:01.038Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInfrastructureConfigurations。

list-tags-for-resource

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出特定资源的标签

以下 `list-tags-for-resource` 示例列出某个特定资源的所有标签。

```
aws imagebuilder list-tags-for-resource \
    --resource-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

输出：

```
{
    "tags": {
        "KeyName": "KeyValue"
    }
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-component-policy

以下代码示例演示了如何使用 `put-component-policy`。

Amazon CLI

将资源策略应用于组件

以下 `put-component-policy` 命令将资源策略应用于一个生成组件，以启用跨账户的生成组件共享。我们建议您使用 RAM CLI 命令 `create-resource-share`。如果您使用 EC2 Image Builder CLI 命令 `put-component-policy`，则还须使用 RAM CLI 命令 `promote-resource-share-create-from-policy`，这样资源才能被所有共享主体看到。

```
aws imagebuilder put-component-policy \
    --component-arn arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/2019.12.02/1 \
    --policy '{ "Version":"2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetComponent", "imagebuilder:ListComponents" ],
```

```
 "Resource": [ "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/2019.12.02/1" ] } ] }'
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/2019.12.02/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutComponentPolicy。

put-image-policy

以下代码示例演示了如何使用 put-image-policy。

Amazon CLI

将资源策略应用于镜像

以下 put-image-policy 命令将资源策略应用于镜像，以启用跨账户的镜像共享。我们建议您使用 RAM CLI 命令 create-resource-share。如果使用 EC2 Image Builder CLI 命令 put-image-policy，则还须使用 RAM CLI 命令 promote-resource-share-create-from-policy，这样资源才能被所有共享主体看到。

```
aws imagebuilder put-image-policy \
    --image-arn arn:aws:imagebuilder:us-west-2:123456789012:image/example-
image/2019.12.02/1 \
    --policy '{ "Version":"2012-10-17",        "Statement": [ { "Effect": "Allow",
 "Principal": { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetImage",
 "imagebuilder:ListImages" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:image/example-image/2019.12.02/1" ] } ] }'
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
    "imageArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/example-
image/2019.12.02/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutImagePolicy。

put-image-recipe-policy

以下代码示例演示了如何使用 put-image-recipe-policy。

Amazon CLI

将资源策略应用于镜像配方

以下 put-image-recipe-policy 命令将资源策略应用于镜像配方，以启用跨账户的镜像配方共享。我们建议您使用 RAM CLI 命令 create-resource-share。如果您使用 EC2 Image Builder CLI 命令 put-image-recipe-policy，则还须使用 RAM CLI 命令 promote-resource-share-create-from-policy，这样资源才能被所有共享主体看到。

```
aws imagebuilder put-image-recipe-policy \
    --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
example-image-recipe/2019.12.02 \
    --policy '{ "Version":"2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/example-image-
recipe/2019.12.02" ] } ] }'
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
example-image-recipe/2019.12.02/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutImageRecipePolicy。

start-image-pipeline-execution

以下代码示例演示了如何使用 start-image-pipeline-execution。

Amazon CLI

手动启动镜像管道

以下 start-image-pipeline-execution 示例手动启动了一个镜像管道。

```
aws imagebuilder start-image-pipeline-execution \
    --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartImagePipelineExecution。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例使用 JSON 文件为 EC2 Image Builder 添加并标记资源。

```
aws imagebuilder tag-resource \
```

```
    --cli-input-json file://tag-resource.json
```

的内容tag-resource.json：

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
    "tags": {
        "KeyName: "KeyValue"
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例使用 JSON 文件从资源中删除标签。

```
aws imagebuilder untag-resource \
    --cli-input-json file://tag-resource.json
```

的内容untag-resource.json：

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
    "tagKeys": [
        "KeyName"
    ]
```

```
}
```

此命令不生成任何输出。

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-distribution-configuration

以下代码示例演示了如何使用 update-distribution-configuration。

Amazon CLI

更新分发配置

以下 update-distribution-configuration 示例使用 JSON 文件更新了分发配置。

```
aws imagebuilder update-distribution-configuration \
    --cli-input-json file://update-distribution-configuration.json
```

的内容update-distribution-configuration.json：

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
    "description": "Copies AMI to eu-west-2 and exports to S3",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter references"
            }
        },
        {
            "region": "eu-west-2",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}"
            }
```

```
        }
    ]
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDistributionConfiguration。

update-image-pipeline

以下代码示例演示了如何使用 update-image-pipeline。

Amazon CLI

更新镜像管道

以下 update-image-pipeline 示例使用 JSON 文件更新了镜像管道。

```
aws imagebuilder update-image-pipeline \
    --cli-input-json file://update-image-pipeline.json
```

的内容update-image-pipeline.json：

```
{
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
```

```
        "timeoutMinutes": 120
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * MON)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "DISABLED"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateImagePipeline。

update-infrastructure-configuration

以下代码示例演示了如何使用 update-infrastructure-configuration。

Amazon CLI

更新基础结构配置

以下 update-infrastructure-configuration 示例使用 JSON 文件更新了基础结构配置。

```
aws imagebuilder update-infrastructure-configuration \
    --cli-input-json file:/update-infrastructure-configuration.json
```

的内容update-infrastructure-configuration.json：

```
{
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
    "description": "An example that will terminate instances of failed builds",
    "instanceTypes": [
```

```
        "m5.large", "m5.2xlarge"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageFactory",
    "securityGroupIds": [
        "sg-a48c95ef"
    ],
    "subnetId": "subnet-a48c95ef",
    "logging": {
        "s3Logs": {
            "s3BucketName": "bucket-name",
            "s3KeyPrefix": "bucket-path"
        }
    },
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name"
}
```

输出：

```
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《EC2 Image Builder 用户指南》中的使用 Amazon CLI 设置和管理 EC2 Image Builder 镜像管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateInfrastructureConfiguration。

使用 Amazon CLI 的 Incident Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Incident Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-replication-set`

以下代码示例演示了如何使用 `create-replication-set`。

Amazon CLI

创建复制集

以下 `create-replication-set` 示例创建复制集，以便 Incident Manager 复制和加密您 Amazon Web Services 账户中的数据。此示例在创建复制集时使用 us-east-1 和 us-east-2 区域。

```
aws ssm-incidents create-replication-set \
    --regions '{"us-east-1": {"sseKmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"}, "us-east-2":
 {"sseKmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"}}'
```

输出：

```
{
    "replicationSetArns": [
        "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-
bb3f-413c-08df53673b57"
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateReplicationSet。

`create-response-plan`

以下代码示例演示了如何使用 `create-response-plan`。

Amazon CLI

创建响应计划

以下 `create-response-plan` 示例创建包含指定详细信息的响应计划。

```
aws ssm-incidents create-response-plan \
    --chat-channel '{"chatbotSns": ["arn:aws:sns:us-
east-1:111122223333:Standard_User"]}' \
    --display-name "Example response plan" \
    --incident-template '{"impact": 5, "title": "example-incident"}' \
    --name "example-response" \
    --actions '[{"ssmAutomation": {"documentName": "AWSIncidents-
CriticalIncidentRunbookTemplate", "documentVersion": "$DEFAULT",
 "roleArn": "arn:aws:iam::111122223333:role/aws-service-role/ssm-
incidents.amazonaws.com/AWSServiceRoleForIncidentManager", "targetAccount":
 "RESPONSE_PLAN_OWNER_ACCOUNT"}}]' \
    --engagements '["arn:aws:ssm-contacts:us-east-1:111122223333:contact/example"]'
```

输出：

```
{
    "arn": "arn:aws:ssm-incidents::111122223333:response-plan/example-response"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResponsePlan。

create-timeline-event

以下代码示例演示了如何使用 create-timeline-event。

Amazon CLI

示例 1：创建自定义时间线事件

以下 create-timeline-event 示例在指定事件的指定时间创建一个自定义时间线事件。

```
aws ssm-incidents create-timeline-event \
    --event-data "\"example timeline event\"" \
    --event-time 2022-10-01T20:30:00.000 \
    --event-type "Custom Event" \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
```

输出：

```
{
    "eventId": "c0bcc885-a41d-eb01-b4ab-9d2deEXAMPLE",
    "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
}
```

示例 2：创建带有事件备注的时间线事件

以下 create-timeline-event 示例创建一个在“事件备注”面板中列出的时间线事件。

```
aws ssm-incidents create-timeline-event \
     --event-data "\"New Note\"" \
     --event-type "Note" \
     --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Test/6cc46130-ca6c-3b38-68f1-f6abeEXAMPLE" \
     --event-time 2023-06-20T12:06:00.000 \
     --event-references '[{"resource":"arn:aws:ssm-incidents::111122223333:incident-
record/Test/6cc46130-ca6c-3b38-68f1-f6abeEXAMPLE"}]'
```

输出：

```
{
    "eventId": "a41dc885-c0bc-b4ab-eb01-de9d2EXAMPLE",
    "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTimelineEvent。

delete-incident-record

以下代码示例演示了如何使用 delete-incident-record。

Amazon CLI

删除事件记录

以下 delete-incident-record 示例删除指定的事件记录。

```
aws ssm-incidents delete-incident-record \
```

```
    --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件跟踪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIncidentRecord。

delete-replication-set

以下代码示例演示了如何使用 delete-replication-set。

Amazon CLI

删除复制集

以下 delete-replication-set 示例从您的 Amazon Web Services 账户中删除复制集。删除复制集也会删除所有 Incident Manager 数据。此操作无法撤消。

```
aws ssm-incidents delete-replication-set \
    --arn "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-bb3f-413c-08df53673b57"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteReplicationSet。

delete-resource-policy

以下代码示例演示了如何使用 delete-resource-policy。

Amazon CLI

删除资源策略

以下 delete-resource-policy 示例从响应计划中删除资源策略。这将撤消与之共享响应计划的主体或组织的访问权限。

```
aws ssm-incidents delete-resource-policy \
    --policy-id "be8b57191f0371f1c6827341aa3f0a03" \
```

```
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-
Response-Plan"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的使用共享的联系人和响应计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourcePolicy。

delete-response-plan

以下代码示例演示了如何使用 delete-response-plan。

Amazon CLI

删除响应计划

以下 delete-response-plan 示例删除指定的响应计划。

```
aws ssm-incidents delete-response-plan \
    --arn "arn:aws:ssm-incidents::111122223333:response-plan/example-response"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResponsePlan。

delete-timeline-event

以下代码示例演示了如何使用 delete-timeline-event。

Amazon CLI

删除时间线事件

以下 delete-timeline-event 示例从指定的事件记录中删除自定义时间线事件。

```
aws ssm-incidents delete-timeline-event \
    --event-id "c0bcc885-a41d-eb01-b4ab-9d2de193643c" \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTimelineEvent。

get-incident-record

以下代码示例演示了如何使用 get-incident-record。

Amazon CLI

获取事件记录

以下 get-incident-record 示例获取指定事件记录的详细信息。

```
aws ssm-incidents get-incident-record \
    --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

输出：

```
{
    "incidentRecord": {
        "arn": "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308",
        "automationExecutions": [],
        "creationTime": "2021-05-21T18:16:57.579000+00:00",
        "dedupeString": "c4bcc812-85e7-938d-2b78-17181176ee1a",
        "impact": 5,
        "incidentRecordSource": {
            "createdBy": "arn:aws:iam::111122223333:user/draliatp",
            "invokedBy": "arn:aws:iam::111122223333:user/draliatp",
            "source": "aws.ssm-incidents.custom"
        },
        "lastModifiedBy": "arn:aws:iam::111122223333:user/draliatp",
        "lastModifiedTime": "2021-05-21T18:16:59.149000+00:00",
        "notificationTargets": [],
        "status": "OPEN",
        "title": "Example-Incident"
    }
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIncidentRecord。

get-replication-set

以下代码示例演示了如何使用 get-replication-set。

Amazon CLI

获取复制集

以下 get-replication-set 示例获取复制集的详细信息，以便 Incident Manager 复制和加密您 Amazon Web Services 账户中的数据。

```
aws ssm-incidents get-replication-set \
    --arn "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-bb3f-413c-08df53673b57"
```

输出：

```
{
    "replicationSet": {
        "createdBy": "arn:aws:sts::111122223333:assumed-role/Admin/username",
        "createdTime": "2021-05-14T17:57:22.010000+00:00",
        "deletionProtected": false,
        "lastModifiedBy": "arn:aws:sts::111122223333:assumed-role/Admin/username",
        "lastModifiedTime": "2021-05-14T17:57:22.010000+00:00",
        "regionMap": {
            "us-east-1": {
                "sseKmsKeyId": "DefaultKey",
                "status": "ACTIVE"
            },
            "us-east-2": {
                "sseKmsKeyId": "DefaultKey",
                "status": "ACTIVE",
                "statusMessage": "Tagging inaccessible"
            }
        },
        "status": "ACTIVE"
    }
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReplicationSet。

get-resource-policies

以下代码示例演示了如何使用 get-resource-policies。

Amazon CLI

列出响应计划的资源策略

以下 command-name 示例列出与指定响应计划关联的资源策略。

```
aws ssm-incidents get-resource-policies \
--resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

输出：

```
{
    "resourcePolicies": [
        {
            "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid
\":\"d901b37a-dbb0-458a-8842-75575c464219-external-principals\",\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::222233334444:root\"},\"Action
\":[\"ssm-incidents:GetResponsePlan\",\"ssm-incidents:StartIncident\",\"ssm-
incidents:UpdateIncidentRecord\",\"ssm-incidents:GetIncidentRecord\",\"ssm-
incidents:CreateTimelineEvent\",\"ssm-incidents:UpdateTimelineEvent\",\"ssm-
incidents:GetTimelineEvent\",\"ssm-incidents:ListTimelineEvents\",\"ssm-
incidents:UpdateRelatedItems\",\"ssm-incidents:ListRelatedItems\"],\"Resource\":
[\"arn:aws:ssm-incidents:*:111122223333:response-plan/Example-Response-Plan\",
\"arn:aws:ssm-incidents:*:111122223333:incident-record/Example-Response-Plan/*
\"]}]}",
            "policyId": "be8b57191f0371f1c6827341aa3f0a03",
            "ramResourceShareRegion": "us-east-1"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用共享的联系人和响应计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourcePolicies。

get-response-plan

以下代码示例演示了如何使用 get-response-plan。

Amazon CLI

获取响应计划的详细信息

以下 command-name 示例获取有关您 Amazon 账户中指定响应计划的详细信息。

```
aws ssm-incidents get-response-plan \
    --arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

输出：

```
{
    "actions": [
        {
            "ssmAutomation": {
                "documentName": "AWSIncidents-CriticalIncidentRunbookTemplate",
                "documentVersion": "$DEFAULT",
                "roleArn": "arn:aws:iam::111122223333:role/aws-service-role/ssm-
incidents.amazonaws.com/AWSServiceRoleForIncidentManager",
                "targetAccount": "RESPONSE_PLAN_OWNER_ACCOUNT"
            }
        }
    ],
    "arn": "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-
Plan",
    "chatChannel": {
        "chatbotSns": [
            "arn:aws:sns:us-east-1:111122223333:Standard_User"
        ]
    },
    "displayName": "Example response plan",
    "engagements": [
        "arn:aws:ssm-contacts:us-east-1:111122223333:contact/example"
    ],
    "incidentTemplate": {
        "impact": 5,
        "title": "Example-Incident"
    },
    "name": "Example-Response-Plan"
```

```
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResponsePlan。

`get-timeline-event`

以下代码示例演示了如何使用 `get-timeline-event`。

Amazon CLI

获取时间线事件的详细信息

以下 `get-timeline-event` 示例返回指定时间线事件的详细信息。

```
aws ssm-incidents get-timeline-event \
    --event-id 20bcc812-8a94-4cd7-520c-0ff742111424 \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

输出：

```
{
    "event": {
        "eventData": "\"Incident Started\"",
        "eventId": "20bcc812-8a94-4cd7-520c-0ff742111424",
        "eventTime": "2021-05-21T18:16:57+00:00",
        "eventType": "Custom Event",
        "eventUpdatedTime": "2021-05-21T18:16:59.944000+00:00",
        "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
    }
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTimelineEvent。

`list-incident-records`

以下代码示例演示了如何使用 `list-incident-records`。

Amazon CLI

列出事件记录

以下 command-name 示例列出您 Amazon Web Services 账户中的事件记录。

```
aws ssm-incidents list-incident-records
```

输出：

```
{
    "incidentRecordSummaries": [
        {
            "arn": "arn:aws:ssm-incidents::111122223333:incident-record/Example-
Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308",
            "creationTime": "2021-05-21T18:16:57.579000+00:00",
            "impact": 5,
            "incidentRecordSource": {
                "createdBy": "arn:aws:iam::111122223333:user/draliatp",
                "invokedBy": "arn:aws:iam::111122223333:user/draliatp",
                "source": "aws.ssm-incidents.custom"
            },
            "status": "OPEN",
            "title": "Example-Incident"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIncidentRecords。

list-related-items

以下代码示例演示了如何使用 list-related-items。

Amazon CLI

列出相关项目

以下 list-related-items 示例列出指定事件的相关项目。

```
aws ssm-incidents list-related-items \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

输出：

```
{
    "relatedItems": [
        {
            "identifier": {
                "type": "OTHER",
                "value": {
                    "url": "https://console.aws.amazon.com/systems-manager/opsitems/
oi-8ef82158e190/workbench?region=us-east-1"
                }
            },
            "title": "Example related item"
        },
        {
            "identifier": {
                "type": "PARENT",
                "value": {
                    "arn": "arn:aws:ssm:us-east-1:111122223333:opsitem/
oi-8084126392ac"
                }
            },
            "title": "parentItem"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRelatedItems。

list-replication-sets

以下代码示例演示了如何使用 list-replication-sets。

Amazon CLI

列出复制集

以下 `list-replication-set` 示例列出复制集，以便 Incident Manager 复制和加密您 Amazon 账户中的数据。

```
aws ssm-incidents list-replication-sets
```

输出：

```
{
    "replicationSetArns": [
        "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-
bb3f-413c-08df53673b57"
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListReplicationSets。

`list-response-plans`

以下代码示例演示了如何使用 `list-response-plans`。

Amazon CLI

列出可用的响应计划

以下 `list-response-plans` 示例列出您 Amazon Web Services 账户中可用的响应计划。

```
aws ssm-incidents list-response-plans
```

输出：

```
{
    "responsePlanSummaries": [
        {
            "arn": "arn:aws:ssm-incidents::111122223333:response-plan/Example-
Response-Plan",
            "displayName": "Example response plan",
            "name": "Example-Response-Plan"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResponsePlans。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出响应计划的标签

以下 list-tags-for-resource 示例列出与指定响应计划关联的标签。

```
aws ssm-incidents list-tags-for-resource \
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

输出：

```
{
    "tags": {
    "group1": "1"
    }
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-timeline-events

以下代码示例演示了如何使用 list-timeline-events。

Amazon CLI

列出事件的时间线事件

以下 command-name 示例列出指定事件的时间线事件。

```
aws ssm-incidents list-timeline-events \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

输出：

```
{
    "eventSummaries": [
        {
            "eventId": "8cbcc889-35e1-a42d-2429-d6f100799915",
            "eventTime": "2021-05-21T22:36:13.766000+00:00",
            "eventType": "SSM Incident Record Update",
            "eventUpdatedTime": "2021-05-21T22:36:13.766000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        },
        {
            "eventId": "a2bcc825-aab5-1787-c605-f9bb2640d85b",
            "eventTime": "2021-05-21T18:58:46.443000+00:00",
            "eventType": "SSM Incident Record Update",
            "eventUpdatedTime": "2021-05-21T18:58:46.443000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        },
        {
            "eventId": "5abcc812-89c0-b0a8-9437-1c74223d4685",
            "eventTime": "2021-05-21T18:16:59.149000+00:00",
            "eventType": "SSM Incident Record Update",
            "eventUpdatedTime": "2021-05-21T18:16:59.149000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        },
        {
            "eventId": "06bcc812-8820-405e-4065-8d2b14d29b92",
            "eventTime": "2021-05-21T18:16:58+00:00",
            "eventType": "SSM Automation Execution Start Failure for Incident",
            "eventUpdatedTime": "2021-05-21T18:16:58.689000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        },
        {
```

```
            "eventId": "20bcc812-8a94-4cd7-520c-0ff742111424",
            "eventTime": "2021-05-21T18:16:57+00:00",
            "eventType": "Custom Event",
            "eventUpdatedTime": "2021-05-21T18:16:59.944000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        },
        {
            "eventId": "c0bcc885-a41d-eb01-b4ab-9d2de193643c",
            "eventTime": "2020-10-01T20:30:00+00:00",
            "eventType": "Custom Event",
            "eventUpdatedTime": "2021-05-21T22:28:26.299000+00:00",
            "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTimelineEvents。

put-resource-policy

以下代码示例演示了如何使用 put-resource-policy。

Amazon CLI

分享响应计划和事件

以下 command-name 示例在 Example-Response-Plan 中添加一个资源策略，与指定的主体共享响应计划和关联事件。

```
aws ssm-incidents put-resource-policy \
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-
Response-Plan" \
    --policy "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":
\"ExampleResourcePolciy\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":
\"arn:aws:iam::222233334444:root\"},\"Action\":[\"ssm-incidents:GetResponsePlan
\",\"ssm-incidents:StartIncident\",\"ssm-incidents:UpdateIncidentRecord
\",\"ssm-incidents:GetIncidentRecord\",\"ssm-incidents:CreateTimelineEvent
\",\"ssm-incidents:UpdateTimelineEvent\",\"ssm-incidents:GetTimelineEvent
\",\"ssm-incidents:ListTimelineEvents\",\"ssm-incidents:UpdateRelatedItems
```

```
\",\"ssm-incidents:ListRelatedItems\"],\"Resource\":[\"arn:aws:ssm-
incidents:*:111122223333:response-plan/Example-Response-Plan\",\"arn:aws:ssm-
incidents:*:111122223333:incident-record/Example-Response-Plan/*\"]}]}"
```

输出：

```
{
    "policyId": "be8b57191f0371f1c6827341aa3f0a03"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用共享的联系人和响应计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutResourcePolicy。

start-incident

以下代码示例演示了如何使用 start-incident。

Amazon CLI

启动事件

以下 start-incident 示例使用指定的响应计划启动一个事件。

```
aws ssm-incidents start-incident \
    --response-plan-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-
Response-Plan"
```

输出：

```
{
    "incidentRecordArn": "arn:aws:ssm-incidents::682428703967:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的事件创建。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartIncident。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记响应计划

以下 tag-resource 示例使用提供的标签键值对标记指定的响应计划。

```
aws ssm-incidents tag-resource \
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan" \
    --tags '{"group1":"1"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除响应计划的标签

以下 untag-resource 示例移除响应计划的指定标签。

```
aws ssm-incidents untag-resource \
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan" \
    --tag-keys '["group1"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-deletion-protection

以下代码示例演示了如何使用 update-deletion-protection。

Amazon CLI

更新复制集删除保护

以下 update-deletion-protection 示例更新您账户中的删除保护，防止您删除复制集中的最后一个区域。

```
aws ssm-incidents update-deletion-protection \
    --arn "arn:aws:ssm-incidents::111122223333:replication-set/
a2bcc5c9-0f53-8047-7fef-c20749989b40" \
    --deletion-protected
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeletionProtection。

update-incident-record

以下代码示例演示了如何使用 update-incident-record。

Amazon CLI

更新事件记录

以下 command-name 示例解决指定的事件。

```
aws ssm-incidents update-incident-record \
    --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-
Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \
    --status "RESOLVED"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIncidentRecord。

update-related-items

以下代码示例演示了如何使用 update-related-items。

Amazon CLI

更新与项目相关的事件

以下 update-related-item 示例从指定的事件记录中移除相关项目。

```
aws ssm-incidents update-related-items \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \
    --related-items-update '{"itemToRemove": {"type": "OTHER", "value": {"url":
 "https://console.aws.amazon.com/systems-manager/opsitems/oi-8ef82158e190/workbench?
region=us-east-1"}}}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRelatedItems。

update-replication-set

以下代码示例演示了如何使用 update-replication-set。

Amazon CLI

更新复制集

以下 command-name 示例从复制集中删除 us-east-2 区域。

```
aws ssm-incidents update-replication-set \
    --arn "arn:aws:ssm-incidents::111122223333:replication-set/
a2bcc5c9-0f53-8047-7fef-c20749989b40" \
    --actions '[{"deleteRegionAction": {"regionName": "us-east-2"}}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的使用 Incident Manager 复制集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateReplicationSet。

update-response-plan

以下代码示例演示了如何使用 update-response-plan。

Amazon CLI

更新响应计划

以下 update-response-plan 示例移除指定响应计划的聊天频道。

```
aws ssm-incidents update-response-plan \
    --arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
 \
    --chat-channel '{"empty":{}}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件准备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResponsePlan。

update-timeline-event

以下代码示例演示了如何使用 update-timeline-event。

Amazon CLI

更新时间线事件

以下 update-timeline-event 示例更新事件发生的时间。

```
aws ssm-incidents update-timeline-event \
    --event-id 20bcc812-8a94-4cd7-520c-0ff742111424 \
    --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \
    --event-time "2021-05-21T18:10:57+00:00"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的事件详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTimelineEvent。

使用 Amazon CLI 的 Incident Manager 联系人示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Incident Manager 联系人结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-page`

以下代码示例演示了如何使用 `accept-page`。

Amazon CLI

在参与期间接受页面

以下 `accept-page` 示例使用发送到联系频道的接受代码来接受页面。

```
aws ssm-contacts accept-page \
    --page-id "arn:aws:ssm-contacts:us-east-2:682428703967:page/
akuam/94ea0c7b-56d9-46c3-b84a-a37c8b067ad3" \
    --accept-type READ \
    --accept-code 425440
```

此命令不生成任何输出

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptPage。

`activate-contact-channel`

以下代码示例演示了如何使用 `activate-contact-channel`。

Amazon CLI

激活联系人的联系频道

以下 activate-contact-channel 示例激活联系频道并将其作为事件的一部分使用。

```
aws ssm-contacts activate-contact-channel \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d" \
    --activation-code "466136"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ActivateContactChannel。

command-name

以下代码示例演示了如何使用 command-name。

Amazon CLI

删除联系人

以下 command-name 示例删除一个联系人。任何升级计划都将无法再联系到该联系人。

```
aws ssm-contacts delete-contact \
    --contact-id "arn:aws:ssm-contacts:us-east-1:682428703967:contact/alejr"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CommandName。

create-contact-channel

以下代码示例演示了如何使用 create-contact-channel。

Amazon CLI

创建联系频道

为联系人 Akua Mansa 创建短信类型的联系频道。可以创建短信、电子邮件或语音类型的联系频道。

```
aws ssm-contacts create-contact-channel \
    --contact-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \
    --name "akuas sms-test" \
    --type SMS \
    --delivery-address '{"SimpleAddress": "+15005550199"}'
```

输出：

```
{
    "ContactChannelArn": "arn:aws:ssm-contacts:us-east-1:111122223333:contact-
channel/akuam/02f506b9-ea5d-4764-af89-2daa793ff024"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateContactChannel。

create-contact

以下代码示例演示了如何使用 create-contact。

Amazon CLI

创建联系人

以下 create-contact 示例使用空白计划在您的环境中创建联系人。创建联系频道后，即可更新计划。使用带有此命令输出 ARN 的“create-contact-channel”命令。为该联系人创建联系频道后，使用“update-contact”更新计划。

```
aws ssm-contacts create-contact \
    --alias "akuam" \
    --display-name "Akua Mansa" \
    --type PERSONAL \
    --plan '{"Stages": []}'
```

输出：

```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateContact。

deactivate-contact-channel

以下代码示例演示了如何使用 deactivate-contact-channel。

Amazon CLI

停用联系频道

以下 deactivate-contact-channel 示例停用一个联系频道。停用联系频道意味着在事件发生期间将不再寻呼该联系频道。您也可以随时使用 activate-contact-channel 命令重新激活联系频道。

```
aws ssm-contacts deactivate-contact-channel \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeactivateContactChannel。

delete-contact-channel

以下代码示例演示了如何使用 delete-contact-channel。

Amazon CLI

删除联系频道

以下 delete-contact-channel 示例删除一个联系频道。删除联系频道可确保在事件发生期间不会寻呼该联系频道。

```
aws ssm-contacts delete-contact-channel \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact-channel/akuam/13149bad-52ee-45ea-ae1e-45857f78f9b2"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteContactChannel。

delete-contact

以下代码示例演示了如何使用 delete-contact。

Amazon CLI

删除联系人

以下 delete-contact 示例删除一个联系人。任何升级计划都将无法再联系到该联系人。

```
aws ssm-contacts delete-contact \
    --contact-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact/alejr"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteContact。

describe-engagement

以下代码示例演示了如何使用 describe-engagement。

Amazon CLI

描述参与的详细信息

以下 describe-engagement 示例列出联系人或升级计划参与的详细信息。主题和内容将发送到联系频道。

```
aws ssm-contacts describe-engagement \
    --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"
```

输出：

```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
    "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356",
    "Sender": "cli",
    "Subject": "cli-test",
    "Content": "Testing engagements via CLI",
    "PublicSubject": "cli-test",
    "PublicContent": "Testing engagements va CLI",
    "StartTime": "2021-05-18T18:25:41.151000+00:00"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngagement。

describe-page

以下代码示例演示了如何使用 describe-page。

Amazon CLI

列出联系频道页面的详细信息

以下 describe-page 示例列出联系频道页面的详细信息。该页面将包括页面提供的主题和内容。

```
aws ssm-contacts describe-page \
    --page-id "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/ad0052bd-
e606-498a-861b-25726292eb93"
```

输出：

```
{
    "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/ad0052bd-
e606-498a-861b-25726292eb93",
    "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
akuam/78a29753-3674-4ac5-9f83-0468563567f0",
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",
    "Sender": "cli",
    "Subject": "cli-test",
    "Content": "Testing engagements via CLI",
```

```
    "PublicSubject": "cli-test",
    "PublicContent": "Testing engagements va CLI",
    "SentTime": "2021-05-18T18:43:29.301000+00:00",
    "ReadTime": "2021-05-18T18:43:55.708000+00:00",
    "DeliveryTime": "2021-05-18T18:43:55.265000+00:00"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePage。

get-contact-channel

以下代码示例演示了如何使用 get-contact-channel。

Amazon CLI

列出联系频道的详细信息

以下 get-contact-channel 示例列出联系频道的详细信息。

```
aws ssm-contacts get-contact-channel \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d"
```

输出：

```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",
    "ContactChannelArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
    "Name": "akuas sms",
    "Type": "SMS",
    "DeliveryAddress": {
        "SimpleAddress": "+15005550199"
    },
    "ActivationStatus": "ACTIVATED"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContactChannel。

get-contact-policy

以下代码示例演示了如何使用 get-contact-policy。

Amazon CLI

列出联系人的资源策略

以下 get-contact-policy 示例列出与指定联系人关联的资源策略。

```
aws ssm-contacts get-contact-policy \
    --contact-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam"
```

输出：

```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam",
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":
\"SharePolicyForDocumentationDralia\",\"Effect\":\"Allow\",\"Principal\":
{\"AWS\":\"222233334444\"},\"Action\":[\"ssm-contacts:GetContact\",\"ssm-
contacts:StartEngagement\",\"ssm-contacts:DescribeEngagement\",\"ssm-
contacts:ListPagesByEngagement\",\"ssm-contacts:StopEngagement\"],\"Resource
\":[\"arn:aws:ssm-contacts:*:111122223333:contact/akuam\",\"arn:aws:ssm-
contacts:*:111122223333:engagement/akuam/*\"]}]}"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的使用共享的联系人和响应计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContactPolicy。

get-contact

以下代码示例演示了如何使用 get-contact。

Amazon CLI

示例 1：描述联系人计划

以下 get-contact 示例描述一个联系人。

```
aws ssm-contacts get-contact \
    --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

输出：


```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",
    "Alias": "akuam",
    "DisplayName": "Akua Mansa",
    "Type": "PERSONAL",
    "Plan": {
        "Stages": [
            {
                "DurationInMinutes": 5,
                "Targets": [
                    {
                        "ChannelTargetInfo": {
                            "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/beb25840-5ac8-4644-95cc-7a8de390fa65",
                            "RetryIntervalInMinutes": 1
                        }
                    }
                ]
            },
            {
                "DurationInMinutes": 5,
                "Targets": [
                    {
                        "ChannelTargetInfo": {
                            "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad",
                            "RetryIntervalInMinutes": 1
                        }
                    }
                ]
            },
            {
                "DurationInMinutes": 5,
                "Targets": [
                    {
                        "ChannelTargetInfo": {
                            "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/77d4f447-f619-4954-afff-85551e369c2a",
                            "RetryIntervalInMinutes": 1
                        }
                    }
                ]
```

```
            }
        ]
    }
}
```

示例 2：描述升级计划

以下 get-contact 示例描述一个升级计划。

```
aws ssm-contacts get-contact \
--contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation"
```

输出：

```
{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
    "Alias": "example_escalation",
    "DisplayName": "Example Escalation",
    "Type": "ESCALATION",
    "Plan": {
        "Stages": [
            {
                "DurationInMinutes": 5,
                "Targets": [
                    {
                        "ContactTargetInfo": {
                            "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/akuam",
                            "IsEssential": true
                        }
                    }
                ]
            },
            {
                "DurationInMinutes": 5,
                "Targets": [
                    {
                        "ContactTargetInfo": {
                            "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/alejr",
                            "IsEssential": false
```

```
                        }
                    }
                ]
            },
            {
                "DurationInMinutes": 0,
                "Targets": [
                    {
                        "ContactTargetInfo": {
                            "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/anasi",
                            "IsEssential": false
                        }
                    }
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContact。

`list-contact-channels`

以下代码示例演示了如何使用 `list-contact-channels`。

Amazon CLI

列出联系人的联系频道

以下 `list-contact-channels` 示例列出指定联系人的可用联系频道。

```
aws ssm-contacts list-contact-channels \
    --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

输出：

```
{
    [
        {
```

```
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Name": "akuas email",
            "Type": "EMAIL",
            "DeliveryAddress": {
                "SimpleAddress": "akuam@example.com"
            },
            "ActivationStatus": "NOT_ACTIVATED"
        },
        {
            "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Name": "akuas sms",
            "Type": "SMS",
            "DeliveryAddress": {
                "SimpleAddress": "+15005550100"
            },
            "ActivationStatus": "ACTIVATED"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContactChannels。

`list-contacts`

以下代码示例演示了如何使用 `list-contacts`。

Amazon CLI

列出所有升级计划和联系人

以下 `list-contacts` 示例列出您账户中的联系人和升级计划。

```
aws ssm-contacts list-contacts
```

输出：

```
{
```

```
    "Contacts": [
        {
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Alias": "akuam",
            "DisplayName": "Akua Mansa",
            "Type": "PERSONAL"
        },
        {
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
alejr",
            "Alias": "alejr",
            "DisplayName": "Alejandro Rosalez",
            "Type": "PERSONAL"
        },
        {
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
anasi",
            "Alias": "anasi",
            "DisplayName": "Ana Carolina Silva",
            "Type": "PERSONAL"
        },
        {
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
            "Alias": "example_escalation",
            "DisplayName": "Example Escalation",
            "Type": "ESCALATION"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListContacts。

`list-engagements`

以下代码示例演示了如何使用 `list-engagements`。

Amazon CLI

列出所有参与情况

以下 list-engagements 示例列出升级计划和联系人的参与情况。您还可以列出单个事件的参与情况。

```
aws ssm-contacts list-engagements
```

输出：

```
{
    "Engagements": [
        {
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/91792571-0b53-4821-9f73-d25d13d9e529",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Sender": "cli",
            "StartTime": "2021-05-18T20:37:50.300000+00:00"
        },
        {
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Sender": "cli",
            "StartTime": "2021-05-18T18:40:26.666000+00:00"
        },
        {
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
            "Sender": "cli",
            "StartTime": "2021-05-18T18:25:41.151000+00:00"
        },
        {
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/607ced0e-e8fa-4ea7-8958-a237b8803f8f",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Sender": "cli",
            "StartTime": "2021-05-18T18:20:58.093000+00:00"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEngagements。

`list-page-receipts`

以下代码示例演示了如何使用 `list-page-receipts`。

Amazon CLI

列出页面回执

以下 command-name 示例列出联系人是否收到页面。

```
aws ssm-contacts list-page-receipts \
    --page-id "arn:aws:ssm-contacts:us-east-2:111122223333:page/
akuam/94ea0c7b-56d9-46c3-b84a-a37c8b067ad3"
```

输出：

```
{
    "Receipts": [
        {
            "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
            "ReceiptType": "DELIVERED",
            "ReceiptInfo": "425440",
            "ReceiptTime": "2021-05-18T20:42:57.485000+00:00"
        },
        {
            "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
            "ReceiptType": "READ",
            "ReceiptInfo": "425440",
            "ReceiptTime": "2021-05-18T20:42:57.907000+00:00"
        },
        {
            "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
            "ReceiptType": "SENT",
```

```
            "ReceiptInfo": "SM6656c19132f1465f9c9c1123a5dde7c9",
            "ReceiptTime": "2021-05-18T20:40:52.962000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPageReceipts。

list-pages-by-contact

以下代码示例演示了如何使用 list-pages-by-contact。

Amazon CLI

按联系人列出页面

以下 list-pages-by-contact 示例列出指定联系人的所有页面。

```
aws ssm-contacts list-pages-by-contact \
    --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

输出：

```
{
    "Pages": [
        {
            "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/
ad0052bd-e606-498a-861b-25726292eb93",
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Sender": "cli",
            "SentTime": "2021-05-18T18:43:29.301000+00:00",
            "DeliveryTime": "2021-05-18T18:43:55.265000+00:00",
            "ReadTime": "2021-05-18T18:43:55.708000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPagesByContact。

list-pages-by-engagement

以下代码示例演示了如何使用 list-pages-by-engagement。

Amazon CLI

列出从参与开始的联系频道页面

以下 list-pages-by-engagement 示例列出在执行已定义参与计划时出现的页面。

```
aws ssm-contacts list-pages-by-engagement \
    --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
akuam/78a29753-3674-4ac5-9f83-0468563567f0"
```

输出：

```
{
    "Pages": [
        {
            "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/
ad0052bd-e606-498a-861b-25726292eb93",
            "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
            "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
            "Sender": "cli",
            "SentTime": "2021-05-18T18:40:27.245000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPagesByEngagement。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出联系人的标签

以下 `list-tags-for-resource` 示例列出指定联系人的标签。

```
aws ssm-contacts list-tags-for-resource \
    --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam"
```

输出：

```
{
    "Tags": [
        {
            "Key": "group1",
            "Value": "1"
        }
    ]
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-contact-policy

以下代码示例演示了如何使用 `put-contact-policy`。

Amazon CLI

共享联系人和参与情况

以下 `put-contact-policy` 示例向联系人 Akua 添加资源策略，与主体共享联系人和相关参与情况。

```
aws ssm-contacts put-contact-policy \
    --contact-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \
    --policy "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":
\"ExampleResourcePolicy\",\"Action\":[\"ssm-contacts:GetContact\",\"ssm-
contacts:StartEngagement\",\"ssm-contacts:DescribeEngagement\",\"ssm-
contacts:ListPagesByEngagement\",\"ssm-contacts:StopEngagement\"],
```

```
\"Principal\":{\"AWS\":\"222233334444\"},\"Effect\":\"Allow\",\"Resource
\":[\"arn:aws:ssm-contacts:*:111122223333:contact\/akuam\",\"arn:aws:ssm-
contacts:*:111122223333:engagement\/akuam\/*\"]}]}"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的使用共享的联系人和响应计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutContactPolicy。

send-activation-code

以下代码示例演示了如何使用 send-activation-code。

Amazon CLI

发送激活码

以下 send-activation-code 示例向指定联系频道发送激活码和消息。

```
aws ssm-contacts send-activation-code \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact-
channel/akuam/8ddae2d1-12c8-4e45-b852-c8587266c400"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendActivationCode。

start-engagement

以下代码示例演示了如何使用 start-engagement。

Amazon CLI

示例 1：寻呼联系人的联系频道

以下 start-engagement 将寻呼联系人的联系频道。发件人、主题、公共主题和公共内容都没有字段。Incident Manager 会将主题和内容发送到提供的语音或电子邮件联系频道。Incident Manager 会将公共主题和公共内容发送到提供的短信联系频道。发件人用于跟踪发起参与的对象。

```
aws ssm-contacts start-engagement \
    --contact-id  "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam" \
    --sender "cli" \
    --subject "cli-test" \
    --content "Testing engagements via CLI" \
    --public-subject "cli-test" \
    --public-content "Testing engagements va CLI"
```

输出：

```
{
    "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
akuam/607ced0e-e8fa-4ea7-8958-a237b8803f8f"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

示例 2：在提供的升级计划中寻呼联系人

以下 `start-engagement` 将通过升级计划吸引联系人。将根据每位联系人的参与计划对其进行寻呼。

```
aws ssm-contacts start-engagement \
    --contact-id  "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation" \
    --sender "cli" \
    --subject "cli-test" \
    --content "Testing engagements via CLI" \
    --public-subject "cli-test" \
    --public-content "Testing engagements va CLI"
```

输出：

```
{
    "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"
}
```

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartEngagement。

stop-engagement

以下代码示例演示了如何使用 stop-engagement。

Amazon CLI

停止参与

以下 stop-engagement 示例阻止参与寻呼更多联系人和联系频道。

```
aws ssm-contacts stop-engagement \
    --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopEngagement。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记联系人

以下 tag-resource 示例使用提供的标签键值对标记指定的联系人。

```
aws ssm-contacts tag-resource \
    --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \
    --tags '[{"Key":"group1","Value":"1"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除联系人的标签

以下 untag-resource 示例移除指定联系人的“group1”标签。

```
aws ssm-contacts untag-resource \
    --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \
    --tag-keys "group1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-contact-channel

以下代码示例演示了如何使用 update-contact-channel。

Amazon CLI

更新联系频道

以下 update-contact-channel 示例更新联系频道的名称和收货地址。

```
aws ssm-contacts update-contact-channel \
    --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad" \
    --name "akuas voice channel" \
    --delivery-address '{"SimpleAddress": "+15005550198"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContactChannel。

update-contact

以下代码示例演示了如何使用 update-contact。

Amazon CLI

更新联系人参与计划

以下 update-contact 示例更新联系人 Akua 的参与计划，使其包括三种类型的联系频道。这将在为 Akua 创建联系频道之后完成。

```
aws ssm-contacts update-contact \
    --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam" \
    --plan '{"Stages": [{"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo":
 {"ContactChannelId": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-
channel/akuam/beb25840-5ac8-4644-95cc-7a8de390fa65","RetryIntervalInMinutes":
 1 }}]}, {"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo":
{"ContactChannelId": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/
akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad", "RetryIntervalInMinutes": 1}}]},
 {"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo": {"ContactChannelId":
 "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/77d4f447-
f619-4954-afff-85551e369c2a","RetryIntervalInMinutes": 1 }}]}]}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Incident Manager 用户指南》中的联系人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateContact。

使用 Amazon CLI 的 Amazon Inspector 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Inspector 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-attributes-to-findings

以下代码示例演示了如何使用 add-attributes-to-findings。

Amazon CLI

为调查发现添加属性

以下 add-attribute-to-finding 命令将键为 Example 且值为 example 的属性分配给 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-8l1VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU 的调查发现：

```
aws inspector add-attributes-to-findings --finding-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-8l1VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU --attributes key=Example,value=example
```

输出：

```
{
    "failedItems": {}
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 调查发现”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddAttributesToFindings。

associate-member

以下代码示例演示了如何使用 associate-member。

Amazon CLI

示例：将 Amazon 账户与 Amazon Inspector 委托管理员关联

以下 associate-member 示例将 Amazon 账户与 Amazon Inspector 委托管理员关联。

```
aws inspector2 associate-member \
    --account-id 123456789012
```

输出：

```
{
    "accountId": "123456789012"
}
```

有关更多信息，请参阅《Amazon Inspector User Guide》中的 Managing multiple accounts in Amazon Inspector with Amazon Organizations。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssociateMember。

create-assessment-target

以下代码示例演示了如何使用 create-assessment-target。

Amazon CLI

创建评估目标

以下 create-assessment-target 命令使用 ARN 为 arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-AB6DMKnv 的资源组创建名为 ExampleAssessmentTarget 的评估目标：

```
aws inspector create-assessment-target --assessment-target-
name ExampleAssessmentTarget --resource-group-arn arn:aws:inspector:us-
west-2:123456789012:resourcegroup/0-AB6DMKnv
```

输出：

```
{
    "assessmentTargetArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX"
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAssessmentTarget。

create-assessment-template

以下代码示例演示了如何使用 create-assessment-template。

Amazon CLI

创建评估模板

以下 create-assessment-template 命令为 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX 的评估目标创建名为 ExampleAssessmentTemplate 的评估模板：

```
aws inspector create-assessment-template --assessment-target-
arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX --assessment-
template-name ExampleAssessmentTemplate --duration-in-seconds 180 --rules-package-
arns arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p --user-
attributes-for-findings key=ExampleTag,value=examplevalue
```

输出：

```
{
    "assessmentTemplateArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX/template/0-it5r2S4T"
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAssessmentTemplate。

create-filter

以下代码示例演示了如何使用 create-filter。

Amazon CLI

创建筛选器

以下 create-filter 示例创建了一个忽略 ECR 实例类型调查发现的抑制规则。

```
aws inspector2 create-filter \
    --name "ExampleSuppressionRuleECR" \
    --description "This suppression rule omits ECR instance type findings" \
    --action SUPPRESS \
    --filter-criteria 'resourceType=[{comparison="EQUALS",
 value="AWS_ECR_INSTANCE"}]'
```

输出：

```
{
    "arn": "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/filter/
EXAMPLE444444444"
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的筛选 Amazon Inspector 调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFilter。

create-findings-report

以下代码示例演示了如何使用 create-findings-report。

Amazon CLI

创建调查发现报告

以下 create-findings-report 示例创建了一个调查发现报告。

```
aws inspector2 create-findings-report \
    --report-format CSV \
    --s3-destination bucketName=inspector-sbom-123456789012,keyPrefix=sbom-
key,kmsKeyArn=arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333 \
    --filter-criteria '{"ecrImageRepositoryName":
[{"comparison":"EQUALS","value":"debian"}]}'
```

输出：

```
{
    "reportId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的在 Amazon Inspector 中管理调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFindingsReport。

create-resource-group

以下代码示例演示了如何使用 create-resource-group。

Amazon CLI

创建资源组

以下 create-resource-group 命令使用标签键为 Name 以及值为 example 创建一个资源组：

```
aws inspector create-resource-group --resource-group-tags key=Name,value=example
```

输出：

```
{
   "resourceGroupArn": "arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-
AB6DMKnv"
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResourceGroup。

create-sbom-export

以下代码示例演示了如何使用 create-sbom-export。

Amazon CLI

创建软件物料清单（SBOM）报告

以下 create-sbom-export 示例创建了一个软件物料清单（SBOM）报告。

```
aws inspector2 create-sbom-export \
    --report-format SPDX_2_3 \
    --resource-filter-criteria
 'ecrRepositoryName=[{comparison="EQUALS",value="debian"}]' \
    --s3-destination bucketName=inspector-sbom-123456789012,keyPrefix=sbom-
key,kmsKeyArn=arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333
```

输出：

```
{
    "reportId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的使用 Amazon Inspector 导出 SBOM。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSbomExport。

delete-assessment-run

以下代码示例演示了如何使用 delete-assessment-run。

Amazon CLI

删除评估运行

以下 delete-assessment-run 命令删除 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-11LMTAVe 的评估运行：

```
aws inspector delete-assessment-run --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-11LMTAVe
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAssessmentRun。

delete-assessment-target

以下代码示例演示了如何使用 delete-assessment-target。

Amazon CLI

删除评估目标

以下 delete-assessment-target 命令删除 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq 的评估目标：

```
aws inspector delete-assessment-target --assessment-target-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAssessmentTarget。

delete-assessment-template

以下代码示例演示了如何使用 delete-assessment-template。

Amazon CLI

删除评估模板

以下 delete-assessment-template 命令删除 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T 的评估模板：

```
aws inspector delete-assessment-template --assessment-template-
arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAssessmentTemplate。

delete-filter

以下代码示例演示了如何使用 delete-filter。

Amazon CLI

删除筛选器

以下 delete-filter 示例删除一个筛选器。

```
aws inspector2 delete-filter \
    --arn "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/filter/
EXAMPLE444444444"
```

输出：

```
{
    "arn": "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/filter/
EXAMPLE444444444"
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的筛选 Amazon Inspector 调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFilter。

describe-assessment-runs

以下代码示例演示了如何使用 describe-assessment-runs。

Amazon CLI

描述评估运行

以下 describe-assessment-run 命令描述 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE 的评估运行：

```
aws inspector describe-assessment-runs --assessment-run-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE
```

输出：

```
{
        "assessmentRuns": [
          {
                "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE",
                "assessmentTemplateArn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw",
                "completedAt": 1458680301.4,
                "createdAt": 1458680170.035,
                "dataCollected": true,
                "durationInSeconds": 3600,
                "name": "Run 1 for ExampleAssessmentTemplate",
                "notifications": [],
                "rulesPackageArns": [
                  "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-X1KXtawP"
                ],
                "startedAt": 1458680170.161,
                "state": "COMPLETED",
                "stateChangedAt": 1458680301.4,
                "stateChanges": [
                  {
                        "state": "CREATED",
                        "stateChangedAt": 1458680170.035
```

```
                    },
                    {
                          "state": "START_DATA_COLLECTION_PENDING",
                          "stateChangedAt": 1458680170.065
                    },
                    {
                          "state": "START_DATA_COLLECTION_IN_PROGRESS",
                          "stateChangedAt": 1458680170.096
                    },
                    {
                          "state": "COLLECTING_DATA",
                          "stateChangedAt": 1458680170.161
                    },
                    {
                          "state": "STOP_DATA_COLLECTION_PENDING",
                          "stateChangedAt": 1458680239.883
                    },
                    {
                          "state": "DATA_COLLECTED",
                          "stateChangedAt": 1458680299.847
                    },
                    {
                          "state": "EVALUATING_RULES",
                          "stateChangedAt": 1458680300.099
                    },
                    {
                          "state": "COMPLETED",
                          "stateChangedAt": 1458680301.4
                    }
                  ],
                  "userAttributesForFindings": []
            }
          ],
          "failedItems": {}
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAssessmentRuns。

describe-assessment-targets

以下代码示例演示了如何使用 describe-assessment-targets。

Amazon CLI

描述评估目标

以下 describe-assessment-targets 命令描述 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq 的评估目标：

```
aws inspector describe-assessment-targets --assessment-target-
arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq
```

输出：

```
{
      "assessmentTargets": [
        {
              "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq",
              "createdAt": 1458074191.459,
              "name": "ExampleAssessmentTarget",
              "resourceGroupArn": "arn:aws:inspector:us-
west-2:123456789012:resourcegroup/0-PyGXopAI",
              "updatedAt": 1458074191.459
        }
      ],
      "failedItems": {}
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAssessmentTargets。

describe-assessment-templates

以下代码示例演示了如何使用 describe-assessment-templates。

Amazon CLI

描述评估模板

以下 describe-assessment-templates 命令描述 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw 的评估模板：

```
aws inspector describe-assessment-templates --assessment-template-
arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw
```

输出：

```
{
      "assessmentTemplates": [
        {
              "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw",
              "assessmentTargetArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq",
              "createdAt": 1458074191.844,
              "durationInSeconds": 3600,
              "name": "ExampleAssessmentTemplate",
              "rulesPackageArns": [
                "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-X1KXtawP"
              ],
              "userAttributesForFindings": []
        }
      ],
      "failedItems": {}
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAssessmentTemplates。

describe-cross-account-access-role

以下代码示例演示了如何使用 describe-cross-account-access-role。

Amazon CLI

描述跨账户访问角色

以下 describe-cross-account-access-role 命令描述使 Amazon Inspector 能够访问您的 Amazon 账户的 IAM 角色：

```
aws inspector describe-cross-account-access-role
```

输出：

```
{
        "registeredAt": 1458069182.826,
        "roleArn": "arn:aws:iam::123456789012:role/inspector",
        "valid": true
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“设置 Amazon Inspector”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCrossAccountAccessRole。

describe-findings

以下代码示例演示了如何使用 describe-findings。

Amazon CLI

描述调查发现

以下 describe-findings 命令描述 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4 的调查发现：

```
aws inspector describe-findings --finding-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4
```

输出：

```
{
      "failedItems": {},
      "findings": [
        {
              "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4",
              "assetAttributes": {
                "ipv4Addresses": [],
                "schemaVersion": 1
              },
              "assetType": "ec2-instance",
              "attributes": [],
```

```
            "confidence": 10,
            "createdAt": 1458680301.37,
            "description": "Amazon Inspector did not find any potential security
 issues during this assessment.",
            "indicatorOfCompromise": false,
            "numericSeverity": 0,
            "recommendation": "No remediation needed.",
            "schemaVersion": 1,
            "service": "Inspector",
            "serviceAttributes": {
              "assessmentRunArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE",
              "rulesPackageArn": "arn:aws:inspector:us-
west-2:758058086616:rulespackage/0-X1KXtawP",
              "schemaVersion": 1
            },
            "severity": "Informational",
            "title": "No potential security issues found",
            "updatedAt": 1458680301.37,
            "userAttributes": []
        }
      ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 调查发现”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFindings。

describe-resource-groups

以下代码示例演示了如何使用 describe-resource-groups。

Amazon CLI

描述资源组

以下 describe-resource-groups 命令描述 ARN 为 arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-PyGXopAI 的资源组：

```
aws inspector describe-resource-groups --resource-group-arns arn:aws:inspector:us-
west-2:123456789012:resourcegroup/0-PyGXopAI
```

输出：

```
{
      "failedItems": {},
      "resourceGroups": [
        {
              "arn": "arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-
PyGXopAI",
              "createdAt": 1458074191.098,
              "tags": [
                {
                      "key": "Name",
                      "value": "example"
                }
              ]
        }
      ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResourceGroups。

describe-rules-packages

以下代码示例演示了如何使用 describe-rules-packages。

Amazon CLI

描述规则包

以下 describe-rules-packages 命令描述 ARN 为 arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p 的规则包：

```
aws inspector describe-rules-packages --rules-package-arns arn:aws:inspector:us-
west-2:758058086616:rulespackage/0-9hgA516p
```

输出：

```
{
      "failedItems": {},
      "rulesPackages": [
        {
```

```
            "arn": "arn:aws:inspector:us-
west-2:758058086616:rulespackage/0-9hgA516p",
            "description": "The rules in this package help verify whether the EC2
 instances in your application are exposed to Common Vulnerabilities and
            Exposures (CVEs). Attacks can exploit unpatched vulnerabilities to
 compromise the confidentiality, integrity, or availability of your service
            or data. The CVE system provides a reference for publicly known
 information security vulnerabilities and exposures. For more information, see
            [https://cve.mitre.org/](https://cve.mitre.org/). If a particular CVE
 appears in one of the produced Findings at the end of a completed
            Inspector assessment, you can search [https://cve.mitre.org/](https://
cve.mitre.org/) using the CVE's ID (for example, \"CVE-2009-0021\") to
            find detailed information about this CVE, its severity, and how to
 mitigate it. ",
            "name": "Common Vulnerabilities and Exposures",
            "provider": "Amazon Web Services, Inc.",
            "version": "1.1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 规则包和规则”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRulesPackages。

disassociate-member

以下代码示例演示了如何使用 disassociate-member。

Amazon CLI

示例：取消成员账户与 Amazon Inspector 委托管理员的关联

以下 disassociate-member 示例取消 Amazon 账户与 Amazon Inspector 委托管理员的关联。

```
aws inspector2 disassociate-member \
    --account-id 123456789012
```

输出：

```
{
    "accountId": "123456789012"
```

```
}
```

有关更多信息，请参阅《Amazon Inspector User Guide》中的 Managing multiple accounts in Amazon Inspector with Amazon Organizations。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DisassociateMember。

`get-configuration`

以下代码示例演示了如何使用 `get-configuration`。

Amazon CLI

获取 Inspector 扫描的设置配置

以下 `get-configuration` 示例获取 Inspector 扫描的设置配置。

```
aws inspector2 get-configuration
```

输出：

```
{
    "ec2Configuration": {
        "scanModeState": {
            "scanMode": "EC2_HYBRID",
            "scanModeStatus": "SUCCESS"
        }
    },
    "ecrConfiguration": {
        "rescanDurationState": {
            "pullDateRescanDuration": "DAYS_90",
            "rescanDuration": "DAYS_30",
            "status": "SUCCESS",
            "updatedAt": "2024-05-14T21:16:20.237000+00:00"
        }
     }
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的使用 Amazon Inspector 自动扫描资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConfiguration。

get-member

以下代码示例演示了如何使用 get-member。

Amazon CLI

示例：获取组织的成员信息

aws inspector2 get-member --account-id 123456789012

输出：

```
{
        "member": {
        "accountId": "123456789012",
        "delegatedAdminAccountId": "123456789012",
        "relationshipStatus": "ENABLED",
        "updatedAt": "2023-09-11T09:57:20.520000-07:00"
    }
}
```

有关更多信息，请参阅《Amazon Inspector User Guide》中的 Managing multiple accounts in Amazon Inspector with Amazon Organizations。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMember。

get-telemetry-metadata

以下代码示例演示了如何使用 get-telemetry-metadata。

Amazon CLI

获取遥测元数据

以下 get-telemetry-metadata 命令生成有关为 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE 的评估运行收集的数据的信息：

```
aws inspector get-telemetry-metadata --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE
```

输出：

```
{
      "telemetryMetadata": [
        {
              "count": 2,
              "dataSize": 345,
              "messageType": "InspectorDuplicateProcess"
        },
        {
              "count": 3,
              "dataSize": 255,
              "messageType": "InspectorTimeEventMsg"
        },
        {
              "count": 4,
              "dataSize": 1082,
              "messageType": "InspectorNetworkInterface"
        },
        {
              "count": 2,
              "dataSize": 349,
              "messageType": "InspectorDnsEntry"
        },
        {
              "count": 11,
              "dataSize": 2514,
              "messageType": "InspectorDirectoryInfoMsg"
        },
        {
              "count": 1,
              "dataSize": 179,
              "messageType": "InspectorTcpV6ListeningPort"
        },
        {
              "count": 101,
              "dataSize": 10949,
              "messageType": "InspectorTerminal"
        },
        {
              "count": 26,
              "dataSize": 5916,
              "messageType": "InspectorUser"
        },
        {
```

```
            "count": 282,
            "dataSize": 32148,
            "messageType": "InspectorDynamicallyLoadedCodeModule"
        },
        {
            "count": 18,
            "dataSize": 10172,
            "messageType": "InspectorCreateProcess"
        },
        {
            "count": 3,
            "dataSize": 8001,
            "messageType": "InspectorProcessPerformance"
        },
        {
            "count": 1,
            "dataSize": 360,
            "messageType": "InspectorOperatingSystem"
        },
        {
            "count": 6,
            "dataSize": 546,
            "messageType": "InspectorStopProcess"
        },
        {
            "count": 1,
            "dataSize": 1553,
            "messageType": "InspectorInstanceMetaData"
        },
        {
            "count": 2,
            "dataSize": 434,
            "messageType": "InspectorTcpV4Connection"
        },
        {
            "count": 474,
            "dataSize": 2960322,
            "messageType": "InspectorPackageInfo"
        },
        {
            "count": 3,
            "dataSize": 2235,
            "messageType": "InspectorSystemPerformance"
        },
```

```
        {
              "count": 105,
              "dataSize": 46048,
              "messageType": "InspectorCodeModule"
        },
        {
              "count": 1,
              "dataSize": 182,
              "messageType": "InspectorUdpV6ListeningPort"
        },
        {
              "count": 2,
              "dataSize": 371,
              "messageType": "InspectorUdpV4ListeningPort"
        },
        {
              "count": 18,
              "dataSize": 8362,
              "messageType": "InspectorKernelModule"
        },
        {
              "count": 29,
              "dataSize": 48788,
              "messageType": "InspectorConfigurationInfo"
        },
        {
              "count": 1,
              "dataSize": 79,
              "messageType": "InspectorMonitoringStart"
        },
        {
              "count": 5,
              "dataSize": 0,
              "messageType": "InspectorSplitMsgBegin"
        },
        {
              "count": 51,
              "dataSize": 4593,
              "messageType": "InspectorGroup"
        },
        {
              "count": 1,
              "dataSize": 184,
              "messageType": "InspectorTcpV4ListeningPort"
```

```
        },
        {
              "count": 1159,
              "dataSize": 3146579,
              "messageType": "Total"
        },
        {
              "count": 5,
              "dataSize": 0,
              "messageType": "InspectorSplitMsgEnd"
        },
        {
              "count": 1,
              "dataSize": 612,
              "messageType": "InspectorLoadImageInProcess"
        }
      ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTelemetryMetadata。

list-account-permissions

以下代码示例演示了如何使用 list-account-permissions。

Amazon CLI

列出账户权限

以下 list-account-permissions 示例列出您的账户权限。

```
aws inspector2 list-account-permissions
```

输出：

```
{
    "permissions": [
        {
            "operation": "ENABLE_SCANNING",
            "service": "ECR"
        },
        {
            "operation": "DISABLE_SCANNING",
```

```
            "service": "ECR"
        },
        {
            "operation": "ENABLE_REPOSITORY",
            "service": "ECR"
        },
        {
            "operation": "DISABLE_REPOSITORY",
            "service": "ECR"
        },
        {
            "operation": "ENABLE_SCANNING",
            "service": "EC2"
        },
        {
            "operation": "DISABLE_SCANNING",
            "service": "EC2"
        },
        {
            "operation": "ENABLE_SCANNING",
            "service": "LAMBDA"
        },
        {
            "operation": "DISABLE_SCANNING",
            "service": "LAMBDA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的 Amazon Inspector 的身份和访问管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccountPermissions。

`list-assessment-run-agents`

以下代码示例演示了如何使用 `list-assessment-run-agents`。

Amazon CLI

列出评估运行代理

以下 `list-assessment-run-agents` 命令列出具有指定 ARN 的评估运行的代理。

```
aws inspector list-assessment-run-agents \
    --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE
```

输出：

```
{
    "assessmentRunAgents": [
        {
            "agentHealth": "HEALTHY",
            "agentHealthCode": "HEALTHY",
            "agentId": "i-49113b93",
            "assessmentRunArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE",
            "telemetryMetadata": [
                {
                    "count": 2,
                    "dataSize": 345,
                    "messageType": "InspectorDuplicateProcess"
                },
                {
                    "count": 3,
                    "dataSize": 255,
                    "messageType": "InspectorTimeEventMsg"
                },
                {
                    "count": 4,
                    "dataSize": 1082,
                    "messageType": "InspectorNetworkInterface"
                },
                {
                    "count": 2,
                    "dataSize": 349,
                    "messageType": "InspectorDnsEntry"
                },
                {
                    "count": 11,
                    "dataSize": 2514,
                    "messageType": "InspectorDirectoryInfoMsg"
                },
                {
                    "count": 1,
                    "dataSize": 179,
```

```
            "messageType": "InspectorTcpV6ListeningPort"
        },
        {
            "count": 101,
            "dataSize": 10949,
            "messageType": "InspectorTerminal"
        },
        {
            "count": 26,
            "dataSize": 5916,
            "messageType": "InspectorUser"
        },
        {
            "count": 282,
            "dataSize": 32148,
            "messageType": "InspectorDynamicallyLoadedCodeModule"
        },
        {
            "count": 18,
            "dataSize": 10172,
            "messageType": "InspectorCreateProcess"
        },
        {
            "count": 3,
            "dataSize": 8001,
            "messageType": "InspectorProcessPerformance"
        },
        {
            "count": 1,
            "dataSize": 360,
            "messageType": "InspectorOperatingSystem"
        },
        {
            "count": 6,
            "dataSize": 546,
            "messageType": "InspectorStopProcess"
        },
        {
            "count": 1,
            "dataSize": 1553,
            "messageType": "InspectorInstanceMetaData"
        },
        {
            "count": 2,
```

```
                    "dataSize": 434,
                    "messageType": "InspectorTcpV4Connection"
                },
                {
                    "count": 474,
                    "dataSize": 2960322,
                    "messageType": "InspectorPackageInfo"
                },
                {
                    "count": 3,
                    "dataSize": 2235,
                    "messageType": "InspectorSystemPerformance"
                },
                {
                    "count": 105,
                    "dataSize": 46048,
                    "messageType": "InspectorCodeModule"
                },
                {
                    "count": 1,
                    "dataSize": 182,
                    "messageType": "InspectorUdpV6ListeningPort"
                },
                {
                    "count": 2,
                    "dataSize": 371,
                    "messageType": "InspectorUdpV4ListeningPort"
                },
                {
                    "count": 18,
                    "dataSize": 8362,
                    "messageType": "InspectorKernelModule"
                },
                {
                    "count": 29,
                    "dataSize": 48788,
                    "messageType": "InspectorConfigurationInfo"
                },
                {
                    "count": 1,
                    "dataSize": 79,
                    "messageType": "InspectorMonitoringStart"
                },
                {
```

```
                    "count": 5,
                    "dataSize": 0,
                    "messageType": "InspectorSplitMsgBegin"
                },
                {
                    "count": 51,
                    "dataSize": 4593,
                    "messageType": "InspectorGroup"
                },
                {
                    "count": 1,
                    "dataSize": 184,
                    "messageType": "InspectorTcpV4ListeningPort"
                },
                {
                    "count": 1159,
                    "dataSize": 3146579,
                    "messageType": "Total"
                },
                {
                    "count": 5,
                    "dataSize": 0,
                    "messageType": "InspectorSplitMsgEnd"
                },
                {
                    "count": 1,
                    "dataSize": 612,
                    "messageType": "InspectorLoadImageInProcess"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的 Amazon 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssessmentRunAgents。

`list-assessment-runs`

以下代码示例演示了如何使用 `list-assessment-runs`。

Amazon CLI

列出评估运行

以下 `list-assessment-runs` 命令列出所有现有的评估运行。

```
aws inspector list-assessment-runs
```

输出：

```
{
    "assessmentRunArns": [
        "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE",
        "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-v5D6fI3v"
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的 Amazon Inspector 评估模板和评估运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssessmentRuns。

list-assessment-targets

以下代码示例演示了如何使用 `list-assessment-targets`。

Amazon CLI

列出评估目标

以下 `list-assessment-targets` 命令列出所有现有的评估目标：

```
aws inspector list-assessment-targets
```

输出：

```
{
       "assessmentTargetArns": [
       "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq"
       ]
```

```
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssessmentTargets。

`list-assessment-templates`

以下代码示例演示了如何使用 `list-assessment-templates`。

Amazon CLI

列出评估模板

以下 `list-assessment-templates` 命令列出所有现有的评估模板：

```
aws inspector list-assessment-templates
```

输出：

```
{
       "assessmentTemplateArns": [
       "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw",
       "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-
Uza6ihLh"
       ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssessmentTemplates。

`list-coverage-statistics`

以下代码示例演示了如何使用 `list-coverage-statistics`。

Amazon CLI

示例 1：按组列出覆盖情况统计信息

以下 `list-coverage-statistics` 示例按组列出您的 Amazon 环境的覆盖情况统计信息。

```
aws inspector2 list-coverage-statistics \
   --group-by RESOURCE_TYPE
```

输出：

```
{
    "countsByGroup": [
        {
            "count": 56,
            "groupKey": "AWS_LAMBDA_FUNCTION"
        },
        {
            "count": 27,
            "groupKey": "AWS_ECR_REPOSITORY"
        },
        {
            "count": 18,
            "groupKey": "AWS_EC2_INSTANCE"
        },
        {
            "count": 3,
            "groupKey": "AWS_ECR_CONTAINER_IMAGE"
        },
        {
            "count": 1,
            "groupKey": "AWS_ACCOUNT"
        }
    ],
    "totalCounts": 105
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的评估您的 Amazon 环境的 Amazon Inspector 覆盖情况。

示例 2：按资源类型列出覆盖情况统计信息

以下 `list-coverage-statistics` 示例按资源类型列出您的 Amazon 环境的覆盖情况统计信息。

```
aws inspector2 list-coverage-statistics
    --filter-criteria '{"resourceType":
[{"comparison":"EQUALS","value":"AWS_ECR_REPOSITORY"}]}'
```

```
    --group-by SCAN_STATUS_REASON
```

输出：

```
{
    "countsByGroup": [
        {
            "count": 27,
            "groupKey": "SUCCESSFUL"
        }
    ],
    "totalCounts": 27
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的评估您的 Amazon 环境的 Amazon Inspector 覆盖情况。

示例 3：按 ECR 存储库名称列出覆盖情况统计信息

以下 `list-coverage-statistics` 示例按 ECR 存储库名称列出您的 Amazon 环境的覆盖情况统计信息。

```
aws inspector2 list-coverage-statistics
   --filter-criteria '{"ecrRepositoryName":
[{"comparison":"EQUALS","value":"debian"}]}'
   --group-by SCAN_STATUS_REASON
```

输出：

```
{
    "countsByGroup": [
        {
            "count": 3,
            "groupKey": "SUCCESSFUL"
        }
    ],
    "totalCounts": 3
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的评估您的 Amazon 环境的 Amazon Inspector 覆盖情况。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCoverageStatistics。

`list-coverage`

以下代码示例演示了如何使用 `list-coverage`。

Amazon CLI

示例 1：列出有关您的环境的覆盖情况详细信息

以下 `list-coverage` 示例列出您的环境的覆盖情况详细信息。

```
aws inspector2 list-coverage
```

输出：

```
{
    "coveredResources": [
        {
            "accountId": "123456789012",
            "lastScannedAt": "2024-05-20T16:23:20-07:00",
            "resourceId": "i-EXAMPLE55555555555",
            "resourceMetadata": {
                "ec2": {
                    "amiId": "ami-EXAMPLE6666666666",
                    "platform": "LINUX"
                }
            },
            "resourceType": "AWS_EC2_INSTANCE",
            "scanStatus": {
                "reason": "SUCCESSFUL",
                "statusCode": "ACTIVE"
            },
            "scanType": "PACKAGE"
        }
    ]
}
```

示例 2：列出有关 Lambda 函数资源类型的覆盖情况详细信息

以下 `list-coverage` 示例列出您的 Lamda 函数资源类型详细信息。

```
aws inspector2 list-coverage
    --filter-criteria '{"resourceType":
[{"comparison":"EQUALS","value":"AWS_LAMBDA_FUNCTION"}]}'
```

输出：

```
{
    "coveredResources": [
        {
            "accountId": "123456789012",
            "resourceId": "arn:aws:lambda:us-west-2:123456789012:function:Eval-
container-scan-results:$LATEST",
            "resourceMetadata": {
                "lambdaFunction": {
                    "functionName": "Eval-container-scan-results",
                    "functionTags": {},
                    "layers": [],
                    "runtime": "PYTHON_3_7"
                }
            },
            "resourceType": "AWS_LAMBDA_FUNCTION",
            "scanStatus": {
                "reason": "SUCCESSFUL",
                "statusCode": "ACTIVE"
            },
            "scanType": "CODE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCoverage。

list-delegated-admin-accounts

以下代码示例演示了如何使用 list-delegated-admin-accounts。

Amazon CLI

列出有关您的组织的委派管理员账户的信息

以下 list-delegated-admin-accounts 示例列出有关您的组织的委派管理员账户的信息。

```
aws inspector2 list-delegated-admin-accounts
```

输出：

```
{
    "delegatedAdminAccounts": [
        {
            "accountId": "123456789012",
            "status": "ENABLED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的为 Amazon Inspector 指定委派管理员。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDelegatedAdminAccounts。

list-event-subscriptions

以下代码示例演示了如何使用 list-event-subscriptions。

Amazon CLI

列出事件订阅

以下 list-event-subscriptions 命令列出 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0 的评估模板的所有事件订阅：

```
aws inspector list-event-subscriptions --resource-arn arn:aws:inspector:us-
west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0
```

输出：

```
{
      "subscriptions": [
        {
              "eventSubscriptions": [
                {
```

```
                    "event": "ASSESSMENT_RUN_COMPLETED",
                    "subscribedAt": 1459455440.867
                }
            ],
            "resourceArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX/template/0-7sbz2Kz0",
            "topicArn": "arn:aws:sns:us-west-2:123456789012:exampletopic"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEventSubscriptions。

list-filters

以下代码示例演示了如何使用 `list-filters`。

Amazon CLI

列出与您用于激活 Amazon Inspector 的账户相关的筛选器

以下 `list-filters` 示例列出与您用于激活 Amazon Inspector 的账户相关的筛选器。

```
aws inspector2 list-filters
```

输出：

```
{
    "filters": [
        {
            "action": "SUPPRESS",
            "arn": "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/
filter/EXAMPLE444444444",
            "createdAt": "2024-05-15T21:11:08.602000+00:00",
            "criteria": {
                "resourceType": [
                    {
                        "comparison": "EQUALS",
                        "value": "AWS_EC2_INSTANCE"
                    },
```

```
                ]
            },
            "description": "This suppression rule omits EC2 instance type findings",
            "name": "ExampleSuppressionRuleEC2",
            "ownerId": "o-EXAMPLE222",
            "tags": {},
            "updatedAt": "2024-05-15T21:11:08.602000+00:00"
        },
        {
            "action": "SUPPRESS",
            "arn": "arn:aws:inspector2:us-east-1:813737243517:owner/o-EXAMPLE222/
filter/EXAMPLE444444444",
            "createdAt": "2024-05-15T21:28:27.054000+00:00",
            "criteria": {
                "resourceType": [
                    {
                        "comparison": "EQUALS",
                        "value": "AWS_ECR_INSTANCE"
                    }
                ]
            },
            "description": "This suppression rule omits ECR instance type findings",
            "name": "ExampleSuppressionRuleECR",
            "ownerId": "o-EXAMPLE222",
            "tags": {},
            "updatedAt": "2024-05-15T21:28:27.054000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的筛选 Amazon Inspector 调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFilters。

`list-findings`

以下代码示例演示了如何使用 `list-findings`。

Amazon CLI

列出调查发现

以下 `list-findings` 命令列出所有生成的调查发现：

```
aws inspector list-findings
```

输出：

```
{
        "findingArns": [
        "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4",
        "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-v5D6fI3v/finding/0-tyvmqBLy"
      ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 调查发现”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFindings。

`list-members`

以下代码示例演示了如何使用 `list-members`。

Amazon CLI

示例 1：列出与组织的 Amazon Inspector 委托管理员关联的所有成员账户

aws inspector2 list-members --only-associated

输出：

```
{
        {
             "members": [
        {
             "accountId": "123456789012",
             "delegatedAdminAccountId": "123456789012",
             "relationshipStatus": "ENABLED",
             "updatedAt": "2023-09-11T09:57:20.520000-07:00"
        },
        {
             "accountId": "123456789012",
             "delegatedAdminAccountId": "123456789012",
             "relationshipStatus": "ENABLED",
```

```
            "updatedAt": "2024-08-12T10:13:01.472000-07:00"
        },
        {
            "accountId": "625032911453",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2023-09-11T09:57:20.438000-07:00"
        },
        {
            "accountId": "715411239211",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2024-04-24T09:14:57.471000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector User Guide》中的 Managing multiple accounts in Amazon Inspector with Amazon Organizations。

示例 2：列出与组织的 Amazon Inspector 委托管理员关联和取消关联的所有成员账户

aws inspector2 list-members --no-only-associated

输出：

```
{
        {
            "members": [
        {
            "accountId": "123456789012",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "REMOVED",
            "updatedAt": "2024-05-15T11:34:53.326000-07:00"
        },
        {
            "accountId": "123456789012",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2023-09-11T09:57:20.520000-07:00"
        },
        {
            "accountId": "123456789012",
```

```
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2024-08-12T10:13:01.472000-07:00"
        },
        {
            "accountId": "123456789012",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2023-09-11T09:57:20.438000-07:00"
        },
        {
            "accountId": "123456789012",
            "delegatedAdminAccountId": "123456789012",
            "relationshipStatus": "ENABLED",
            "updatedAt": "2024-04-24T09:14:57.471000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector User Guide》中的 Managing multiple accounts in Amazon Inspector with Amazon Organizations。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMembers。

`list-rules-packages`

以下代码示例演示了如何使用 `list-rules-packages`。

Amazon CLI

列出规则包

以下 `list-rules-packages` 命令列出所有可用的 Inspector 规则包：

```
aws inspector list-rules-packages
```

输出：

```
{
       "rulesPackageArns": [
         "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p",
         "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-H5hpSawc",
```

```
        "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-JJOtZiqQ",
        "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-vg5GGHSD"
      ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 规则包和规则”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRulesPackages。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 命令列出与 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-gcwFliYu 的评估模板关联的所有标签：

```
aws inspector list-tags-for-resource --resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-gcwFliYu
```

输出：

```
{
      "tags": [
        {
              "key": "Name",
              "value": "Example"
        }
      ]
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-usage-totals

以下代码示例演示了如何使用 list-usage-totals。

Amazon CLI

列出过去 30 天的总使用量

以下 `list-usage-totals` 示例列出了过去 30 天的总使用量。

```
aws inspector2 list-usage-totals
```

输出：

```
{
    "totals": [
        {
            "accountId": "123456789012",
            "usage": [
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 4.6022044647,
                    "total": 1893.4784083333334,
                    "type": "EC2_AGENTLESS_INSTANCE_HOURS"
                },
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 18.892449279,
                    "total": 10882.050784722222,
                    "type": "EC2_INSTANCE_HOURS"
                },
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 5.4525363736,
                    "total": 6543.043648333333,
                    "type": "LAMBDA_FUNCTION_CODE_HOURS"
                },
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 3.9064080309,
                    "total": 9375.379274166668,
                    "type": "LAMBDA_FUNCTION_HOURS"
                },
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 0.06,
                    "total": 6.0,
```

```
                    "type": "ECR_RESCAN"
                },
                {
                    "currency": "USD",
                    "estimatedMonthlyCost": 0.09,
                    "total": 1.0,
                    "type": "ECR_INITIAL_SCAN"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的在 Amazon Inspector 中监控使用量和成本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsageTotals。

preview-agents

以下代码示例演示了如何使用 preview-agents。

Amazon CLI

预览代理

以下 preview-agents 命令预览安装在 EC2 实例上的代理，这些代理属于 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq 的评估目标的一部分：

```
aws inspector preview-agents --preview-agents-arn arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq
```

输出：

```
{
      "agentPreviews": [
        {
              "agentId": "i-49113b93"
        }
      ]
```

```
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PreviewAgents。

register-cross-account-access-role

以下代码示例演示了如何使用 register-cross-account-access-role。

Amazon CLI

注册跨账户访问角色

以下 register-cross-account-access-role 命令注册 ARN 为 arn:aws:iam::123456789012:role/inspector 的 IAM 角色，当您调用 preview-agents 命令时，Amazon Inspector 使用该角色在评估运行开始时列出您的 EC2 实例：

```
aws inspector register-cross-account-access-role --role-
arn arn:aws:iam::123456789012:role/inspector
```

有关更多信息，请参阅《Amazon Inspector》指南中的“设置 Amazon Inspector”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterCrossAccountAccessRole。

remove-attributes-from-findings

以下代码示例演示了如何使用 remove-attributes-from-findings。

Amazon CLI

从调查发现中移除属性

以下 remove-attributes-from-finding 命令从 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-8l1VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU 的调查发现中移除键为 Example 且值为 example 的属性：

```
aws inspector remove-attributes-from-findings --finding-arns arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-8l1VIE0D/run/0-Z02cjjug/finding/0-
T8yM9mEU --attribute-keys key=Example,value=example
```

输出：

```
{
      "failedItems": {}
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 调查发现”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveAttributesFromFindings。

set-tags-for-resource

以下代码示例演示了如何使用 set-tags-for-resource。

Amazon CLI

为资源设置标签

以下 set-tags-for-resource 命令为 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0 的评估模板设置键为 Example 且值为 example 的标签：

```
aws inspector set-tags-for-resource --resource-arn arn:aws:inspector:us-
west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0 --
tags key=Example,value=example
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetTagsForResource。

start-assessment-run

以下代码示例演示了如何使用 start-assessment-run。

Amazon CLI

启动评估运行

以下 start-assessment-run 命令使用 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T 的评估模板启动名为 examplerun 的评估运行：

```
aws inspector start-assessment-run --assessment-run-name examplerun --assessment-
template-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-
it5r2S4T
```

输出：

```
{
      "assessmentRunArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX/template/0-it5r2S4T/run/0-jOoroxyY"
}
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartAssessmentRun。

stop-assessment-run

以下代码示例演示了如何使用 stop-assessment-run。

Amazon CLI

停止评估运行

以下 stop-assessment-run 命令停止 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-jOoroxyY 的评估运行：

```
aws inspector stop-assessment-run --assessment-run-arn arn:aws:inspector:us-
west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-jOoroxyY
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopAssessmentRun。

subscribe-to-event

以下代码示例演示了如何使用 subscribe-to-event。

Amazon CLI

订阅事件

以下示例启用向 ARN 为 arn:aws:sns:us-west-2:123456789012:exampletopic 的主题发送有关 ASSESSMENT_RUN_COMPLETED 事件的 Amazon SNS 通知的流程，

```
aws inspector subscribe-to-event \
    --event ASSESSMENT_RUN_COMPLETED \
    --resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/
template/0-7sbz2Kz0 \
    --topic-arn arn:aws:sns:us-west-2:123456789012:exampletopic
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Inspector》指南中的 Amazon Inspector 评估模板和评估运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SubscribeToEvent。

unsubscribe-from-event

以下代码示例演示了如何使用 unsubscribe-from-event。

Amazon CLI

取消订阅事件

以下 unsubscribe-from-event 命令禁用向 ARN 为 arn:aws:sns:us-west-2:123456789012:exampletopic 的主题发送有关 ASSESSMENT_RUN_COMPLETED 事件的 Amazon SNS 通知的流程：

```
aws inspector unsubscribe-from-event --event ASSESSMENT_RUN_COMPLETED --resource-
arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0
 --topic arn:aws:sns:us-west-2:123456789012:exampletopic
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估模板和评估运行”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnsubscribeFromEvent。

update-assessment-target

以下代码示例演示了如何使用 update-assessment-target。

Amazon CLI

更新评估目标

以下 update-assessment-target 命令将评估目标更新为 ARN 为 arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX，名称为 Example，资源组的 ARN 为 arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-yNbgL5Pt：

```
aws inspector update-assessment-target --assessment-target-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX --assessment-target-name Example --resource-group-arn arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-yNbgL5Pt
```

有关更多信息，请参阅《Amazon Inspector》指南中的“Amazon Inspector 评估目标”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAssessmentTarget。

update-filter

以下代码示例演示了如何使用 update-filter。

Amazon CLI

更新筛选器

以下 update-filter 示例更新筛选器，以忽略 Lambda 调查发现而不是 ECR 实例调查发现。

```
aws inspector2 update-filter \
   --filter-arn "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/filter/EXAMPLE444444444" \
   --name "ExampleSuppressionRuleLambda" \
   --description "This suppression rule omits Lambda instance findings" \
   --reason "Updating filter to omit Lambda instance findings instead of ECR instance findings"
```

输出：

```
{
    "filters": [
        {
            "action": "SUPPRESS",
            "arn": "arn:aws:inspector2:us-west-2:123456789012:owner/o-EXAMPLE222/filter/EXAMPLE444444444",
            "createdAt": "2024-05-15T21:28:27.054000+00:00",
            "criteria": {
                "resourceType": [
```

```
                    {
                        "comparison": "EQUALS",
                        "value": "AWS_ECR_INSTANCE"
                    }
                ]
            },
            "description": "This suppression rule omits Lambda instance findings",
            "name": "ExampleSuppressionRuleLambda",
            "ownerId": "o-EXAMPLE222",
            "reason": "Updating filter to omit Lambda instance findings instead of
 ECR instance findings",
            "tags": {},
            "updatedAt": "2024-05-15T22:23:13.665000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Inspector 用户指南》中的在 Amazon Inspector 中管理调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFilter。

Amazon IoT使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-certificate-transfer`

以下代码示例演示了如何使用 `accept-certificate-transfer`。

Amazon CLI

接受从其他 Amazon 账户转移的设备证书

以下 accept-certificate-transfer 示例接受从其他 Amazon 账户转移的设备证书。证书通过其 ID 来标识。

```
aws iot accept-certificate-transfer \
    --certificate-
id 488b6a7f2acdeb00a77384e63c4e40b18bEXAMPLEe57b7272ba44c45e3448142
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的将证书转移到另一个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptCertificateTransfer。

add-thing-to-billing-group

以下代码示例演示了如何使用 add-thing-to-billing-group。

Amazon CLI

示例 1：按名称向账单组添加事物

以下 add-thing-to-billing-group 示例将名为 MyLightBulb 的事物添加到名为 GroupOne 的账单组。

```
aws iot add-thing-to-billing-group \
    --billing-group-name GroupOne \
    --thing-name MyLightBulb
```

此命令不生成任何输出。

示例 2：按 ARN 向账单组添加事物

以下 add-thing-to-billing-group 示例将具有指定 ARN 的事物添加到具有指定 ARN 的账单组。如果您使用多个 Amazon 区域或账户，则指定 ARN 会很有用。它有助于确保您添加到正确的区域和账户。

```
aws iot add-thing-to-thing-group \
```

```
    --billing-group-arn "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne" \
    --thing-arn "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddThingToBillingGroup。

add-thing-to-thing-group

以下代码示例演示了如何使用 add-thing-to-thing-group。

Amazon CLI

向组中添加事物

以下 add-thing-to-thing-group 示例将指定事物添加到指定事物组。

```
aws iot add-thing-to-thing-group \
    --thing-name MyLightBulb \
    --thing-group-name LightBulbs
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddThingToThingGroup。

associate-targets-with-job

以下代码示例演示了如何使用 associate-targets-with-job。

Amazon CLI

将事物组与连续作业关联

以下 associate-targets-with-job 示例将指定事物组与指定连续作业相关联。

```
aws iot associate-targets-with-job \
    --targets "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \
    --job-id "example-job-04"
```

输出：

```
{
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-04",
    "jobId": "example-job-04",
    "description": "example continuous job"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateTargetsWithJob。

attach-policy

以下代码示例演示了如何使用 attach-policy。

Amazon CLI

示例 1：将策略附加到事物组

以下 attach-policy 示例将指定策略附加到由其 ARN 标识的事物组。

```
aws iot attach-policy \
    --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \
    --policy-name "UpdateDeviceCertPolicy"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

示例 2：将策略附加到证书

以下 attach-policy 示例将策略 UpdateDeviceCertPolicy 附加到通过证书指定的主体。

```
aws iot attach-policy \
    --policy-name UpdateDeviceCertPolicy \
    --target "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将 Amazon IoT 策略附加到设备证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachPolicy。

attach-security-profile

以下代码示例演示了如何使用 attach-security-profile。

Amazon CLI

将安全配置文件与所有未注册的设备关联

以下 attach-security-profile 示例将名为 Testprofile 的 Amazon IoT Device Defender 安全配置文件与 us-west-2 区域中该 Amazon 账户的所有未注册设备相关联。

```
aws iot attach-security-profile \
    --security-profile-name Testprofile \
    --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/
unregistered-things"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachSecurityProfile。

attach-thing-principal

以下代码示例演示了如何使用 attach-thing-principal。

Amazon CLI

将证书附加到事物

以下 attach-thing-principal 示例将证书附加到 MyTemperatureSensor 事物。证书通过其 ARN 来标识。您可以在 Amazon物联网控制台中找到证书的 ARN。

```
aws iot attach-thing-principal \
    --thing-name MyTemperatureSensor \
    --principal arn:aws:iot:us-
west-2:123456789012:cert/2e1eb273792174ec2b9bf4e9b37e6c6c692345499506002a35159767055278e8
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的如何使用注册表管理事物。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachThingPrincipal。

`cancel-audit-mitigation-actions-task`

以下代码示例演示了如何使用 `cancel-audit-mitigation-actions-task`。

Amazon CLI

取消审计缓解操作任务

以下 `cancel-audit-mitigations-action-task` 示例取消对指定任务应用缓解操作。您无法取消已完成的任务。

```
aws iot cancel-audit-mitigation-actions-task
    --task-id "myActionsTaskId"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 CancelAuditMitigationActionsTask（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelAuditMitigationActionsTask。

`cancel-audit-task`

以下代码示例演示了如何使用 `cancel-audit-task`。

Amazon CLI

取消审计任务

以下 `cancel-audit-task` 示例取消了具有指定作业 ID 的审计任务。您无法取消已完成的任务。

```
aws iot cancel-audit-task \
    --task-id a3aea009955e501a31b764abe1bebd3d
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelAuditTask。

cancel-certificate-transfer

以下代码示例演示了如何使用 cancel-certificate-transfer。

Amazon CLI

取消将证书转移到其他 Amazon 账户

以下 cancel-certificate-transfer 示例取消指定证书的转移。证书通过其证书 ID 来标识。您可以在 Amazon IoT 控制台中找到证书的 ID。

```
aws iot cancel-certificate-transfer \
    --certificate-
id f0f33678c7c9a046e5cc87b2b1a58dfa0beec26db78addd5e605d630e05c7fc8
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的将证书转移到另一个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelCertificateTransfer。

cancel-job-execution

以下代码示例演示了如何使用 cancel-job-execution。

Amazon CLI

取消设备上的作业执行

以下 cancel-job-execution 示例取消了在设备上执行指定作业。如果作业不是处于 QUEUED 状态，则必须添加 --force 参数。

```
aws iot cancel-job-execution \
    --job-id "example-job-03" \
    --thing-name "MyRPi"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelJobExecution。

cancel-job

以下代码示例演示了如何使用 cancel-job。

Amazon CLI

取消作业

以下 cancel-job 示例取消指定作业。

```
aws iot cancel-job \
    --job-job "example-job-03"
```

输出：

```
{
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-03",
    "jobId": "example-job-03",
    "description": "example job test"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelJob。

clear-default-authorizer

以下代码示例演示了如何使用 clear-default-authorizer。

Amazon CLI

清除默认授权方

以下 clear-default-authorizer 示例清除当前配置的默认自定义授权方。运行此命令后，将没有默认授权方。当您使用一个自定义授权方时，必须在 HTTP 请求标头中按名称指定它。

```
aws iot clear-default-authorizer
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 ClearDefaultAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ClearDefaultAuthorizer。

confirm-topic-rule-destination

以下代码示例演示了如何使用 confirm-topic-rule-destination。

Amazon CLI

确认主题规则目标

以下 confirm-topic-rule-destination 示例使用在 HTTP 端点收到的确认令牌来确认主题规则目标。

```
aws iot confirm-topic-rule-destination \
    --confirmation-token "AYADeIcmtq-
ZkxfpiWIQqHWM5ucAXwABABVhd3MtY3J5cHRvLXB1YmxpYy1rZXkAREFxY1E0UmlGeDg0V21BZWZ1VjZtZWFRVUJJUkt
aywpPqg8YEsa1lD4B40aJ2s1wEHKMybiF1RoOZzYisI0IvslzQY5UmCkqq3tV-3f7-
nKfosgIAAAADAAAEAAAAAAAAAAAAAAAAAAi9RMgy-
V19V9m6Iw2xfbw_____wAAAAEAAAAAAAAAAAAAAAEAAAB1hw4SokgUcxiJ3gTO6n50NLJVpzyQR1UmPIj5sShqXEQGcO
iufgrzTePl8RZYOWr0O6Aj9DiVzJZx-1iD6Pu-
G6PUw1kaO7Knzs2B4AD0qfrHUF4pYRTvyUgBnMGUCMQC8ZRmhKqntd_c6Kgrow3bMUDBvNqo2qZr8Z8Jm2rzgseROlAn
PIetJ803Z4IlIlF8xXlcdPGP-PV1dOXFemyL8g"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的确认主题规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConfirmTopicRuleDestination。

create-audit-suppression

以下代码示例演示了如何使用 create-audit-suppression。

Amazon CLI

创建审计调查发现抑制

以下 create-audit-suppression 示例为名为“virtualMachinePolicy”的策略创建了审计调查发现抑制，该策略被标记为过于宽松。

```
aws iot create-audit-suppression \
    --check-name IOT_POLICY_OVERLY_PERMISSIVE_CHECK \
    --resource-identifier
 policyVersionIdentifier={"policyName"="virtualMachinePolicy","policyVersionId"="1"}
 \
```

```
    --no-suppress-indefinitely \
    --expiration-date 2020-10-20
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计调查发现抑制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAuditSuppression。

create-authorizer

以下代码示例演示了如何使用 create-authorizer。

Amazon CLI

创建自定义授权方

以下 create-authorizer 示例创建了一个自定义授权方，该授权方使用指定的 Lambda 函数作为自定义身份验证服务的一部分。

```
   aws iot create-authorizer \
       --authorizer-name "CustomAuthorizer" \
       --authorizer-function-arn "arn:aws:lambda:us-
west-2:123456789012:function:CustomAuthorizerFunction" \
       --token-key-name "MyAuthToken" \
       --status ACTIVE \
       --token-signing-public-keys FIRST_KEY="-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1uJOB4lQPgG/lM6ZfIwo
Z+7ENxAio9q6QD4FFqjGZsvjtYwjoe1RKK0U8Eq9xb5O3kRSmyIwTzwzm/f4Gf0Y
ZUloJ+t3PUUwHrmbYTAgTrCUgRFygjfgVwGCPs5ZAX4Eyqt5cr+AIHIiUDbxSa7p
zwOBKPeic0asNJpqT8PkBbRaKyleJh5oo81NDHHmVtbBm5A5YiJjqYXLaVAowKzZ
+GqsNvAQ9Jy1wI2VrEa1OfL8flDB/BJLm7zjpfPOHDJQgID0XnZwAlNnZcOhCwIx
50g2LW2Oy9R/dmqtDmJiVP97Z4GykxPvwlYHrUXY0iW1R3AR/Ac1NhCTGZMwVDB1
lQIDAQAB
-----END PUBLIC KEY-----"
```

输出：

```
{
    "authorizerName": "CustomAuthorizer",
    "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer2"
```

```
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAuthorizer。

create-billing-group

以下代码示例演示了如何使用 create-billing-group。

Amazon CLI

创建账单组

以下 create-billing-group 示例创建一个名为 GroupOne 的简单账单组。

```
aws iot create-billing-group \
    --billing-group-name GroupOne
```

输出：

```
{
    "billingGroupName": "GroupOne",
    "billingGroupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne",
    "billingGroupId": "103de383-114b-4f51-8266-18f209ef5562"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBillingGroup。

create-certificate-from-csr

以下代码示例演示了如何使用 create-certificate-from-csr。

Amazon CLI

根据证书签名请求（CSR）创建设备证书

以下 create-certificate-from-csr 示例根据 CSR 创建设备证书。您可以使用 openssl 命令创建 CSR。

```
aws iot create-certificate-from-csr \
    --certificate-signing-request=file://certificate.csr
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/
c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbee1428d216d54d53ac9",
        "certificateId":
 "c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbee1428d216d54d53ac9",
        "certificatePem": "<certificate-text>"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateCertificateFromCSR。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCertificateFromCsr。

create-custom-metric

以下代码示例演示了如何使用 create-custom-metric。

Amazon CLI

创建由您的设备发布到 Device Defender 的自定义指标

以下 create-custom-metric 示例创建用于测量电池百分比的自定义指标。

```
aws iot create-custom-metric \
    --metric-name "batteryPercentage" \
    --metric-type "number" \
    --display-name "Remaining battery percentage." \
    --region us-east-1 \
    --client-request-token "02ccb92b-33e8-4dfa-a0c1-35b181ed26b0"
```

输出：

```
{
    "metricName": "batteryPercentage",
    "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/
batteryPercentage"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的自定义指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomMetric。

create-dimension

以下代码示例演示了如何使用 create-dimension。

Amazon CLI

创建维度

以下 create-dimension 使用称为 TopicFilterForAuthMessages 的单个主题筛选器创建一个维度。

```
aws iot create-dimension \
    --name TopicFilterForAuthMessages \
    --type TOPIC_FILTER \
    --string-values device/+/auth
```

输出：

```
{
    "name": "TopicFilterForAuthMessages",
    "arn": "arn:aws:iot:eu-west-2:123456789012:dimension/TopicFilterForAuthMessages"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDimension。

create-domain-configuration

以下代码示例演示了如何使用 create-domain-configuration。

Amazon CLI

创建域配置

以下 create-domain-configuration 示例使用服务类型 DATA 创建 Amazon 托管的域配置。

```
aws iot create-domain-configuration \
```

```
    --domain-configuration-name "additionalDataDomain" \
    --service-type "DATA"
```

输出：

```
{
    "domainConfigurationName": "additionalDataDomain",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的可配置的端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomainConfiguration。

create-dynamic-thing-group

以下代码示例演示了如何使用 create-dynamic-thing-group。

Amazon CLI

创建动态事物组

以下 create-dynamic-thing-group 示例创建了一个动态事物组，其中包含温度属性大于 60 度的任何事物。必须先启用 Amazon IoT 实例集索引，然后才能使用动态事物组。

```
aws iot create-dynamic-thing-group \
    --thing-group-name "RoomTooWarm" \
    --query-string "attributes.temperature>60"
```

输出：

```
{
    "thingGroupName": "RoomTooWarm",
    "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/RoomTooWarm",
    "thingGroupId": "9d52492a-fc87-43f4-b6e2-e571d2ffcad1",
    "indexName": "AWS_Things",
    "queryString": "attributes.temperature>60",
    "queryVersion": "2017-09-30"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的动态事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDynamicThingGroup。

create-job

以下代码示例演示了如何使用 create-job。

Amazon CLI

示例 1：创建作业

以下 create-job 示例创建了一个简单的 Amazon IoT 作业，该作业向 MyRaspberryPi 设备发送 JSON 文档。

```
aws iot create-job \
    --job-id "example-job-01" \
    --targets "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi" \
    --document file://example-job.json \
    --description "example job test" \
    --target-selection SNAPSHOT
```

输出：

```
{
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
    "jobId": "example-job-01",
    "description": "example job test"
}
```

示例 2：创建连续作业

以下 create-job 示例创建了一个可在指定为目标的事物完成作业后继续运行的作业。在此示例中，目标是一个事物组，因此，当新设备被添加到该组时，连续作业将在这些新事物上运行。

aws iot create-job --job-id "example-job-04" --targets "arn:aws:iot:us-west-2:123456789012:thinggroup/DeadBulbs" --document file://example-job.json --description "example continuous job" --target-selection CONTINUOUS

输出：

```
{
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-04",
    "jobId": "example-job-04",
```

```
    "description": "example continuous job"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJob。

create-keys-and-certificate

以下代码示例演示了如何使用 create-keys-and-certificate。

Amazon CLI

创建 RSA 密钥对并发放 X.509 证书

以下 create-keys-and-certificate 命令创建一个 2048 位 RSA 密钥对，并使用发出的公钥发放 X.509 证书。因为这是 Amazon物联网唯一一次为此证书提供私钥，所以请务必将其保存在安全的地方。

```
aws iot create-keys-and-certificate \
    --certificate-pem-outfile "myTest.cert.pem" \
    --public-key-outfile "myTest.public.key" \
    --private-key-outfile "myTest.private.key"
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/9894ba17925e663f1d29c23af4582b8e3b7619c31f3fbd93adcb51ae54b83dc2",
    "certificateId":
 "9894ba17925e663f1d29c23af4582b8e3b7619c31f3fbd93adcb51ae54b83dc2",
    "certificatePem": "
-----BEGIN CERTIFICATE-----
MIICiTCCEXAMPLE6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgEXAMPLEAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSEXAMPLE2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYEXAMPLEb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCEXAMPLEJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDAEXAMPLEsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEXAMPLE25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+aEXAMPLE
EXAMPLEfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
```

```
rDHudUZEXAMPLELG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAEXAMPLEWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9qEXAMPLEyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDEXAMPLEBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----\n",
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkEXAMPLEQEFAAOCAQ8AMIIBCgKCAQEAEXAMPLE1nnyJwKSMHw4h\nMMEXAMPLEuuN/
dMAS3fyce8DW/4+EXAMPLEyjmoF/YVF/gHr99VEEXAMPLE5VF13\n59VK7cEXAMPLE67GK+y+jikqXOgHh/
xJTwo
+sGpWEXAMPLEDz18xOd2ka4tCzuWEXAMPLEahJbYkCPUBSU8opVkR7qkEXAMPLE1DR6sx2HocliOOLtu6Fkw91swQWEX
\GB3ZPrNh0PzQYvjUStZeccyNCx2EXAMPLEvp9mQOUXP6plfgxwKRX2fEXAMPLEDa
\nhJLXkX3rHU2xbxJSq7D+XEXAMPLEcw+LyFhI5mgFRl88eGdsAEXAMPLElnI9EesG\nFQIDAQAB\n-----
END PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----\nkey omittted for security
 reasons\n-----END RSA PRIVATE KEY-----\n"
    }
}
```

有关更多信息，请参阅《Amazon物联网开发人员指南》中的创建和注册 Amazon物联网设备证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKeysAndCertificate。

create-mitigation-action

以下代码示例演示了如何使用 create-mitigation-action。

Amazon CLI

创建缓解操作

以下 create-mitigation-action 示例定义了一个名为 AddThingsToQuarantineGroup1Action 的缓解操作，当应用该操作时，事物将被移入名为 QuarantineGroup1 的事物组中。此操作将覆盖动态事物组。

```
aws iot create-mitigation-action --cli-input-json file::params.json
```

的内容params.json：

```
{
```

```
    "actionName": "AddThingsToQuarantineGroup1Action",
    "actionParams": {
        "addThingsToThingGroupParams": {
            "thingGroupNames": [
                "QuarantineGroup1"
            ],
            "overrideDynamicGroups": true
        }
    },
    "roleArn": "arn:aws:iam::123456789012:role/service-role/
MoveThingsToQuarantineGroupRole"
}
```

输出：

```
{
    "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroup1Action",
    "actionId": "992e9a63-a899-439a-aa50-4e20c52367e1"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 CreateMitigationAction（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMitigationAction。

create-ota-update

以下代码示例演示了如何使用 create-ota-update。

Amazon CLI

创建用于 Amazon FreeRTOS 的 OTA 更新

以下 create-ota-update 示例在目标事物组上创建 Amazon IoT OTAUpdate。这是 Amazon FreeRTOS 无线更新的一部分，它使您可以将新的固件镜像部署到单个设备或一组设备。

```
aws iot create-ota-update \
    --cli-input-json file://create-ota-update.json
```

的内容create-ota-update.json：

```
{
    "otaUpdateId": "ota12345",
    "description": "A critical update needed right away.",
    "targets": [
        "device1",
        "device2",
        "device3",
        "device4"
    ],
    "targetSelection": "SNAPSHOT",
    "awsJobExecutionsRolloutConfig": {
        "maximumPerMinute": 10
    },
    "files": [
        {
          "fileName": "firmware.bin",
          "fileLocation": {
            "stream": {
              "streamId": "004",
              "fileId":123
            }
          },
          "codeSigning": {
            "awsSignerJobId": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
          }
        }
    ]
    "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_role"
}
```

输出：

```
{
     "otaUpdateId": "ota12345",
     "awsIotJobId": "job54321",
     "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/itsaupdate",
     "awsIotJobArn": "arn:aws:iot:us-west-2:123456789012:job/itsajob",
     "otaUpdateStatus": "CREATE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateOTAUpdate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOtaUpdate。

create-policy-version

以下代码示例演示了如何使用 create-policy-version。

Amazon CLI

使用新版本更新策略

以下 create-policy-version 示例更新了策略定义，创建了新的策略版本。此示例还将新版本设为默认版本。

```
aws iot create-policy-version \
    --policy-name UpdateDeviceCertPolicy \
    --policy-document file://policy.json \
    --set-as-default
```

的内容policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action":  "iot:UpdateCertificate",
            "Resource": "*"
        }
    ]
}
```

输出：

```
{
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
    "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\":
 \"Allow\", \"Action\":  \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
    "policyVersionId": "2",
    "isDefaultVersion": true
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicyVersion。

create-policy

以下代码示例演示了如何使用 create-policy。

Amazon CLI

创建 Amazon IoT 策略

以下 create-policy 示例创建了一个名为 TemperatureSensorPolicy 的 Amazon IoT 策略。该 policy.json 文件包含允许 Amazon IoT 策略操作的语句。

```
aws iot create-policy \
    --policy-name TemperatureSensorPolicy \
    --policy-document file://policy.json
```

的内容policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:Publish",
                "iot:Receive"
            ],
            "Resource": [
                "arn:aws:iot:us-west-2:123456789012:topic/topic_1",
                "arn:aws:iot:us-west-2:123456789012:topic/topic_2"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iot:Subscribe"
            ],
            "Resource": [
                "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_1",
                "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_2"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "iot:Connect"
            ],
            "Resource": [
                "arn:aws:iot:us-west-2:123456789012:client/basicPubSub"
            ]
        }
    ]
}
```

输出：

```
{
    "policyName": "TemperatureSensorPolicy",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TemperatureSensorPolicy",
    "policyDocument": "{
        \"Version\": \"2012-10-17\",
        \"Statement\": [
            {
                \"Effect\": \"Allow\",
                \"Action\": [
                    \"iot:Publish\",
                    \"iot:Receive\"
                ],
                \"Resource\": [
                    \"arn:aws:iot:us-west-2:123456789012:topic/topic_1\",
                    \"arn:aws:iot:us-west-2:123456789012:topic/topic_2\"
                ]
            },
            {
                \"Effect\": \"Allow\",
                \"Action\": [
                    \"iot:Subscribe\"
                ],
                \"Resource\": [
                    \"arn:aws:iot:us-west-2:123456789012:topicfilter/topic_1\",
                    \"arn:aws:iot:us-west-2:123456789012:topicfilter/topic_2\"
                ]
            },
            {
                \"Effect\": \"Allow\",
                \"Action\": [
```

```
                    \"iot:Connect\"
                ],
                \"Resource\": [
                    \"arn:aws:iot:us-west-2:123456789012:client/basicPubSub\"
                ]
            }
        ]
    }",
    "policyVersionId": "1"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicy。

create-provisioning-claim

以下代码示例演示了如何使用 create-provisioning-claim。

Amazon CLI

创建预配声明

以下 create-provisioning-claim 示例根据预配模板创建预配声明。

```
aws iot create-provisioning-claim \
    --template-name MyTestProvisioningTemplate
```

输出：

```
{
    "certificateId":
 "78de02184b2ce80cf8fb709bda59e62b19fb83513590483eb0434589476ab09f",
    "certificatePem": "-----BEGIN CERTIFICATE-----\nMIIDdzCCAl
+gAwIBAgIUXSZhEBLztMLZ2fHG
14gV0NymYY0wDQYJKoZIhvcNAQEL
\nBQAwfjELMAkGA1UEBhMCVVMxEzARBgNVBAgMCldhc2hpbmd0b24xEDAOBg
VBAcM\nB1NlYXR0bGUxGDAWBgNVBAoMD0FtYXpvbi5jb20gSW5jLjEgMB4GA1UECwwXQW1h
\nem9uIElvVCBQcm9
2aXNpb25pbmcxDDAKBgNVBAUTAzEuMDAeFw0yMDA3MjgxNjQ0\nMDZaFw0yMDA3MjgxNjUxMDZaMEsxSTBHBgNVB
AMMQDFhNDEyM2VkNmIxYjU3MzE3\nZTgzMTJmY2MzN2FiNTdhY2MzYTZkZGVjOGQ5OGY3NzUwMWRlMjc0YjhmYTQ
xN2Iw\nggEiMA0GCSqGSIb3EXAMPLEAA4IBDwAwggEKAoIBAQDBhKI94ktKLqTwnj+ayOq1\nTAJt/
N6s6IJDZvl
```

```
rYjkC0E7wzaeY3TprWk03S29vUzVuEOXHXQXZbihgpg2m6fza\nkWm9/
wpjzE9ny5+xkPGVH4Wnwz7yK5m8S0agL
T96cRBSWnWmonOWdY0GKVzni0CA\n+iyGudgrFKm7Eae/
v18oXrf82KtOAGO4xG0KE2WKYHsT1fx3c9xZhlXP/eX
Lhv00\n+lGp0WVw9PbhKfrxliKJ5q6sL5nVUaUHq6hlQPYwsATeOvAp3u0ak5zgTyL0fg7Y
\nPyKk6VYwLW62r+V
YBSForEMOAhkq3LsP/rjxpEKmi2W4lPVS6oFZRKcD+H1Kyil5\nAgMBAAGjIDAeMAwGA1UdEwEB/
wQCMAAwDgYDV
R0PAQH/BAQDAgeAMA0GCSqGSIb3\nDQEBCwUAA4IBAQAGgix2k6nVqbZFKq97/fZBzLGS0dyz5rT/
E41cDIRX+1j
EPW41\nw0D+2sXheCZLZZnSkvIiP74IToNeXDrjdcaodeGFVHIElRjhMIq+4ZebPbRLtidF
\nRc2hfcTAlqq9Z6v
5Vk6BeM1tu0RqH1wPoVUccLPya8EjNCbnJZUmGdOfrN/Y9pho\n5ikV+HPeZhG/k6dhE2GsQJyKFVHL/
uBgKSily
1bRyWU1r6qcpWBNBHjUoD7HgOwD
\nnzMh4XRb2FQDsqFalkCSYmeL8IVC49sgPD9Otyp5uteGMTy62usAAUQdq/f
ZvrWg\nOkFpwMVnGKVKT7Kg0kKOLzKWOBB2Jm4/gmrJ\n-----END CERTIFICATE-----\n",
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEAwYSiPeJLSi6k8J4/msjq
\ntUwCbfzerOiCQ2b5a2I5AtBO8M2nmN06a1pNN0tvb1M1bhDlx10F2W4oYKYN
pun8\n2pFpvf8KY8xPZ8ufsZDxlR+Fp8M+8iuZvEtGoC0/enEQUlp1pqJzlnWNBilc54tA
\ngPoshrnYKxSpuxGn
v79fKF63/NirTgBjuMRtChNlimEXAMPLE3PcWYZVz/3ly4b9\nNPpRqdFlcPT24Sn68ZYiieaurC
+Z1VGlB6uoZU
D2MLAE3jrwKd7tGpOc4E8i9H4O\n2D8ipOlWMC1utq/
lWAUhaKxDDgIZKty7D/648aRCpotluJT1UuqBWUSnA/h9
Ssop\neQIDAQAB\n-----END PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIBAAKCAQEAwYSiPeJLSi6k8J4/
msjqtUwCbfzerOiCQ2b5a2I5AtBO8M2n
\nmN06a1pNN0tvb1M1bhDlx10F2W4oYKYNpun82pFpvf8KY8xPZ8ufsZ
DxlR+Fp8M+\n8iuZvEtGoC0/enEQUlp1pqJzlnWNBilc54tAgPoshrnYKxSpuxGnv79fKF63/Nir
\nTgBjuMRtCh
NlimB7E9X8d3PcWYZVz/3ly4b9NPpRqdFlcPT24Sn68ZYiieaurC+Z
\n1VGlB6uoZUD2MLAE3jrwKd7tGpOc4E8i
9H4O2D8ipOlWMC1utq/lWAUhaKxDDgIZ\nKty7D/648aRCpotluJT1UuqBWUSnA/
h9SsopeQIDAQABAoIBAEAybN
QUtx9T2/nK\ntZT2pA4iugecxI4dz+DmT0XVXs5VJmrx/
nBSq6ejXExEpSIMO4RY7LE3ZdJcnd56\nF7tQkkY7yR
VzfxHeXFU1krOIPuxWebNOrRoPZr+1RSer+wv2aBC525+88pVuR6tM
\nm3pgkrR2ycCj9FdOUoQxdjHBHaM5PDmJ
```

```
9aSxCKdg3nReepeGwsR2TQA+m2vVxWk7\nou0+91eTOP+/QfP7P8ZjOIkO2XivlRcVDyN/
E4QXPKuIkM/8vS8VK+
E9pATQ0MtB\n2lw8R/YU5AJd6jlEXAMPLEGU2UzRzInNWiLtkPPPqgqXXhxOf+mxByjcMalVJk0L
\nhOG2ROUCgY
EA+ROcHNHy/XbsP7FihOhEh+6Q2QxQ2ncBUPYbBazrR8Hn+7SCICQK
\nVyYfd8Ajfq3e7RsKVL5SlMBp7Slidxak
bIn28fKfPn62DaemGCIoyDgLpF+eUxBx
\ngzbCiBZga8brfurza43UZjKZLpg3hq721+FeAiXi1Nma4Yr9YWEHEN
8CgYEAxuWt\npzdWWmsiFzfsAw0sy9ySDA/xr5WRWzJyAqUsjsks6rxNzWebpufnYHcmtW7pLdqM
\nkboHwN2pXa
kmZvrk2nKkEMq5brBYGDXuxDe+V369Bianx8aZFyIsckA7OwXW1w1h
\ngRC5rQ4XOgp3+Jmw7eAO8LRYDjaN846+
QbtO2KcCgYAWS0UL51bijQR0ZwI0dz27\nFQVuCAYsp748aurcRTACCj8jbnK/
QbqTNlxWsaH7ssBjZKo2D5sAqY
BRtASWODab\naHXsDhVm2Jye+ESLoHMaCLoyCkT3ll8yqXIcEDStMO7fO1Ryag164EiJvSIrMfny\nNL/
fXVjCSH
/udCxdzPt+7QKBgQC+LAD7rxdr4J9538hTqpc4XK9vxRbrMXEH55XH
\nHbMa2xONZXpmeTgEQBukyohCVceyRhK9
i0e6irZTjVXghOeoTpC8VXkzcnzouTiQ
\neFQQSGfnp7Ioe6UIz23715pKduzSNkMSKrG924ktv7CyDBF1gBQI5g
aDoHnddJBJ\nPRTIZQKBgA8MASXtTxQntRwXXzR92U0vAighiuRkB/mx9jQpUcK1qiqHbkAMqgNF
\nPFCBYIUbFT
iYKKKeJNbyJQvjfsJCkAnaFJ+RnTxk0Q6Wjm20peJ/ii4QiDdnigoE\nvdlc5cFQewWb4/
zqAtPdinkPlN94ileI
79XQdc7RlJ0jpgTimL+V\n-----END RSA PRIVATE KEY-----\n"
    },
    "expiration": 1595955066.0
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的通过可信用户预配。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProvisioningClaim。

create-provisioning-template-version

以下代码示例演示了如何使用 create-provisioning-template-version。

Amazon CLI

创建预配模板版本

以下示例为指定预配模板创建了一个版本。文件 template.json 中提供了新版本的正文。

```
aws iot create-provisioning-template-version \
```

```
    --template-name widget-template \
    --template-body file://template.json
```

的内容template.json：

```
{
    "Parameters" : {
        "DeviceLocation": {
            "Type": "String"
        }
    },
    "Mappings": {
        "LocationTable": {
            "Seattle": {
                "LocationUrl": "https://example.aws"
            }
        }
    },
    "Resources" : {
        "thing" : {
            "Type" : "AWS::IoT::Thing",
            "Properties" : {
                "AttributePayload" : {
                    "version" : "v1",
                    "serialNumber" : "serialNumber"
                },
                "ThingName" : {"Fn::Join":["",["ThingPrefix_",
{"Ref":"SerialNumber"}]]},
                "ThingTypeName" : {"Fn::Join":["",["ThingTypePrefix_",
{"Ref":"SerialNumber"}]]},
                "ThingGroups" : ["widgets", "WA"],
                "BillingGroup": "BillingGroup"
            },
            "OverrideSettings" : {
                "AttributePayload" : "MERGE",
                "ThingTypeName" : "REPLACE",
                "ThingGroups" : "DO_NOTHING"
            }
        },
        "certificate" : {
            "Type" : "AWS::IoT::Certificate",
            "Properties" : {
                "CertificateId": {"Ref": "AWS::IoT::Certificate::Id"},
```

```
                "Status" : "Active"
            }
        },
        "policy" : {
            "Type" : "AWS::IoT::Policy",
            "Properties" : {
                "PolicyDocument" : {
                    "Version":"2012-10-17",
                    "Statement": [{
                        "Effect": "Allow",
                        "Action":["iot:Publish"],
                        "Resource": ["arn:aws:iot:us-east-1:123456789012:topic/foo/
bar"]
                    }]
                }
            }
        }
    },
    "DeviceConfiguration": {
        "FallbackUrl": "https://www.example.com/test-site",
        "LocationUrl": {
            "Fn::FindInMap": ["LocationTable",{"Ref": "DeviceLocation"},
 "LocationUrl"]}
        }
    }
}
```

输出：

```
{
    "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/widget-
template",
    "templateName": "widget-template",
    "versionId": 2,
    "isDefaultVersion": false
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProvisioningTemplateVersion。

create-provisioning-template

以下代码示例演示了如何使用 create-provisioning-template。

Amazon CLI

创建预配模板

以下 create-provisioning-template 示例按照文件 template.json 的定义创建预配模板。

```
aws iot create-provisioning-template \
    --template-name widget-template \
    --description "A provisioning template for widgets" \
    --provisioning-role-arn arn:aws:iam::123456789012:role/Provision_role \
    --template-body file://template.json
```

的内容template.json：

```
{
    "Parameters" : {
        "DeviceLocation": {
            "Type": "String"
        }
    },
    "Mappings": {
        "LocationTable": {
            "Seattle": {
                "LocationUrl": "https://example.aws"
            }
        }
    },
    "Resources" : {
        "thing" : {
            "Type" : "AWS::IoT::Thing",
            "Properties" : {
                "AttributePayload" : {
                    "version" : "v1",
                    "serialNumber" : "serialNumber"
                },
                "ThingName" : {"Fn::Join":["",["ThingPrefix_",
{"Ref":"SerialNumber"}]]},
```

```
                "ThingTypeName" : {"Fn::Join":["",["ThingTypePrefix_",
{"Ref":"SerialNumber"}]]},
                "ThingGroups" : ["widgets", "WA"],
                "BillingGroup": "BillingGroup"
            },
            "OverrideSettings" : {
                "AttributePayload" : "MERGE",
                "ThingTypeName" : "REPLACE",
                "ThingGroups" : "DO_NOTHING"
            }
        },
        "certificate" : {
            "Type" : "AWS::IoT::Certificate",
            "Properties" : {
                "CertificateId": {"Ref": "AWS::IoT::Certificate::Id"},
                "Status" : "Active"
            }
        },
        "policy" : {
            "Type" : "AWS::IoT::Policy",
            "Properties" : {
                "PolicyDocument" : {
                    "Version":"2012-10-17",
                    "Statement": [{
                        "Effect": "Allow",
                        "Action":["iot:Publish"],
                        "Resource": ["arn:aws:iot:us-east-1:504350838278:topic/foo/
bar"]
                    }]
                }
            }
        }
    },
    "DeviceConfiguration": {
        "FallbackUrl": "https://www.example.com/test-site",
        "LocationUrl": {
            "Fn::FindInMap": ["LocationTable",{"Ref": "DeviceLocation"},
 "LocationUrl"]}
        }
    }
}
```

输出：

```
{
    "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/widget-
template",
    "templateName": "widget-template",
    "defaultVersionId": 1
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProvisioningTemplate。

create-role-alias

以下代码示例演示了如何使用 create-role-alias。

Amazon CLI

创建角色别名

以下 create-role-alias 示例为指定角色创建名为 LightBulbRole 的角色别名。

```
aws iot create-role-alias \
    --role-alias LightBulbRole \
    --role-arn arn:aws:iam::123456789012:role/lightbulbrole-001
```

输出：

```
{
    "roleAlias": "LightBulbRole",
    "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/LightBulbRole"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateRoleAlias。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoleAlias。

create-scheduled-audit

以下代码示例演示了如何使用 create-scheduled-audit。

Amazon CLI

创建预定审计

以下 create-scheduled-audit 示例创建了一个预定审计，该审计每周星期三运行，以检查 CA 证书或设备证书是否即将过期。

```
aws iot create-scheduled-audit \
    --scheduled-audit-name WednesdayCertCheck \
    --frequency WEEKLY \
    --day-of-week WED \
    --target-check-
names CA_CERTIFICATE_EXPIRING_CHECK DEVICE_CERTIFICATE_EXPIRING_CHECK
```

输出：

```
{
    "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
WednesdayCertCheck"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateScheduledAudit。

create-security-profile

以下代码示例演示了如何使用 create-security-profile。

Amazon CLI

创建安全配置文件

以下 create-security-profile 示例创建了一个安全配置文件，用于检查蜂窝带宽是否超过阈值，或者在五分钟内是否出现超过 10 次授权失败。

```
aws iot create-security-profile \
    --security-profile-name PossibleIssue \
    --security-profile-description "Check to see if authorization fails 10 times in
 5 minutes or if cellular bandwidth exceeds 128"  \
    --behaviors "[{\"name\":\"CellularBandwidth\",\"metric\":\"aws:message-byte-size
\",\"criteria\":{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":128},
```

```
\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}},{\"name
\":\"Authorization\",\"metric\":\"aws:num-authorization-failures\",\"criteria\":
{\"comparisonOperator\":\"less-than\",\"value\":{\"count\":10},\"durationSeconds
\":300,\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}}]"
```

输出：

```
{
    "securityProfileName": "PossibleIssue",
    "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
PossibleIssue"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSecurityProfile。

create-stream

以下代码示例演示了如何使用 create-stream。

Amazon CLI

创建一个流以便通过 MQTT 以块的形式传送一个或多个大文件

以下 create-stream 示例创建一个流，以便通过 MQTT 以块的形式传送一个或多个大文件。流将来自 S3 之类的源的数据字节打包为 MQTT 消息，以块或数据块的形式传输。您可以将一个或多个文件与一个流关联。

```
aws iot create-stream \
    --cli-input-json file://create-stream.json
```

的内容create-stream.json：

```
{
    "streamId": "stream12345",
    "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
    "files": [
        {
            "fileId": 123,
            "s3Location": {
```

```
                "bucket":"codesign-ota-bucket",
                "key":"48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
            }
        }
    ],
    "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_stream_role"
}
```

输出：

```
{
     "streamId": "stream12345",
     "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
     "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
     "streamVersion": "1"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateStream。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStream。

create-thing-group

以下代码示例演示了如何使用 create-thing-group。

Amazon CLI

示例 1：创建事物组

以下 create-thing-group 示例使用一个描述和两个属性创建了一个名为 LightBulbs 的事物组。

```
aws iot create-thing-group \
    --thing-group-name LightBulbs \
    --thing-group-properties "thingGroupDescription=\"Generic bulb group\",
 attributePayload={attributes={Manufacturer=AnyCompany,wattage=60}}"
```

输出：

```
{
    "thingGroupName": "LightBulbs",
```

```
    "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs",
    "thingGroupId": "9198bf9f-1e76-4a88-8e8c-e7140142c331"
}
```

示例 2：创建属于父组的事物组

以下 create-thing-group 示例创建了一个名为 HalogenBulbs 的事物组，其父事物组名为 LightBulbs。

```
aws iot create-thing-group \
    --thing-group-name HalogenBulbs \
    --parent-group-name LightBulbs
```

输出：

```
{
    "thingGroupName": "HalogenBulbs",
    "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs",
    "thingGroupId": "f4ec6b84-b42b-499d-9ce1-4dbd4d4f6f6e"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateThingGroup。

create-thing-type

以下代码示例演示了如何使用 create-thing-type。

Amazon CLI

定义事物类型

以下 create-thing-type 示例定义事物类型和关联的属性。

```
aws iot create-thing-type \
    --thing-type-name "LightBulb" \
    --thing-type-properties "thingTypeDescription=light bulb type,
 searchableAttributes=wattage,model"
```

输出：

```
{
    "thingTypeName": "LightBulb",
    "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/LightBulb",
    "thingTypeId": "ce3573b0-0a3c-45a7-ac93-4e0ce14cd190"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateThingType。

create-thing

以下代码示例演示了如何使用 create-thing。

Amazon CLI

示例 1：在注册表中创建事物记录

以下 create-thing 示例在 Amazon物联网事物注册表中为设备创建条目。

```
aws iot create-thing \
    --thing-name SampleIoTThing
```

输出：

```
{
    "thingName": "SampleIoTThing",
    "thingArn": "arn:aws:iot:us-west-2: 123456789012:thing/SampleIoTThing",
    "thingId": " EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE "
}
```

示例 2：定义与事物类型关联的事物

以下 create-thing 示例创建了一个事物，该事物具有指定事物类型及其属性。

```
aws iot create-thing \
    --thing-name "MyLightBulb" \
    --thing-type-name "LightBulb" \
    --attribute-payload "{"attributes": {"wattage":"75", "model":"123"}}"
```

输出：

```
{
    "thingName": "MyLightBulb",
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
    "thingId": "40da2e73-c6af-406e-b415-15acae538797"
}
```

有关更多信息，请参阅《Amazon物联网开发人员治安》中的如何使用注册表管理事物和事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateThing。

`create-topic-rule-destination`

以下代码示例演示了如何使用 `create-topic-rule-destination`。

Amazon CLI

创建主题规则目标

以下 `create-topic-rule-destination` 示例为 HTTP 端点创建主题规则目标。

```
aws iot create-topic-rule-destination \
    --destination-configuration httpUrlConfiguration={confirmationUrl=https://
example.com}
```

输出：

```
{
    "topicRuleDestination": {
        "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "status": "IN_PROGRESS",
        "statusReason": "Awaiting confirmation. Confirmation message sent on
 2020-07-09T22:47:54.154Z; no response received from the endpoint.",
        "httpUrlProperties": {
            "confirmationUrl": "https://example.com"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建主题规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTopicRuleDestination。

create-topic-rule

以下代码示例演示了如何使用 create-topic-rule。

Amazon CLI

创建发送 Amazon SNS 警报的规则

以下 create-topic-rule 示例创建了一条规则，当设备影子中发现土壤湿度读数较低时，该规则就会发送 Amazon SNS 消息。

```
aws iot create-topic-rule \
    --rule-name "LowMoistureRule" \
    --topic-rule-payload file://plant-rule.json
```

该示例要求将以下 JSON 代码保存到名为 plant-rule.json 的文件中：

```
{
    "sql": "SELECT * FROM '$aws/things/MyRPi/shadow/update/accepted' WHERE
 state.reported.moisture = 'low'\n",
    "description": "Sends an alert whenever soil moisture level readings are too
 low.",
    "ruleDisabled": false,
    "awsIotSqlVersion": "2016-03-23",
    "actions": [{
            "sns": {
                "targetArn": "arn:aws:sns:us-
west-2:123456789012:MyRPiLowMoistureTopic",
                "roleArn": "arn:aws:iam::123456789012:role/service-role/
MyRPiLowMoistureTopicRole",
                "messageFormat": "RAW"
            }
    }]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon物联网开发人员指南》中的创建 Amazon物联网规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTopicRule。

delete-account-audit-configuration

以下代码示例演示了如何使用 delete-account-audit-configuration。

Amazon CLI

禁用针对 Amazon 账户的所有审计检查

以下 delete-account-audit-configuration 示例恢复该账户的 Amazon IoT Device Defender 的默认设置，禁用所有审计检查并清除配置数据。它还删除针对此账户的所有预定审计。请谨慎使用此命令。

```
aws iot delete-account-audit-configuration \
    --delete-scheduled-audits
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccountAuditConfiguration。

delete-audit-suppression

以下代码示例演示了如何使用 delete-audit-suppression。

Amazon CLI

删除审计调查发现抑制

以下 delete-audit-suppression 示例删除了 DEVICE_CERTIFICATE_EXPIRING_CHECK 的审计调查发现抑制。

```
aws iot delete-audit-suppression \
    --check-name DEVICE_CERTIFICATE_EXPIRING_CHECK \
    --resource-identifier deviceCertificateId="c7691e<shortened>"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计调查发现抑制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAuditSuppression。

delete-authorizer

以下代码示例演示了如何使用 delete-authorizer。

Amazon CLI

删除自定义授权方

以下 delete-authorizer 示例删除名为 CustomAuthorizer 的授权方。自定义授权方必须处于 INACTIVE 状态才能被删除。

```
aws iot delete-authorizer \
    --authorizer-name CustomAuthorizer
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 DeleteAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAuthorizer。

delete-billing-group

以下代码示例演示了如何使用 delete-billing-group。

Amazon CLI

删除账单组

以下 delete-billing-group 示例删除指定账单组。即使账单组包含一个或多个事物，您也可以将其删除。

```
aws iot delete-billing-group \
    --billing-group-name BillingGroupTwo
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBillingGroup。

delete-ca-certificate

以下代码示例演示了如何使用 delete-ca-certificate。

Amazon CLI

删除 CA 证书

以下 delete-ca-certificate 示例删除具有指定证书 ID 的 CA 证书。

```
aws iot delete-ca-certificate \
    --certificate-
id f4efed62c0142f16af278166f61962501165c4f053629520742646005 8cd1467
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 DeleteCACertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCaCertificate。

delete-certificate

以下代码示例演示了如何使用 delete-certificate。

Amazon CLI

删除设备证书

以下 delete-certificate 示例删除具有指定 ID 的设备证书。

```
aws iot delete-certificate \
    --certificate-
id c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbee1428d216d54d53ac9
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API Reference》中的 DeleteCertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCertificate。

delete-custom-metric

以下代码示例演示了如何使用 delete-custom-metric。

Amazon CLI

删除自定义指标

以下 delete-custom-metric 示例删除一个自定义指标。

```
aws iot delete-custom-metric \
    --metric-name batteryPercentage \
    --region us-east-1
```

输出：

```
HTTP 200
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的自定义指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCustomMetric。

delete-dimension

以下代码示例演示了如何使用 delete-dimension。

Amazon CLI

删除维度

以下 delete-dimension 示例删除名为 TopicFilterForAuthMessages 的维度。

```
aws iot delete-dimension \
    --name TopicFilterForAuthMessages
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDimension。

delete-domain-configuration

以下代码示例演示了如何使用 delete-domain-configuration。

Amazon CLI

删除域配置

以下 delete-domain-configuration 示例从您的 Amazon 账户中删除名为 additionalDataDomain 的域配置。

```
aws iot delete-domain-configuration \
    --domain-configuration-name "additionalDataDomain" \
    --domain-configuration-status "OK"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的可配置的端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomainConfiguration。

delete-dynamic-thing-group

以下代码示例演示了如何使用 delete-dynamic-thing-group。

Amazon CLI

删除动态事物组

以下 delete-dynamic-thing-group 示例删除指定动态事物组。

```
aws iot delete-dynamic-thing-group \
    --thing-group-name "RoomTooWarm"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的动态事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDynamicThingGroup。

delete-job-execution

以下代码示例演示了如何使用 delete-job-execution。

Amazon CLI

删除作业执行

以下 delete-job-execution 示例删除在设备上执行指定作业。使用 describe-job-execution 获取执行编号。

```
aws iot delete-job-execution
    --job-id "example-job-02"
    --thing-name "MyRaspberryPi"
    --execution-number 1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteJobExecution。

delete-job

以下代码示例演示了如何使用 delete-job。

Amazon CLI

删除作业

以下 delete-job 示例删除指定作业。通过指定 --force 选项，即使作业的状态为 IN_PROGRESS，也会被删除。

```
aws iot delete-job \
    --job-id "example-job-04" \
    --force
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteJob。

delete-mitigation-action

以下代码示例演示了如何使用 delete-mitigation-action。

Amazon CLI

删除缓解操作

以下 delete-mitigation-action 示例删除指定的缓解操作。

```
aws iot delete-mitigation-action \
```

```
    --action-name AddThingsToQuarantineGroup1Action
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 DeleteMitigationAction（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMitigationAction。

delete-ota-update

以下代码示例演示了如何使用 delete-ota-update。

Amazon CLI

删除 OTA 更新

以下 delete-ota-update 示例删除指定 OTA 更新。

```
aws iot delete-ota-update \
    --ota-update-id ota12345 \
    --delete-stream \
    --force-delete-aws-job
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 DeleteOTAUpdate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOtaUpdate。

delete-policy-version

以下代码示例演示了如何使用 delete-policy-version。

Amazon CLI

删除策略的某个版本

以下 delete-policy-version 示例从您的 Amazon 账户中删除指定策略的版本 2。

```
aws iot delete-policy-version \
    --policy-name UpdateDeviceCertPolicy \
    --policy-version-id 2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicyVersion。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除策略

以下 delete-policy 示例从您的 Amazon 账户删除指定策略。

```
aws iot delete-policy --policy-name UpdateDeviceCertPolicy
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

delete-provisioning-template-version

以下代码示例演示了如何使用 delete-provisioning-template-version。

Amazon CLI

删除预配模板版本

以下 delete-provisioning-template-version 示例删除指定预配模板的版本 2。

```
aws iot delete-provisioning-template-version \
    --version-id 2 \
    --template-name "widget-template"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProvisioningTemplateVersion。

delete-provisioning-template

以下代码示例演示了如何使用 delete-provisioning-template。

Amazon CLI

删除预配模板

以下 delete-provisioning-template 示例删除指定预配模板。

```
aws iot delete-provisioning-template \
    --template-name widget-template
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProvisioningTemplate。

delete-registration-code

以下代码示例演示了如何使用 delete-registration-code。

Amazon CLI

删除您的注册码

以下 delete-registration-code 示例删除 Amazon IoT 特定账户的注册码。

```
aws iot delete-registration-code
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用您自己的证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRegistrationCode。

delete-role-alias

以下代码示例演示了如何使用 delete-role-alias。

Amazon CLI

删除 Amazon IoT 角色别名

以下 delete-role-alias 示例删除名为 LightBulbRole 的 Amazon IoT 角色别名。

```
aws iot delete-role-alias \
    --role-alias LightBulbRole
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的授权直接调用 Amazon 服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoleAlias。

delete-scheduled-audit

以下代码示例演示了如何使用 delete-scheduled-audit。

Amazon CLI

删除预定审计

以下 delete-scheduled-audit 示例删除名为 AWSIoTDeviceDefenderDailyAudit 的 Amazon IoT Device Defender 预定审计。

```
aws iot delete-scheduled-audit \
    --scheduled-audit-name AWSIoTDeviceDefenderDailyAudit
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScheduledAudit。

delete-security-profile

以下代码示例演示了如何使用 delete-security-profile。

Amazon CLI

删除安全配置文件

以下 delete-security-profile 示例删除名为 PossibleIssue 的安全配置文件。

```
aws iot delete-security-profile \
```

```
    --security-profile-name PossibleIssue
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSecurityProfile。

delete-stream

以下代码示例演示了如何使用 delete-stream。

Amazon CLI

删除流

以下 delete-stream 示例删除指定流。

```
aws iot delete-stream \
    --stream-id stream12345
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 DeleteStream。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStream。

delete-thing-group

以下代码示例演示了如何使用 delete-thing-group。

Amazon CLI

删除事物组

以下 delete-thing-group 示例删除指定事物组。如果事物组中包含子事物组，则无法将其删除。

```
aws iot delete-thing-group \
    --thing-group-name DefectiveBulbs
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteThingGroup。

delete-thing-type

以下代码示例演示了如何使用 delete-thing-type。

Amazon CLI

示例 1：删除事物类型

以下 delete-thing-type 示例删除一个已弃用的事物类型。

```
aws iot delete-thing-type \
    --thing-type-name "obsoleteThingType"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteThingType。

delete-thing

以下代码示例演示了如何使用 delete-thing。

Amazon CLI

显示有关事物的详细信息

以下 delete-thing 示例从您的 Amazon 账户的 Amazon物联网注册表中删除一个事物。

aws iot delete-thing --thing-name "FourthBulb"

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的如何使用注册表管理事物。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteThing。

delete-topic-rule-destination

以下代码示例演示了如何使用 delete-topic-rule-destination。

Amazon CLI

删除主题规则目标

以下 delete-topic-rule-destination 示例删除指定的主题规则目标。

```
aws iot delete-topic-rule-destination \
    --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的删除主题规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTopicRuleDestination。

delete-topic-rule

以下代码示例演示了如何使用 delete-topic-rule。

Amazon CLI

删除规则

以下 delete-topic-rule 示例删除指定规则。

```
aws iot delete-topic-rule \
    --rule-name "LowMoistureRule"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon物联网开发人员指南》中的删除规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTopicRule。

delete-v2-logging-level

以下代码示例演示了如何使用 delete-v2-logging-level。

Amazon CLI

删除事物组的日志级别

以下 delete-v2-logging-level 示例删除指定事物组的日志级别。

```
aws iot delete-v2-logging-level \
    --target-type THING_GROUP \
    --target-name LightBulbs
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteV2LoggingLevel。

deprecate-thing-type

以下代码示例演示了如何使用 deprecate-thing-type。

Amazon CLI

示例 1：弃用事物类型

以下 deprecate-thing-type 示例弃用了某一事物类型，因此用户无法将任何新事物与该类型关联。

```
aws iot deprecate-thing-type \
    --thing-type-name "obsoleteThingType"
```

此命令不生成任何输出。

示例 2：撤消对某一事物类型的弃用

以下 deprecate-thing-type 示例撤消了对某一事物类型的弃用，这样用户就可以再次将新事物与其关联起来。

```
aws iot deprecate-thing-type \
    --thing-type-name "obsoleteThingType" \
    --undo-deprecate
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprecateThingType。

describe-account-audit-configuration

以下代码示例演示了如何使用 describe-account-audit-configuration。

Amazon CLI

查看当前的审计配置设置

以下 `describe-account-audit-configuration` 示例列出您的 Amazon IoT Device Defender 审计配置的当前设置。

```
aws iot describe-account-audit-configuration
```

输出：

```
{
    "roleArn": "arn:aws:iam::123456789012:role/service-role/
AWSIoTDeviceDefenderAudit_1551201085996",
    "auditNotificationTargetConfigurations": {
        "SNS": {
            "targetArn": "arn:aws:sns:us-west-2:123456789012:ddaudits",
            "roleArn": "arn:aws:iam::123456789012:role/service-role/
AWSIoTDeviceDefenderAudit",
            "enabled": true
        }
    },
    "auditCheckConfigurations": {
        "AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK": {
            "enabled": true
        },
        "CA_CERTIFICATE_EXPIRING_CHECK": {
            "enabled": true
        },
        "CONFLICTING_CLIENT_IDS_CHECK": {
            "enabled": true
        },
        "DEVICE_CERTIFICATE_EXPIRING_CHECK": {
            "enabled": true
        },
        "DEVICE_CERTIFICATE_SHARED_CHECK": {
            "enabled": true
        },
        "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": {
            "enabled": true
        },
        "LOGGING_DISABLED_CHECK": {
            "enabled": true
```

```
        },
        "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK": {
            "enabled": true
        },
        "REVOKED_DEVICE_CERTIFICATE_STILL_ACTIVE_CHECK": {
            "enabled": true
        },
        "UNAUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK": {
            "enabled": true
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAuditConfiguration。

`describe-audit-finding`

以下代码示例演示了如何使用 `describe-audit-finding`。

Amazon CLI

列出审计调查发现的详细信息

以下 `describe-audit-finding` 示例列出指定 Amazon IoT Device Defender 审计调查发现的详细信息。一个审计可以产生多个调查发现。使用 `list-audit-findings` 命令得到一个审计的调查发现列表以获取 `findingId`。

```
aws iot describe-audit-finding \
    --finding-id "ef4826b8-e55a-44b9-b460-5c485355371b"
```

输出：

```
{
    "finding": {
        "findingId": "ef4826b8-e55a-44b9-b460-5c485355371b",
        "taskId": "873ed69c74a9ec8fa9b8e88e9abc4661",
        "checkName": "IOT_POLICY_OVERLY_PERMISSIVE_CHECK",
        "taskStartTime": 1576012045.745,
        "findingTime": 1576012046.168,
```

```
        "severity": "CRITICAL",
        "nonCompliantResource": {
            "resourceType": "IOT_POLICY",
            "resourceIdentifier": {
                "policyVersionIdentifier": {
                    "policyName": "smp-ggrass-group_Core-policy",
                    "policyVersionId": "1"
                }
            }
         },
        "reasonForNonCompliance": "Policy allows broad access to IoT data plane
 actions: [iot:Subscribe, iot:Connect, iot:GetThingShadow, iot:DeleteThingShadow,
 iot:UpdateThingShadow, iot:Publish].",
        "reasonForNonComplianceCode":
 "ALLOWS_BROAD_ACCESS_TO_IOT_DATA_PLANE_ACTIONS"
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检查审计结果（审计命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAuditFinding。

describe-audit-mitigation-actions-task

以下代码示例演示了如何使用 describe-audit-mitigation-actions-task。

Amazon CLI

显示审计缓解操作任务的详细信息

以下 describe-audit-mitigation-actions-task 示例显示指定任务的详细信息，其中 ResetPolicyVersionAction 应用于一个调查发现。结果包括任务何时开始和结束、针对多少调查发现（以及成果）、作为该任务一部分应用的操作的定义。

```
aws iot describe-audit-mitigation-actions-task \
    --task-id ResetPolicyTask01
```

输出：

```
{
    "taskStatus": "COMPLETED",
    "startTime": "2019-12-10T15:13:19.457000-08:00",
    "endTime": "2019-12-10T15:13:19.947000-08:00",
```

```
    "taskStatistics": {
        "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": {
            "totalFindingsCount": 1,
            "failedFindingsCount": 0,
            "succeededFindingsCount": 1,
            "skippedFindingsCount": 0,
            "canceledFindingsCount": 0
        }
    },
    "target": {
        "findingIds": [
            "ef4826b8-e55a-44b9-b460-5c485355371b"
        ]
    },
    "auditCheckToActionsMapping": {
        "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": [
            "ResetPolicyVersionAction"
        ]
    },
    "actionsDefinition": [
        {
            "name": "ResetPolicyVersionAction",
            "id": "1ea0b415-bef1-4a01-bd13-72fb63c59afb",
            "roleArn": "arn:aws:iam::123456789012:role/service-role/
ReplacePolicyVersionRole",
            "actionParams": {
                "replaceDefaultPolicyVersionParams": {
                    "templateName": "BLANK_POLICY"
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 DescribeAuditMitigationActionsTask（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAuditMitigationActionsTask。

describe-audit-suppression

以下代码示例演示了如何使用 describe-audit-suppression。

Amazon CLI

获取有关审计调查发现抑制的详细信息

以下 describe-audit-suppression 示例列出有关审计调查发现抑制的详细信息。

```
aws iot describe-audit-task \
    --task-id "787ed873b69cb4d6cdbae6ddd06996c5"
```

输出：

```
{
    "taskStatus": "COMPLETED",
    "taskType": "SCHEDULED_AUDIT_TASK",
    "taskStartTime": 1596168096.157,
    "taskStatistics": {
        "totalChecks": 1,
        "inProgressChecks": 0,
        "waitingForDataCollectionChecks": 0,
        "compliantChecks": 0,
        "nonCompliantChecks": 1,
        "failedChecks": 0,
        "canceledChecks": 0
    },
    "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",
    "auditDetails": {
        "DEVICE_CERTIFICATE_EXPIRING_CHECK": {
            "checkRunStatus": "COMPLETED_NON_COMPLIANT",
            "checkCompliant": false,
            "totalResourcesCount": 195,
            "nonCompliantResourcesCount": 2
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计调查发现抑制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAuditSuppression。

describe-audit-task

以下代码示例演示了如何使用 describe-audit-task。

Amazon CLI

获取有关审计实例的信息

以下 `describe-audit-task` 示例获取有关 Amazon IoT Device Defender 审计实例的信息。如果审计完成，结果中将包括运行的汇总统计信息。

```
aws iot describe-audit-task \
    --task-id a3aea009955e501a31b764abe1bebd3d
```

输出：

```
{
   "taskStatus": "COMPLETED",
   "taskType": "ON_DEMAND_AUDIT_TASK",
   "taskStartTime": 1560356923.434,
   "taskStatistics": {
       "totalChecks": 3,
       "inProgressChecks": 0,
       "waitingForDataCollectionChecks": 0,
       "compliantChecks": 3,
       "nonCompliantChecks": 0,
       "failedChecks": 0,
       "canceledChecks": 0
   },
   "auditDetails": {
       "CA_CERTIFICATE_EXPIRING_CHECK": {
           "checkRunStatus": "COMPLETED_COMPLIANT",
           "checkCompliant": true,
           "totalResourcesCount": 0,
           "nonCompliantResourcesCount": 0
       },
       "DEVICE_CERTIFICATE_EXPIRING_CHECK": {
           "checkRunStatus": "COMPLETED_COMPLIANT",
           "checkCompliant": true,
           "totalResourcesCount": 6,
           "nonCompliantResourcesCount": 0
       },
       "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK": {
           "checkRunStatus": "COMPLETED_COMPLIANT",
           "checkCompliant": true,
           "totalResourcesCount": 0,
           "nonCompliantResourcesCount": 0
```

```
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAuditTask。

describe-authorizer

以下代码示例演示了如何使用 describe-authorizer。

Amazon CLI

获取有关自定义授权方的信息

以下 describe-authorizer 示例显示指定自定义授权方的详细信息。

```
aws iot describe-authorizer \
    --authorizer-name CustomAuthorizer
```

输出：

```
{
    "authorizerDescription": {
        "authorizerName": "CustomAuthorizer",
        "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer",
        "authorizerFunctionArn": "arn:aws:lambda:us-
west-2:123456789012:function:CustomAuthorizerFunction",
        "tokenKeyName": "MyAuthToken",
        "tokenSigningPublicKeys": {
            "FIRST_KEY": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1uJOB4lQPgG/lM6ZfIwo
\nZ+7ENxAio9q6QD4FFqjGZsvjtYwjoe1RKK0U8Eq9xb5O3kRSmyIwTzwzm/f4Gf0Y
\nZUloJ+t3PUUwHrmbYTAgTrCUgRFygjfgVwGCPs5ZAX4Eyqt5cr+AIHIiUDbxSa7p
\nzwOBKPeic0asNJpqT8PkBbRaKyleJh5oo81NDHHmVtbBm5A5YiJjqYXLaVAowKzZ\n
+GqsNvAQ9Jy1wI2VrEa1OfL8flDB/BJLm7zjpfPOHDJQgID0XnZwAlNnZcOhCwIx\n50g2LW2Oy9R/
dmqtDmJiVP97Z4GykxPvwlYHrUXY0iW1R3AR/Ac1NhCTGZMwVDB1\nlQIDAQAB\n-----END PUBLIC
 KEY-----"
        },
        "status": "ACTIVE",
```

```
        "creationDate": 1571245658.069,
        "lastModifiedDate": 1571245658.069
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAuthorizer。

describe-billing-group

以下代码示例演示了如何使用 describe-billing-group。

Amazon CLI

获取有关账单组的信息

以下 describe-billing-group 示例获取指定账单组的信息。

```
aws iot describe-billing-group --billing-group-name GroupOne
```

输出：

```
{
    "billingGroupName": "GroupOne",
    "billingGroupId": "103de383-114b-4f51-8266-18f209ef5562",
    "billingGroupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne",
    "version": 1,
    "billingGroupProperties": {},
    "billingGroupMetadata": {
        "creationDate": 1560199355.378
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBillingGroup。

describe-ca-certificate

以下代码示例演示了如何使用 describe-ca-certificate。

Amazon CLI

获取有关 CA 证书的详细信息

以下 describe-ca-certificate 示例显示指定 CA 证书的详细信息。

```
aws iot describe-ca-certificate \
    --certificate-
id f4efed62c0142f16af278166f61962501165c4f05362952074264600058cd1467
```

输出：

```
{
    "certificateDescription": {
        "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/
f4efed62c0142f16af278166f61962501165c4f05362952074264600058cd1467",
        "certificateId":
 "f4efed62c0142f16af278166f61962501165c4f05362952074264600058cd1467",
        "status": "INACTIVE",
        "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIICzzCCAbegEXAMPLEJANVEPWXl8taPMA0GCSqGSIb3DQEBBQUAMB4xCzAJBgNV
\nBAYTAlVTMQ8wDQYDVQQKDAZBbWF6b24wHhcNMTkwOTI0MjEzMTE1WhcNMjkwOTIx
\nMjEzMTE1WjAeMQswCQYDVQQGEwJVUzEPMA0GA1UECgwGQW1hem9uMIIBIjANBgkq
\nhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzd3R3ioalCS0MhFWfBrVGR036EK07UAf
\nVdz9EXAMPLE1VczICbADnATK522kEIB51/18VzlFtAhQL5V5eybXKnB7QebNer5m
\n4Yibx7shR5oqNzFsrXWxuugN5+w5gEfqNMawOjhF4LsculKG49yuqjcDU19/13ua
\n3B2gxs1Pe7TiWWvUskzxnbO1F2WCshbEJvqY8fIWtGYCjTeJAgQ9hvZx/69XhKen
\nwV9LJwOQxrsUS0Ty8IHwbB8fRy72VM3u7fJoaU+nO4jD5cqaoEPtzoeFUEXAMPLE
\nyVAJpqHwgbYbcUfn7V+AB6yh1+0Fa1rEQGuZDPGyJslxwr5vh8nRewIDAQABoxAw
\nDjAMBgNVHRMEBTADAQH/MA0GCSqGSIb3DQEBBQUAA4IBAQA+3a5CV3IJgOnd0AgI
\nBgVMtmYzTvqAngx26aG9/spvCjXckh2SBF+EcBlCFwH1yakwjJL1dR4yarnrfxgI
\nEqP4AOYVimAVoQ5FBwnloHe16+3qtDiblU9DeXBUCtS55EcfrEXAMPLEYtXdqU5C
\nU9ia4KAjV0dxW1+EFYMwX5eGeb0gDTNHBylV6B/fOSZiQAwDYp4x3B+gAP+a/bWB
\nu1umOqtBdWe6L6/83L+JhaTByqV25iVJ4c/UZUnG8926wUlDM9zQvEXuEVvzZ7+m\n4PSNqst/
nVOvnLpoG4e0WgcJgANuB33CSWtjWSuYsbhmqQRknGhREXAMPLEZT4fm\nfo0e\n-----END
 CERTIFICATE-----\n",
        "ownedBy": "123456789012",
        "creationDate": 1569365372.053,
        "autoRegistrationStatus": "DISABLE",
        "lastModifiedDate": 1569365372.053,
        "customerVersion": 1,
        "generationId": "c5c2eb95-140b-4f49-9393-6aaac85b2a90",
        "validity": {
```

```
            "notBefore": 1569360675.0,
            "notAfter": 1884720675.0
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeCACertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCaCertificate。

describe-certificate

以下代码示例演示了如何使用 describe-certificate。

Amazon CLI

获取有关证书的信息

以下 describe-certificate 示例显示指定证书的详细信息。

```
aws iot describe-certificate \
    --certificate-
id "4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e"
```

输出：

```
{
    "certificateDescription": {
        "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
        "certificateId":
 "4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
        "status": "ACTIVE",
        "certificatePem": "-----BEGIN CERTIFICATE-----
MIICiTEXAMPLEQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBEXAMPLEMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDEXAMPLElMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5EXAMPLEcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNEXAMPLEdBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBEXAMPLEz
b2xEXAMPLEYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8EXAMPLEZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
```

```
21uUSfwfEvySWtC2XADZ4nB+BLYEXAMPLEpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7EXAMPLEGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFEXAMPLEAtCu4
nUhVVxYUnEXAMPLE8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GEXAMPLEl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
        "ownedBy": "123456789012",
        "creationDate": 1541022751.983,
        "lastModifiedDate": 1541022751.983,
        "customerVersion": 1,
        "transferData": {},
        "generationId": "6974fbed-2e61-4114-bc5e-4204cc79b045",
        "validity": {
            "notBefore": 1541022631.0,
            "notAfter": 2524607999.0
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeCertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificate。

describe-custom-metric

以下代码示例演示了如何使用 describe-custom-metric。

Amazon CLI

获取有关 Device Defender 自定义指标的信息

以下 describe-custom-metric 示例获取有关名为 myCustomMetric 的自定义指标的信息。

```
aws iot describe-custom-metric \
    --metric-name myCustomMetric
```

输出：

```
{
    "metricName": "myCustomMetric",
    "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/myCustomMetric",
```

```
    "metricType": "number",
    "displayName": "My custom metric",
    "creationDate": 2020-11-17T23:02:12.879000-09:00,
    "lastModifiedDate": 2020-11-17T23:02:12.879000-09:00
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的自定义指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomMetric。

describe-default-authorizer

以下代码示例演示了如何使用 describe-default-authorizer。

Amazon CLI

获取有关默认自定义授权方的信息

以下 describe-default-authorizer 示例显示默认自定义授权方的详细信息。

```
aws iot describe-default-authorizer
```

输出：

```
{
    "authorizerName": "CustomAuthorizer",
    "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeDefaultAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDefaultAuthorizer。

describe-dimension

以下代码示例演示了如何使用 describe-dimension。

Amazon CLI

获取有关维度的信息

以下 describe-dimension 示例获取有关名为 TopicFilterForAuthMessages 的维度的信息。

```
aws iot describe-dimension \
    --name TopicFilterForAuthMessages
```

输出：

```
{
    "name": "TopicFilterForAuthMessages",
    "arn": "arn:aws:iot:eu-west-2:123456789012:dimension/
TopicFilterForAuthMessages",
    "type": "TOPIC_FILTER",
    "stringValues": [
        "device/+/auth"
    ],
    "creationDate": 1578620223.255,
    "lastModifiedDate": 1578620223.255
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDimension。

describe-domain-configuration

以下代码示例演示了如何使用 describe-domain-configuration。

Amazon CLI

描述域配置

以下 describe-domain-configuration 示例显示有关指定域配置的详细信息。

```
aws iot describe-domain-configuration \
    --domain-configuration-name "additionalDataDomain"
```

输出：

```
{
```

```
    "domainConfigurationName": "additionalDataDomain",
    "domainConfigurationArn": "arn:aws:iot:us-
east-1:758EXAMPLE143:domainconfiguration/additionalDataDomain/norpw",
    "domainName": "d055exampleed74y71zfd-ats.beta.us-east-1.iot.amazonaws.com",
    "serverCertificates": [],
    "domainConfigurationStatus": "ENABLED",
    "serviceType": "DATA",
    "domainType": "AWS_MANAGED",
    "lastStatusChangeDate": 1601923783.774
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的可配置的端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDomainConfiguration。

describe-endpoint

以下代码示例演示了如何使用 describe-endpoint。

Amazon CLI

示例 1：获取您当前的 Amazon 端点

以下 describe-endpoint 示例检索应用所有命令的默认 Amazon 端点。

```
aws iot describe-endpoint
```

输出：

```
{
    "endpointAddress": "abc123defghijk.iot.us-west-2.amazonaws.com"
}
```

有关更多信息，请参阅《Amazon物联网开发人员指南》中的 DescribeEndpoint。

示例 2：获取您的 ATS 端点

以下 describe-endpoint 示例检索 Amazon Trust Services（ATS）端点。

```
aws iot describe-endpoint \
```

```
    --endpoint-type iot:Data-ATS
```

输出：

```
{
    "endpointAddress": "abc123defghijk-ats.iot.us-west-2.amazonaws.com"
}
```

有关更多信息，请参阅《Amazon物联网开发人员指南》中的 X.509 证书和 Amazon IoT。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpoint。

describe-event-configurations

以下代码示例演示了如何使用 describe-event-configurations。

Amazon CLI

显示已发布的事件类型

以下 describe-event-configurations 示例列出了控制添加、更新或删除内容时生成哪些事件的配置。

```
aws iot describe-event-configurations
```

输出：

```
{
    "eventConfigurations": {
        "CA_CERTIFICATE": {
            "Enabled": false
        },
        "CERTIFICATE": {
            "Enabled": false
        },
        "JOB": {
            "Enabled": false
        },
        "JOB_EXECUTION": {
            "Enabled": false
        },
```

```
        "POLICY": {
            "Enabled": false
        },
        "THING": {
            "Enabled": false
        },
        "THING_GROUP": {
            "Enabled": false
        },
        "THING_GROUP_HIERARCHY": {
            "Enabled": false
        },
        "THING_GROUP_MEMBERSHIP": {
            "Enabled": false
        },
        "THING_TYPE": {
            "Enabled": false
        },
        "THING_TYPE_ASSOCIATION": {
            "Enabled": false
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事件消息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventConfigurations。

describe-index

以下代码示例演示了如何使用 describe-index。

Amazon CLI

检索事物索引的当前状态

以下 describe-index 示例检索事物索引的当前状态。

```
aws iot describe-index \
    --index-name "AWS_Things"
```

输出：

```
{
    "indexName": "AWS_Things",
    "indexStatus": "ACTIVE",
    "schema": "REGISTRY_AND_SHADOW_AND_CONNECTIVITY_STATUS"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的管理事物索引。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIndex。

describe-job-execution

以下代码示例演示了如何使用 describe-job-execution。

Amazon CLI

获取设备上作业的执行详细信息

以下 describe-job-execution 示例获取指定作业的执行详细信息。

```
aws iot describe-job-execution \
    --job-id "example-job-01" \
    --thing-name "MyRaspberryPi"
```

输出：

```
{
    "execution": {
        "jobId": "example-job-01",
        "status": "QUEUED",
        "statusDetails": {},
        "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi",
        "queuedAt": 1560787023.636,
        "lastUpdatedAt": 1560787023.636,
        "executionNumber": 1,
        "versionNumber": 1
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJobExecution。

describe-job

以下代码示例演示了如何使用 describe-job。

Amazon CLI

获取作业的详细状态

以下 describe-job 示例获取 ID 为 example-job-01 的作业的详细状态。

```
aws iot describe-job \
    --job-id "example-job-01"
```

输出：

```
{
    "job": {
        "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
        "jobId": "example-job-01",
        "targetSelection": "SNAPSHOT",
        "status": "IN_PROGRESS",
        "targets": [
            "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi"
        ],
        "description": "example job test",
        "presignedUrlConfig": {},
        "jobExecutionsRolloutConfig": {},
        "createdAt": 1560787022.733,
        "lastUpdatedAt": 1560787026.294,
        "jobProcessDetails": {
            "numberOfCanceledThings": 0,
            "numberOfSucceededThings": 0,
            "numberOfFailedThings": 0,
            "numberOfRejectedThings": 0,
            "numberOfQueuedThings": 1,
            "numberOfInProgressThings": 0,
            "numberOfRemovedThings": 0,
            "numberOfTimedOutThings": 0
        },
        "timeoutConfig": {}
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJob。

describe-mitigation-action

以下代码示例演示了如何使用 describe-mitigation-action。

Amazon CLI

查看已定义的缓解操作的详细信息

以下 describe-mitigation-action 示例显示指定缓解操作的详细信息。

```
aws iot describe-mitigation-action \
    --action-name AddThingsToQuarantineGroupAction
```

输出：

```
{
    "actionName": "AddThingsToQuarantineGroupAction",
    "actionType": "ADD_THINGS_TO_THING_GROUP",
    "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
    "actionId": "2fd2726d-98e1-4abf-b10f-09465ccd6bfa",
    "roleArn": "arn:aws:iam::123456789012:role/service-role/
MoveThingsToQuarantineGroupRole",
    "actionParams": {
        "addThingsToThingGroupParams": {
            "thingGroupNames": [
                "QuarantineGroup1"
            ],
            "overrideDynamicGroups": true
        }
    },
    "creationDate": "2019-12-10T11:09:35.999000-08:00",
    "lastModifiedDate": "2019-12-10T11:09:35.999000-08:00"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 DescribeMitigationAction（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMitigationAction。

describe-provisioning-template-version

以下代码示例演示了如何使用 describe-provisioning-template-version。

Amazon CLI

描述预配模板版本

以下 describe-provisioning-template-version 示例描述一个预配模板版本。

```
aws iot describe-provisioning-template-version \
    --template-name MyTestProvisioningTemplate \
    --version-id 1
```

输出：

```
{
    "versionId": 1,
    "creationDate": 1589308310.574,
    "templateBody": "{
        \"Parameters\":{
            \"SerialNumber\":{
                \"Type\":\"String\"
            },
            \"AWS::IoT::Certificate::Id\":{
                \"Type\":\"String\"
            }
        },
        \"Resources\":{
            \"certificate\":{
                \"Properties\":{
                    \"CertificateId\":{
                        \"Ref\":\"AWS::IoT::Certificate::Id\"
                    },
                    \"Status\":\"Active\"
                },
                \"Type\":\"AWS::IoT::Certificate\"
            },
            \"policy\":{
                \"Properties\":{
                    \"PolicyName\":\"MyIotPolicy\"
                },
                \"Type\":\"AWS::IoT::Policy\"
```

```
            },
            \"thing\":{
                \"OverrideSettings\":{
                    \"AttributePayload\":\"MERGE\",
                    \"ThingGroups\":\"DO_NOTHING\",
                    \"ThingTypeName\":\"REPLACE\"
                },
                \"Properties\":{
                    \"AttributePayload\":{},
                    \"ThingGroups\":[],
                    \"ThingName\":{
                        \"Fn::Join\":[
                            \"\",
                            [
                                \"DemoGroup_\",
                                {\"Ref\":\"SerialNumber\"}
                            ]
                        ]
                    },
                    \"ThingTypeName\":\"VirtualThings\"
                },
                \"Type\":\"AWS::IoT::Thing\"
            }
        }
    }",
    "isDefaultVersion": true
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的使用实例集预配来预配没有设备证书的设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProvisioningTemplateVersion。

`describe-provisioning-template`

以下代码示例演示了如何使用 `describe-provisioning-template`。

Amazon CLI

描述预配模板

以下 `describe-provisioning-template` 示例描述一个预配模板。

```
aws iot describe-provisioning-template \
    --template-name MyTestProvisioningTemplate
```

输出：

```
{
    "templateArn": "arn:aws:iot:us-west-2:57EXAMPLE833:provisioningtemplate/
MyTestProvisioningTemplate",
    "templateName": "MyTestProvisioningTemplate",
    "creationDate": 1589308310.574,
    "lastModifiedDate": 1589308345.539,
    "defaultVersionId": 1,
    "templateBody": "{
        \"Parameters\":{
            \"SerialNumber\":{
                \"Type\":\"String\"
            },
            \"AWS::IoT::Certificate::Id\":{
                \"Type\":\"String\"
            }
        },
        \"Resources\":{
            \"certificate\":{
                \"Properties\":{
                    \"CertificateId\":{
                        \"Ref\":\"AWS::IoT::Certificate::Id\"
                    },
                    \"Status\":\"Active\"
                },
                \"Type\":\"AWS::IoT::Certificate\"
            },
            \"policy\":{
                \"Properties\":{
                    \"PolicyName\":\"MyIotPolicy\"
                },
                \"Type\":\"AWS::IoT::Policy\"
            },
            \"thing\":{
                \"OverrideSettings\":{
                    \"AttributePayload\":\"MERGE\",
                    \"ThingGroups\":\"DO_NOTHING\",
                    \"ThingTypeName\":\"REPLACE\"
                },
```

```
                \"Properties\":{
                    \"AttributePayload\":{},
                    \"ThingGroups\":[],
                    \"ThingName\":{
                        \"Fn::Join\":[
                            \"\",
                            [
                                \"DemoGroup_\",
                                {\"Ref\":\"SerialNumber\"}
                            ]
                        ]
                    },
                    \"ThingTypeName\":\"VirtualThings\"
                },
                \"Type\":\"AWS::IoT::Thing\"
            }
        }
    }",
    "enabled": true,
    "provisioningRoleArn": "arn:aws:iam::571032923833:role/service-role/IoT_access"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的使用实例集预配来预配没有设备证书的设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProvisioningTemplate。

describe-role-alias

以下代码示例演示了如何使用 describe-role-alias。

Amazon CLI

获取有关 Amazon IoT 角色别名的信息

以下 describe-role-alias 示例显示指定角色别名的详细信息。

```
aws iot describe-role-alias \
    --role-alias LightBulbRole
```

输出：

```
{
```

```
    "roleAliasDescription": {
        "roleAlias": "LightBulbRole",
        "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/
LightBulbRole",
        "roleArn": "arn:aws:iam::123456789012:role/light_bulb_role_001",
        "owner": "123456789012",
        "credentialDurationSeconds": 3600,
        "creationDate": 1570558643.221,
        "lastModifiedDate": 1570558643.221
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeRoleAlias。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeRoleAlias。

describe-scheduled-audit

以下代码示例演示了如何使用 describe-scheduled-audit。

Amazon CLI

获取有关预定审计的信息

以下 describe-scheduled-audit 示例获取有关名为 AWSIoTDeviceDefenderDailyAudit 的 Amazon IOT Device Defender 的预定审计的详细信息。

```
aws iot describe-scheduled-audit \
    --scheduled-audit-name AWSIoTDeviceDefenderDailyAudit
```

输出：

```
{
    "frequency": "DAILY",
    "targetCheckNames": [
        "AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK",
        "CONFLICTING_CLIENT_IDS_CHECK",
        "DEVICE_CERTIFICATE_SHARED_CHECK",
        "IOT_POLICY_OVERLY_PERMISSIVE_CHECK",
        "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK",
        "UNAUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK"
    ],
```

```
    "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",
    "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSIoTDeviceDefenderDailyAudit"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledAudit。

describe-security-profile

以下代码示例演示了如何使用 describe-security-profile。

Amazon CLI

获取有关安全配置文件的信息

以下 describe-security-profile 示例获取有关名为 PossibleIssue. 的 Amazon IoT Device Defender 安全配置文件的信息。

```
aws iot describe-security-profile \
    --security-profile-name PossibleIssue
```

输出：

```
{
    "securityProfileName": "PossibleIssue",
    "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
PossibleIssue",
    "securityProfileDescription": "check to see if authorization fails 10 times in 5
 minutes or if cellular bandwidth exceeds 128",
    "behaviors": [
        {
            "name": "CellularBandwidth",
            "metric": "aws:message-byte-size",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 128
                },
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
```

```
            }
        },
        {
            "name": "Authorization",
            "metric": "aws:num-authorization-failures",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 10
                },
                "durationSeconds": 300,
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        }
    ],
    "version": 1,
    "creationDate": 1560278102.528,
    "lastModifiedDate": 1560278102.528
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecurityProfile。

describe-stream

以下代码示例演示了如何使用 describe-stream。

Amazon CLI

获取有关直播的信息

以下 describe-stream 示例显示有关指定流的详细信息。

```
aws iot describe-stream \
    --stream-id stream12345
```

输出：

```
{
    "streamInfo": {
```

```
        "streamId": "stream12345",
        "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
        "streamVersion": 1,
        "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
        "files": [
            {
                "fileId": "123",
                "s3Location": {
                    "bucket":"codesign-ota-bucket",
                    "key":"48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
                }
            }
        ],
        "createdAt": 1557863215.995,
        "lastUpdatedAt": 1557863215.995,
        "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_stream_role"
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 DescribeStream。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStream。

describe-thing-group

以下代码示例演示了如何使用 describe-thing-group。

Amazon CLI

获取有关事物组的信息

以下 describe-thing-group 示例获取有关名为 HalogenBulbs 的事物组的信息。

```
aws iot describe-thing-group \
    --thing-group-name HalogenBulbs
```

输出：

```
{
    "thingGroupName": "HalogenBulbs",
    "thingGroupId": "f4ec6b84-b42b-499d-9ce1-4dbd4d4f6f6e",
    "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs",
```

```
    "version": 1,
    "thingGroupProperties": {},
    "thingGroupMetadata": {
        "parentGroupName": "LightBulbs",
        "rootToParentThingGroups": [
            {
                "groupName": "LightBulbs",
                "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
LightBulbs"
            }
        ],
        "creationDate": 1559927609.897
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeThingGroup。

describe-thing-type

以下代码示例演示了如何使用 describe-thing-type。

Amazon CLI

获取有关某一事物类型的信息

以下 describe-thing-type 示例显示有关您的 Amazon 账户中定义的指定事物类型的信息。

```
aws iot describe-thing-type \
    --thing-type-name "LightBulb"
```

输出：

```
{
    "thingTypeName": "LightBulb",
    "thingTypeId": "ce3573b0-0a3c-45a7-ac93-4e0ce14cd190",
    "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/LightBulb",
    "thingTypeProperties": {
        "thingTypeDescription": "light bulb type",
        "searchableAttributes": [
            "model",
            "wattage"
```

```
        ]
    },
    "thingTypeMetadata": {
        "deprecated": false,
        "creationDate": 1559772562.498
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeThingType。

`describe-thing`

以下代码示例演示了如何使用 `describe-thing`。

Amazon CLI

显示有关事物的详细信息

以下 `describe-thing` 示例显示有关您的 Amazon 账户的 Amazon物联网注册表中定义的事物（设备）的信息。

aws iot describe-thing --thing-name "MyLightBulb"

输出：

```
{
    "defaultClientId": "MyLightBulb",
    "thingName": "MyLightBulb",
    "thingId": "40da2e73-c6af-406e-b415-15acae538797",
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
    "thingTypeName": "LightBulb",
    "attributes": {
        "model": "123",
        "wattage": "75"
    },
    "version": 1
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的如何使用注册表管理事物。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeThing。

detach-policy

以下代码示例演示了如何使用 detach-policy。

Amazon CLI

示例 1：将 Amazon IoT 策略与事物组分离

以下 detach-policy 示例将指定策略与一个事物组分离，进而与该事物组中的所有事物以及该组的任何子组分离。

```
aws iot detach-policy \
    --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \
    --policy-name "MyFirstGroup_Core-policy"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

示例 2：将 Amazon IoT 策略与设备证书分离

以下 detach-policy 示例将 TemperatureSensorPolicy 策略与 ARN 标识的一个设备证书分离。

```
aws iot detach-policy \
    --policy-name TemperatureSensorPolicy \
    --target arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachPolicy。

detach-security-profile

以下代码示例演示了如何使用 detach-security-profile。

Amazon CLI

取消安全配置文件与目标的关联

以下 detach-security-profile 示例删除名为 Testprofile 的Amazon IoT Device Defender 安全配置文件与所有注册事物目标之间的关联。

```
aws iot detach-security-profile \
    --security-profile-name Testprofile \
    --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/
registered-things"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachSecurityProfile。

detach-thing-principal

以下代码示例演示了如何使用 detach-thing-principal。

Amazon CLI

将证书/主体与事物分离

以下 detach-thing-principal 示例从指定事物中移除代表一个主体的证书。

```
aws iot detach-thing-principal \
    --thing-name "MyLightBulb" \
    --principal "arn:aws:iot:us-
west-2:123456789012:cert/604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的如何使用注册表管理事物。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachThingPrincipal。

disable-topic-rule

以下代码示例演示了如何使用 disable-topic-rule。

Amazon CLI

禁用主题规则

以下 disable-topic-rule 示例禁用指定主题规则。

```
aws iot disable-topic-rule \
```

```
    --rule-name "MyPlantPiMoistureAlertRule"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的查看您的规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableTopicRule。

enable-topic-rule

以下代码示例演示了如何使用 enable-topic-rule。

Amazon CLI

启用主题规则

以下 enable-topic-rule 示例启用（或重新启用）指定的主题规则。

```
aws iot enable-topic-rule \
    --rule-name "MyPlantPiMoistureAlertRule"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的查看您的规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableTopicRule。

get-behavior-model-training-summaries

以下代码示例演示了如何使用 get-behavior-model-training-summaries。

Amazon CLI

列出 Device Defender 的 ML Detect 安全配置文件训练模型的状态

以下 get-behavior-model-training-summaries 示例列出所选安全配置文件中已配置行为的模型训练状态。对于每种行为，都会列出名称、模型状态和收集的数据点百分比。

```
aws iot get-behavior-model-training-summaries \
    --security-profile-name MySecuirtyProfileName
```

输出：

```
{
    "summaries": [
        {
            "securityProfileName": "MySecuirtyProfileName",
            "behaviorName": "Messages_sent_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        },
        {
            "securityProfileName": "MySecuirtyProfileName",
            "behaviorName": "Messages_received_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        },
        {
            "securityProfileName": "MySecuirtyProfileName",
            "behaviorName": "Authorization_failures_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        },
        {
            "securityProfileName": "MySecuirtyProfileName",
            "behaviorName": "Message_size_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        },
        {
            "securityProfileName": "MySecuirtyProfileName",
            "behaviorName": "Connection_attempts_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        },
        {
            "securityProfileName": "MySPNoALerts",
            "behaviorName": "Disconnects_ML_behavior",
            "modelStatus": "PENDING_BUILD",
            "datapointsCollectionPercentage": 0.0
        }
    ]
}
```


有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 GetBehaviorModelTrainingSummaries（检测命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBehaviorModelTrainingSummaries。

get-cardinality

以下代码示例演示了如何使用 get-cardinality。

Amazon CLI

返回与查询匹配的唯一值的近似计数

您可以使用以下设置脚本来创建 10 个事物，代表 10 个温度传感器。每个新事物有 3 个属性。

```
# Bash script. If in other shells, type `bash` before running
Temperatures=(70 71 72 73 74 75 47 97 98 99)
Racks=(Rack1 Rack1 Rack2 Rack2 Rack3 Rack4 Rack5 Rack6 Rack6 Rack6)
IsNormal=(true true true true true true false false false false)
for ((i=0; i<10 ; i++))
do
  thing=$(aws iot create-thing --thing-name "TempSensor$i" --attribute-payload
 attributes="{temperature=${Temperatures[i]},rackId=${Racks[i]},stateNormal=
${IsNormal[i]}}")
  aws iot describe-thing --thing-name "TempSensor$i"
done
```

设置脚本的输出示例：

```
{
    "version": 1,
    "thingName": "TempSensor0",
    "defaultClientId": "TempSensor0",
    "attributes": {
        "rackId": "Rack1",
        "stateNormal": "true",
        "temperature": "70"
    },
    "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/TempSensor0",
    "thingId": "example1-90ab-cdef-fedc-ba987example"
}
```

以下 get-cardinality 示例查询由设置脚本创建的 10 个传感器，并返回具有报告异常温度值的温度传感器的机架数量。如果温度值低于 60 或高于 80，则温度传感器处于异常状态。

```
aws iot get-cardinality \
    --aggregation-field "attributes.rackId" \
    --query-string "thingName:TempSensor* AND attributes.stateNormal:false"
```

输出：

```
{
    "cardinality": 2
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的“查询聚合数据”<https://docs.aws.amazon.com/iot/latest/developerguide/index-aggregate.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCardinality。

get-effective-policies

以下代码示例演示了如何使用 get-effective-policies。

Amazon CLI

列出影响事物的策略

以下 get-effective-policies 示例列出影响指定事物的策略，包括附加到其所属任何事物组的策略。

```
aws iot get-effective-policies \
    --thing-name TemperatureSensor-001 \
    --principal arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

输出：

```
{
    "effectivePolicies": [
        {
            "policyName": "TemperatureSensorPolicy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TemperatureSensorPolicy",
            "policyDocument": "{
```

```
                \"Version\": \"2012-10-17\",
                \"Statement\": [
                    {
                        \"Effect\": \"Allow\",
                        \"Action\": [
                            \"iot:Publish\",
                            \"iot:Receive\"
                        ],
                        \"Resource\": [
                            \"arn:aws:iot:us-west-2:123456789012:topic/topic_1\",
                            \"arn:aws:iot:us-west-2:123456789012:topic/topic_2\"
                        ]
                    },
                    {
                        \"Effect\": \"Allow\",
                        \"Action\": [
                            \"iot:Subscribe\"
                        ],
                        \"Resource\": [
                            \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_1\",
                            \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_2\"
                        ]
                    },
                    {
                        \"Effect\": \"Allow\",
                        \"Action\": [
                            \"iot:Connect\"
                        ],
                        \"Resource\": [
                            \"arn:aws:iot:us-west-2:123456789012:client/basicPubSub
\"
                        ]
                    }
                ]
            }"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的获取事物的有效策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEffectivePolicies。

get-indexing-configuration

以下代码示例演示了如何使用 get-indexing-configuration。

Amazon CLI

获取事物索引配置

以下 get-indexing-configuration 示例获取 Amazon IoT 实例集索引的当前配置数据。

```
aws iot get-indexing-configuration
```

输出：

```
{
    "thingIndexingConfiguration": {
        "thingIndexingMode": "OFF",
        "thingConnectivityIndexingMode": "OFF"
    },
    "thingGroupIndexingConfiguration": {
        "thingGroupIndexingMode": "OFF"
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的管理事物索引。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIndexingConfiguration。

get-job-document

以下代码示例演示了如何使用 get-job-document。

Amazon CLI

检索作业的文档

以下 get-job-document 示例显示 ID 为 example-job-01 的作业的文档的详细信息。

```
aws iot get-job-document \
    --job-id "example-job-01"
```

输出：

```
{
    "document": "\n{\n    \"operation\":\"customJob\",\n    \"otherInfo\":
\"someValue\"\n}\n"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobDocument。

get-logging-options

以下代码示例演示了如何使用 get-logging-options。

Amazon CLI

获取日志选项

以下 get-logging-options 示例获取您的 Amazon 账户的当前日志选项。

```
aws iot get-logging-options
```

输出：

```
{
    "roleArn": "arn:aws:iam::123456789012:role/service-role/iotLoggingRole",
    "logLevel": "ERROR"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的相关标题。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoggingOptions。

get-ota-update

以下代码示例演示了如何使用 get-ota-update。

Amazon CLI

检索有关 OTA 更新的信息

以下 get-ota-update 示例显示有关指定 OTA 更新的详细信息。

```
aws iot get-ota-update \
    --ota-update-id ota12345
```

输出：

```
{
    "otaUpdateInfo": {
        "otaUpdateId": "ota12345",
        "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/itsaupdate",
        "creationDate": 1557863215.995,
        "lastModifiedDate": 1557863215.995,
        "description": "A critical update needed right away.",
        "targets": [
           "device1",
           "device2",
           "device3",
           "device4"
        ],
        "targetSelection": "SNAPSHOT",
        "protocols": ["HTTP"],
        "awsJobExecutionsRolloutConfig": {
           "maximumPerMinute": 10
        },
        "otaUpdateFiles": [
            {
                "fileName": "firmware.bin",
                "fileLocation": {
                    "stream": {
                        "streamId": "004",
                        "fileId":123
                    }
                },
                "codeSigning": {
                    "awsSignerJobId": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
                }
            }
        ],
        "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_role"
        "otaUpdateStatus": "CREATE_COMPLETE",
        "awsIotJobId": "job54321",
        "awsIotJobArn": "arn:aws:iot:us-west-2:123456789012:job/job54321",
        "errorInfo": {
        }
```

```
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 GetOTAUpdate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOtaUpdate。

get-percentiles

以下代码示例演示了如何使用 get-percentiles。

Amazon CLI

将与查询匹配的聚合值分组为百分位数分组

您可以使用以下设置脚本来创建 10 个事物，代表 10 个温度传感器。每个新事物都有 1 个属性。

```
# Bash script. If in other shells, type `bash` before running
Temperatures=(70 71 72 73 74 75 47 97 98 99)
for ((i=0; i<10 ; i++))
do
    thing=$(aws iot create-thing --thing-name "TempSensor$i" --attribute-payload
 attributes="{temperature=${Temperatures[i]}}")
    aws iot describe-thing --thing-name "TempSensor$i"
done
```

设置脚本的输出示例：

```
{
    "version": 1,
    "thingName": "TempSensor0",
    "defaultClientId": "TempSensor0",
    "attributes": {
        "temperature": "70"
    },
    "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/TempSensor0",
    "thingId": "example1-90ab-cdef-fedc-ba987example"
}
```

以下 get-percentiles 示例查询由设置脚本创建的 10 个传感器，并为指定的每个百分位数组返回一个值。百分位数组“10”包含的聚合字段值发生在大约 10% 的与查询匹配的值中。在以下输出中，{"percent": 10.0, "value": 67.7} 表示大约 10.0% 的温度值低于 67.7。

```
aws iot get-percentiles \
    --aggregation-field "attributes.temperature" \
    --query-string "thingName:TempSensor*" \
    --percents 10 25 50 75 90
```

输出：

```
{
    "percentiles": [
        {
            "percent": 10.0,
            "value": 67.7
        },
        {
            "percent": 25.0,
            "value": 71.25
        },
        {
            "percent": 50.0,
            "value": 73.5
        },
        {
            "percent": 75.0,
            "value": 91.5
        },
        {
            "percent": 90.0,
            "value": 98.1
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的查询聚合数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPercentiles。

get-policy-version

以下代码示例演示了如何使用 get-policy-version。

Amazon CLI

获取有关策略特定版本的信息

以下 get-policy-version 示例获取指定策略的第一个版本的信息。

```
aws iot get-policy \
    --policy-name UpdateDeviceCertPolicy
    --policy-version-id "1"
```

输出：

```
{
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
    "policyName": "UpdateDeviceCertPolicy",
    "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\":
 \"Allow\", \"Action\":  \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
    "policyVersionId": "1",
    "isDefaultVersion": false,
    "creationDate": 1559925941.924,
    "lastModifiedDate": 1559926175.458,
    "generationId":
 "5066f1b6712ce9d2a1e56399771649a272d6a921762fead080e24fe52f24e042"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicyVersion。

get-policy

以下代码示例演示了如何使用 get-policy。

Amazon CLI

获取有关策略默认版本的信息

以下 get-policy 示例检索指定策略的默认版本的信息。

```
aws iot get-policy \
    --policy-name UpdateDeviceCertPolicy
```

输出：

```
{
```

```
    "policyName": "UpdateDeviceCertPolicy",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
    "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\":
 \"Allow\", \"Action\":  \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
    "defaultVersionId": "2",
    "creationDate": 1559925941.924,
    "lastModifiedDate": 1559925941.924,
    "generationId":
 "5066f1b6712ce9d2a1e56399771649a272d6a921762fead080e24fe52f24e042"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicy。

get-registration-code

以下代码示例演示了如何使用 get-registration-code。

Amazon CLI

获取您的特定 Amazon 账户的注册码

以下 get-registration-code 示例检索您的特定 Amazon 账户的注册码。

```
aws iot get-registration-code
```

输出：

```
{
    "registrationCode":
 "15c51ae5e36ba59ba77042df1115862076bea4bd15841c838fcb68d5010a614c"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用您自己的证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRegistrationCode。

get-statistics

以下代码示例演示了如何使用 get-statistics。

Amazon CLI

在设备索引中搜索聚合数据

以下 get-statistics 示例返回设备影子中的事物数量，这些事物的名为 connectivity.connected 的属性设置为 false（即未连接的设备数量）。

```
aws iot get-statistics \
    --index-name AWS_Things \
    --query-string "connectivity.connected:false"
```

输出：

```
{
    "statistics": {
        "count": 6
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的获取有关您的设备实例集的统计信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStatistics。

get-topic-rule-destination

以下代码示例演示了如何使用 get-topic-rule-destination。

Amazon CLI

获取主题规则目标

以下 get-topic-rule-destination 示例获取某个主题规则目标的信息。

```
aws iot get-topic-rule-destination \
    --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

输出：

```
{
    "topicRuleDestination": {
```

```
        "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "status": "DISABLED",
        "httpUrlProperties": {
            "confirmationUrl": "https://example.com"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用主题规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTopicRuleDestination。

get-topic-rule

以下代码示例演示了如何使用 get-topic-rule。

Amazon CLI

获取有关规则的信息

以下 get-topic-rule 示例获取有关指定规则的信息。

```
aws iot get-topic-rule \
    --rule-name MyRPiLowMoistureAlertRule
```

输出：

```
{
    "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/MyRPiLowMoistureAlertRule",
    "rule": {
        "ruleName": "MyRPiLowMoistureAlertRule",
        "sql": "SELECT * FROM '$aws/things/MyRPi/shadow/update/accepted' WHERE
 state.reported.moisture = 'low'\n                    ",
        "description": "Sends an alert whenever soil moisture level readings are too
 low.",
        "createdAt": 1558624363.0,
        "actions": [
            {
                "sns": {
                    "targetArn": "arn:aws:sns:us-
west-2:123456789012:MyRPiLowMoistureTopic",
```

```
                    "roleArn": "arn:aws:iam::123456789012:role/service-role/
MyRPiLowMoistureTopicRole",
                    "messageFormat": "RAW"
                }
            }
        ],
        "ruleDisabled": false,
        "awsIotSqlVersion": "2016-03-23"
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的查看您的规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTopicRule。

get-v2-logging-options

以下代码示例演示了如何使用 get-v2-logging-options。

Amazon CLI

列出当前的日志选项

以下 get-v2-logging-options 示例列出 Amazon IoT 的当前日志选项。

```
aws iot get-v2-logging-options
```

输出：

```
{
    "roleArn": "arn:aws:iam::094249569039:role/service-role/iotLoggingRole",
    "defaultLogLevel": "WARN",
    "disableAllLogs": false
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的相关标题。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetV2LoggingOptions。

list-active-violations

以下代码示例演示了如何使用 list-active-violations。

Amazon CLI

列出活跃违规项

以下 `list-active-violations` 示例列出指定安全配置文件的所有违规项。

```
aws iot list-active-violations \
    --security-profile-name Testprofile
```

输出：

```
{
    "activeViolations": [
        {
            "violationId": "174db59167fa474c80a652ad1583fd44",
            "thingName": "iotconsole-1560269126751-1",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "Authorization",
                "metric": "aws:num-authorization-failures",
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 10
                    },
                    "durationSeconds": 300,
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "lastViolationValue": {
                "count": 0
            },
            "lastViolationTime": 1560293700.0,
            "violationStartTime": 1560279000.0
        },
        {
            "violationId": "c8a9466a093d3b7b35cd44ca58bdbeab",
            "thingName": "TvnQoEoU",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "CellularBandwidth",
                "metric": "aws:message-byte-size",
```

```
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 128
                    },
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "lastViolationValue": {
                "count": 110
            },
            "lastViolationTime": 1560369000.0,
            "violationStartTime": 1560276600.0
        },
        {
            "violationId": "74aa393adea02e6648f3ac362beed55e",
            "thingName": "iotconsole-1560269232412-2",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "Authorization",
                "metric": "aws:num-authorization-failures",
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 10
                    },
                    "durationSeconds": 300,
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "lastViolationValue": {
                "count": 0
            },
            "lastViolationTime": 1560276600.0,
            "violationStartTime": 1560276600.0
        },
        {
            "violationId": "1e6ab5f7cf39a1466fcd154e1377e406",
            "thingName": "TvnQoEoU",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "Authorization",
```

```
                "metric": "aws:num-authorization-failures",
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 10
                    },
                    "durationSeconds": 300,
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "lastViolationValue": {
                "count": 0
            },
            "lastViolationTime": 1560369000.0,
            "violationStartTime": 1560276600.0
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListActiveViolations。

list-attached-policies

以下代码示例演示了如何使用 list-attached-policies。

Amazon CLI

示例 1：列出附加到组的策略

以下 list-attached-policies 示例列出附加到指定组的策略。

```
aws iot list-attached-policies \
    --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"
```

输出：

```
{
    "policies": [
        {
            "policyName": "UpdateDeviceCertPolicy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
UpdateDeviceCertPolicy"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

示例 2：列出附加到设备证书的策略

以下 `list-attached-policies` 示例列出附加到设备证书的 Amazon IoT 策略。证书通过其 ARN 来标识。

```
aws iot list-attached-policies \
    --target arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

输出：

```
{
    "policies": [
        {
            "policyName": "TemperatureSensorPolicy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TemperatureSensorPolicy"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttachedPolicies。

`list-audit-findings`

以下代码示例演示了如何使用 `list-audit-findings`。

Amazon CLI

示例 1：列出审计的所有调查发现

以下 `list-audit-findings` 示例列出具有指定作业 ID 的 Amazon IoT Device Defender 审计的所有调查发现。

```
aws iot list-audit-findings \
```

```
    --task-id a3aea009955e501a31b764abe1bebd3d
```

输出：

```
{
    "findings": []
}
```

示例 2：列出审计检查类型的调查发现

以下 `list-audit-findings` 示例显示 2019 年 6 月 5 日至 2019 年 6 月 19 日期间运行的 Amazon IoT Device Defender 审计的调查发现，其中多个设备共享一个设备证书。当您指定检查名称时，必须提供开始和结束时间。

```
aws iot list-audit-findings \
    --check-name DEVICE_CERTIFICATE_SHARED_CHECK \
    --start-time 1559747125 \
    --end-time 1560962028
```

输出：

```
{
    "findings": [
        {
            "taskId": "eeef61068b0eb03c456d746c5a26ee04",
            "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",
            "taskStartTime": 1560161017.172,
            "findingTime": 1560161017.592,
            "severity": "CRITICAL",
            "nonCompliantResource": {
                "resourceType": "DEVICE_CERTIFICATE",
                "resourceIdentifier": {
                    "deviceCertificateId":
 "b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"
                }
            },
            "relatedResources": [
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "ZipxgAIl"
                    },
```

```
                    "additionalInfo": {
                        "CONNECTION_TIME": "1560086374068"
                    }
                },
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "ZipxgAIl"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1560081552187",
                        "DISCONNECTION_TIME": "1560086371552"
                    }
                },
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "ZipxgAIl"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1559289863631",
                        "DISCONNECTION_TIME": "1560081532716"
                    }
                }
            ],
            "reasonForNonCompliance": "Certificate shared by one or more devices.",
            "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
        },
        {
            "taskId": "bade6b5efd2e1b1569822f6021b39cf5",
            "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",
            "taskStartTime": 1559988217.27,
            "findingTime": 1559988217.655,
            "severity": "CRITICAL",
            "nonCompliantResource": {
                "resourceType": "DEVICE_CERTIFICATE",
                "resourceIdentifier": {
                    "deviceCertificateId":
"b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"
                }
            },
            "relatedResources": [
                {
                    "resourceType": "CLIENT_ID",
```

```
                    "resourceIdentifier": {
                        "clientId": "xShGENLW"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1559972350825"
                    }
                },
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "xShGENLW"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1559255062002",
                        "DISCONNECTION_TIME": "1559972350616"
                    }
                }
            ],
            "reasonForNonCompliance": "Certificate shared by one or more devices.",
            "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
        },
        {
            "taskId": "c23f6233ba2d35879c4bb2810fb5ffd6",
            "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",
            "taskStartTime": 1559901817.31,
            "findingTime": 1559901817.767,
            "severity": "CRITICAL",
            "nonCompliantResource": {
                "resourceType": "DEVICE_CERTIFICATE",
                "resourceIdentifier": {
                    "deviceCertificateId":
"b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"
                }
            },
            "relatedResources": [
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "TvnQoEoU"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1559826729768"
                    }
                },
```

```
                {
                    "resourceType": "CLIENT_ID",
                    "resourceIdentifier": {
                        "clientId": "TvnQoEoU"
                    },
                    "additionalInfo": {
                        "CONNECTION_TIME": "1559345920964",
                        "DISCONNECTION_TIME": "1559826728402"
                    }
                }
            ],
            "reasonForNonCompliance": "Certificate shared by one or more devices.",
            "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuditFindings。

`list-audit-mitigation-actions-executions`

以下代码示例演示了如何使用 `list-audit-mitigation-actions-executions`。

Amazon CLI

列出审计缓解操作执行的详细信息

审计缓解操作任务针对 Amazon IoT Device Defender 审计的一个或多个调查发现应用缓解操作。以下 `list-audit-mitigation-actions-executions` 示例列出具有指定 `taskId` 的缓解操作任务以及指定调查发现的详细信息。

```
aws iot list-audit-mitigation-actions-executions \
    --task-id myActionsTaskId \
    --finding-id 0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464
```

输出：

```
{
    "actionsExecutions": [
        {
            "taskId": "myActionsTaskId",
```

```
            "findingId": "0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464",
            "actionName": "ResetPolicyVersionAction",
            "actionId": "1ea0b415-bef1-4a01-bd13-72fb63c59afb",
            "status": "COMPLETED",
            "startTime": "2019-12-10T15:19:13.279000-08:00",
            "endTime": "2019-12-10T15:19:13.337000-08:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 ListAuditMitigationActionsExecutions（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuditMitigationActionsExecutions。

`list-audit-mitigation-actions-tasks`

以下代码示例演示了如何使用 `list-audit-mitigation-actions-tasks`。

Amazon CLI

列出审计缓解操作任务

以下 `list-audit-mitigation-actions-tasks` 示例列出应用于指定时间段内的调查发现的缓解操作。

```
aws iot list-audit-mitigation-actions-tasks \
    --start-time 1594157400 \
    --end-time 1594157430
```

输出：

```
{
    "tasks": [
        {
                "taskId": "0062f2d6-3999-488f-88c7-bef005414103",
                "startTime": "2020-07-07T14:30:15.172000-07:00",
            "taskStatus": "COMPLETED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 ListAuditMitigationActionsTasks（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuditMitigationActionsTasks。

`list-audit-suppressions`

以下代码示例演示了如何使用 `list-audit-suppressions`。

Amazon CLI

列出所有审计调查发现抑制

以下 `list-audit-suppressions` 示例列出所有活跃的审计调查发现抑制。

```
aws iot list-audit-suppressions
```

输出：

```
{
    "suppressions": [
        {
        "checkName": "DEVICE_CERTIFICATE_EXPIRING_CHECK",
            "resourceIdentifier": {
                "deviceCertificateId": "c7691e<shortened>"
            },
        "expirationDate": 1597881600.0,
        "suppressIndefinitely": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计调查发现抑制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuditSuppressions。

`list-audit-tasks`

以下代码示例演示了如何使用 `list-audit-tasks`。

Amazon CLI

列出审计的所有调查发现

以下 list-audit-tasks 示例列出在 2019 年 6 月 5 日至 2019 年 6 月 12 日之间运行的审计任务。

```
aws iot list-audit-tasks \
    --start-time 1559747125 \
    --end-time 1560357228
```

输出：

```
{
    "tasks": [
        {
            "taskId": "a3aea009955e501a31b764abe1bebd3d",
            "taskStatus": "COMPLETED",
            "taskType": "ON_DEMAND_AUDIT_TASK"
        },
        {
            "taskId": "f76b4b5102b632cd9ae38a279c266da1",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "51d9967d9f9ff4d26529505f6d2c444a",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "eeef61068b0eb03c456d746c5a26ee04",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "041c49557b7c7b04c079a49514b55589",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "82c7f2afac1562d18a4560be73998acc",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "bade6b5efd2e1b1569822f6021b39cf5",
```

```
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "c23f6233ba2d35879c4bb2810fb5ffd6",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        },
        {
            "taskId": "ac9086b7222a2f5e2e17bb6fd30b3aeb",
            "taskStatus": "COMPLETED",
            "taskType": "SCHEDULED_AUDIT_TASK"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuditTasks。

`list-authorizers`

以下代码示例演示了如何使用 `list-authorizers`。

Amazon CLI

列出您的自定义授权方

以下 `list-authorizers` 示例列出您的 Amazon 账户中的自定义授权方。

```
aws iot list-authorizers
```

输出：

```
{
    "authorizers": [
        {
            "authorizerName": "CustomAuthorizer",
            "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer"
        },
        {
```

```
            "authorizerName": "CustomAuthorizer2",
            "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer2"
        },
        {
            "authorizerName": "CustomAuthorizer3",
            "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer3"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListAuthorizers。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAuthorizers。

`list-billing-groups`

以下代码示例演示了如何使用 `list-billing-groups`。

Amazon CLI

列出您的 Amazon 账户和区域的账单组

以下 `list-billing-groups` 示例列出为您的 Amazon 账户和 Amazon 区域定义的所有账单组。

```
aws iot list-billing-groups
```

输出：

```
{
    "billingGroups": [
        {
            "groupName": "GroupOne",
            "groupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBillingGroups。

list-ca-certificates

以下代码示例演示了如何使用 list-ca-certificates。

Amazon CLI

列出在您的 Amazon 账户中注册的 CA 证书

以下 list-ca-certificates 示例列出在您的 Amazon 账户中注册的 CA 证书。

```
aws iot list-ca-certificates
```

输出：

```
{
    "certificates": [
        {
            "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/
f4efed62c0142f16af278166f61962501165c4f053629520742646005 8cd1467",
            "certificateId":
 "f4efed62c0142f16af278166f61962501165c4f053629520742646005 8cd1467",
            "status": "INACTIVE",
            "creationDate": 1569365372.053
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用您自己的证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCaCertificates。

list-certificates-by-ca

以下代码示例演示了如何使用 list-certificates-by-ca。

Amazon CLI

列出所有使用 CA 证书签名的设备证书

以下 list-certificates-by-ca 示例列出您的 Amazon 账户中使用指定 CA 证书签名的所有设备证书。

```
aws iot list-certificates-by-ca \
```

```
    --ca-certificate-
id f4efed62c0142f16af278166f61962501165c4f053629520742646005 8cd1467
```

输出：

```
{
    "certificates": [
        {
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
            "certificateId":
 "488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
            "status": "ACTIVE",
            "creationDate": 1569363250.557
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListCertificatesByCA。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCertificatesByCa。

list-certificates

以下代码示例演示了如何使用 list-certificates。

Amazon CLI

示例 1：列出在您的 Amazon 账户中注册的证书

以下 list-certificates 示例列出在您的账户中注册的所有证书。如果您的分页数超过默认限制值 25，则您可以使用此命令中的 nextMarker 响应值并将其提供给下一个命令来获取下一批结果。重复此操作，直到 nextMarker 返回时没有值。

```
aws iot list-certificates
```

输出：

```
{
    "certificates": [
        {
```

```
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36",
            "certificateId":
 "604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36",
            "status": "ACTIVE",
            "creationDate": 1556810537.617
        },
        {
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/262a1ac8a7d8aa72f6e96e365480f7313aa9db74b8339ec65d34dc3074e1c31e",
            "certificateId":
 "262a1ac8a7d8aa72f6e96e365480f7313aa9db74b8339ec65d34dc3074e1c31e",
            "status": "ACTIVE",
            "creationDate": 1546447050.885
        },
        {
            "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/
b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b",
            "certificateId":
 "b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b",
            "status": "ACTIVE",
            "creationDate": 1546292258.322
        },
        {
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/7aebeea3845d14a44ec80b06b8b78a89f3f8a706974b8b34d18f5adf0741db42",
            "certificateId":
 "7aebeea3845d14a44ec80b06b8b78a89f3f8a706974b8b34d18f5adf0741db42",
            "status": "ACTIVE",
            "creationDate": 1541457693.453
        },
        {
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/54458aa39ebb3eb39c91ffbbdcc3a6ca1c7c094d1644b889f735a6fc2cd9a7e3",
            "certificateId":
 "54458aa39ebb3eb39c91ffbbdcc3a6ca1c7c094d1644b889f735a6fc2cd9a7e3",
            "status": "ACTIVE",
            "creationDate": 1541113568.611
        },
        {
            "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
            "certificateId":
 "4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
```

```
            "status": "ACTIVE",
            "creationDate": 1541022751.983
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCertificates。

`list-custom-metrics`

以下代码示例演示了如何使用 `list-custom-metrics`。

Amazon CLI

列出您的自定义指标

以下 `list-custom-metrics` 示例列出您的所有自定义指标。

```
aws iot list-custom-metrics \
    --region us-east-1
```

输出：

```
{
    "metricNames": [
        "batteryPercentage"
    ]
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的自定义指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCustomMetrics。

`list-dimensions`

以下代码示例演示了如何使用 `list-dimensions`。

Amazon CLI

列出您的 Amazon 账户的维度

以下 `list-dimensions` 示例列出在您的 Amazon 账户中定义的所有 Amazon IoT Device Defender 维度。

```
aws iot list-dimensions
```

输出：

```
{
    "dimensionNames": [
        "TopicFilterForAuthMessages",
        "TopicFilterForActivityMessages"
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDimensions。

`list-domain-configurations`

以下代码示例演示了如何使用 `list-domain-configurations`。

Amazon CLI

列出域配置

以下 `list-domain-configurations` 示例列出您的 Amazon 账户中具有指定服务类型的域配置。

```
aws iot list-domain-configurations \
    --service-type "DATA"
```

输出：

```
{
    "domainConfigurations":
    [
        {
            "domainConfigurationName": "additionalDataDomain",
            "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh",
            "serviceType": "DATA"
```

```
        },

        {
            "domainConfigurationName": "iot:Jobs",
            "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:Jobs",
            "serviceType": "JOBS"
        },
        {
            "domainConfigurationName": "iot:Data-ATS",
            "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:Data-ATS",
            "serviceType": "DATA"
        },
        {
            "domainConfigurationName": "iot:CredentialProvider",
            "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:CredentialProvider",
            "serviceType": "CREDENTIAL_PROVIDER"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的可配置的端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDomainConfigurations。

`list-indices`

以下代码示例演示了如何使用 `list-indices`。

Amazon CLI

列出已配置的搜索索引

以下 `list-indices` 示例列出您的 Amazon 账户中所有已配置的搜索索引。如果您尚未启用事物索引，则可能没有任何索引。

```
aws iot list-indices
```

输出：

```
{
```

```
    "indexNames": [
        "AWS_Things"
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的管理事物索引。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIndices。

`list-job-executions-for-job`

以下代码示例演示了如何使用 `list-job-executions-for-job`。

Amazon CLI

列出您的 Amazon 账户中的作业

以下 `list-job-executions-for-job` 示例列出您的 Amazon 账户中由 jobId 指定的某个作业的所有作业执行情况。

```
aws iot list-job-executions-for-job \
    --job-id my-ota-job
```

输出：

```
{
     "executionSummaries": [
         {
             "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/my_thing",
             "jobExecutionSummary": {
                 "status": "QUEUED",
                 "queuedAt": "2022-03-07T15:58:42.195000-08:00",
                 "lastUpdatedAt": "2022-03-07T15:58:42.195000-08:00",
                 "executionNumber": 1,
                 "retryAttempt": 0
             }
         }
     ]
 }
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobExecutionsForJob。

list-job-executions-for-thing

以下代码示例演示了如何使用 list-job-executions-for-thing。

Amazon CLI

列出为某个事物执行的作业

以下 list-job-executions-for-thing 示例列出为名为 MyRaspberryPi 的事物执行的所有作业。

```
aws iot list-job-executions-for-thing \
    --thing-name "MyRaspberryPi"
```

输出：

```
{
    "executionSummaries": [
        {
            "jobId": "example-job-01",
            "jobExecutionSummary": {
                "status": "QUEUED",
                "queuedAt": 1560787023.636,
                "lastUpdatedAt": 1560787023.636,
                "executionNumber": 1
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobExecutionsForThing。

list-jobs

以下代码示例演示了如何使用 list-jobs。

Amazon CLI

列出您的 Amazon 账户中的作业

以下 list-jobs 示例列出您的 Amazon 账户中的所有作业并按作业状态排序。

```
aws iot list-jobs
```

输出：

```
{
    "jobs": [
        {
            "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
            "jobId": "example-job-01",
            "targetSelection": "SNAPSHOT",
            "status": "IN_PROGRESS",
            "createdAt": 1560787022.733,
            "lastUpdatedAt": 1560787026.294
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

list-mitigation-actions

以下代码示例演示了如何使用 list-mitigation-actions。

Amazon CLI

列出所有已定义的缓解操作

以下 list-mitigation-actions 示例列出为您的 Amazon 账户和区域定义的所有缓解操作。对于每个操作，都会列出名称、ARN 和创建日期。

```
aws iot list-mitigation-actions
```

输出：

```
{
    "actionIdentifiers": [
        {
            "actionName": "DeactivateCACertAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
DeactivateCACertAction",
```

```
            "creationDate": "2019-12-10T11:12:47.574000-08:00"
        },
        {
            "actionName": "ResetPolicyVersionAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
ResetPolicyVersionAction",
            "creationDate": "2019-12-10T11:11:48.920000-08:00"
        },
        {
            "actionName": "PublishFindingToSNSAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
PublishFindingToSNSAction",
            "creationDate": "2019-12-10T11:10:49.546000-08:00"
        },
        {
            "actionName": "AddThingsToQuarantineGroupAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
            "creationDate": "2019-12-10T11:09:35.999000-08:00"
        },
        {
            "actionName": "UpdateDeviceCertAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
UpdateDeviceCertAction",
            "creationDate": "2019-12-10T11:08:44.263000-08:00"
        },
        {
            "actionName": "SampleMitigationAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
SampleMitigationAction",
            "creationDate": "2019-12-10T11:03:41.840000-08:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 ListMitigationActions（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMitigationActions。

list-mitigations-actions

以下代码示例演示了如何使用 list-mitigations-actions。

Amazon CLI

列出所有已定义的缓解操作

以下 `list-mitigations-actions` 示例列出为您的 Amazon 账户和区域定义的所有缓解操作。对于每个操作，都会列出名称、ARN 和创建日期。

```
aws iot list-mitigation-actions
```

输出：

```
{
    "actionIdentifiers": [
        {
            "actionName": "DeactivateCACertAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
DeactivateCACertAction",
            "creationDate": "2019-12-10T11:12:47.574000-08:00"
        },
        {
            "actionName": "ResetPolicyVersionAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
ResetPolicyVersionAction",
            "creationDate": "2019-12-10T11:11:48.920000-08:00"
        },
        {
            "actionName": "PublishFindingToSNSAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
PublishFindingToSNSAction",
            "creationDate": "2019-12-10T11:10:49.546000-08:00"
        },
        {
            "actionName": "AddThingsToQuarantineGroupAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
            "creationDate": "2019-12-10T11:09:35.999000-08:00"
        },
        {
            "actionName": "UpdateDeviceCertAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
UpdateDeviceCertAction",
            "creationDate": "2019-12-10T11:08:44.263000-08:00"
        },
```

```
        {
            "actionName": "SampleMitigationAction",
            "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
SampleMitigationAction",
            "creationDate": "2019-12-10T11:03:41.840000-08:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 ListMitigationActions（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMitigationsActions。

`list-ota-updates`

以下代码示例演示了如何使用 `list-ota-updates`。

Amazon CLI

列出账户的 OTA 更新

以下 `list-ota-updates` 示例列出可用的 OTA 更新。

```
aws iot list-ota-updates
```

输出：

```
{
    "otaUpdates": [
        {
            "otaUpdateId": "itsaupdate",
            "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/
itsaupdate",
            "creationDate": 1557863215.995
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListOTAUpdates。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOtaUpdates。

list-outgoing-certificates

以下代码示例演示了如何使用 list-outgoing-certificates。

Amazon CLI

列出正在转移到其他 Amazon 账户的证书

以下 list-outgoing-certificates 示例列出正在使用 transfer-certificate 命令转移到其他 Amazon 账户的所有设备证书。

```
aws iot list-outgoing-certificates
```

输出：

```
{
    "outgoingCertificates": [
        {
            "certificateArn": "arn:aws:iot:us-
west-2:030714055129:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
            "certificateId":
 "488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
            "transferredTo": "030714055129",
            "transferDate": 1569427780.441,
            "creationDate": 1569363250.557
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListOutgoingCertificates。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOutgoingCertificates。

list-policies

以下代码示例演示了如何使用 list-policies。

Amazon CLI

列出您的 Amazon 账户中定义的策略

以下 list-policies 示例列出您的 Amazon 账户中定义的所有策略。

```
aws iot list-policies
```

输出：

```
{
    "policies": [
        {
            "policyName": "UpdateDeviceCertPolicy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
UpdateDeviceCertPolicy"
        },
        {
            "policyName": "PlantIoTPolicy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/PlantIoTPolicy"
        },
        {
            "policyName": "MyPiGroup_Core-policy",
            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/MyPiGroup_Core-
policy"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicies。

`list-policy-versions`

以下代码示例演示了如何使用 `list-policy-versions`。

Amazon CLI

示例 1：查看一个策略的所有版本

以下 `list-policy-versions` 示例列出指定策略的所有版本及其创建日期。

```
aws iot list-policy-versions \
    --policy-name LightBulbPolicy
```

输出：

```
{
    "policyVersions": [
        {
            "versionId": "2",
            "isDefaultVersion": true,
            "createDate": 1559925941.924
        },
        {
            "versionId": "1",
            "isDefaultVersion": false,
            "createDate": 1559925941.924
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyVersions。

`list-principal-things`

以下代码示例演示了如何使用 `list-principal-things`。

Amazon CLI

列出附加到一个主体的事物

以下 `list-principal-things` 示例列出附加到由 ARN 指定的一个主体的事物。

```
aws iot list-principal-things \
    --principal arn:aws:iot:us-
west-2:123456789012:cert/2e1eb273792174ec2b9bf4e9b37e6c6c692345499506002a351597670552 78e8
```

输出：

```
{
    "things": [
        "DeskLamp",
        "TableLamp"
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListPrincipalThings。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPrincipalThings。

list-provisioning-template-versions

以下代码示例演示了如何使用 list-provisioning-template-versions。

Amazon CLI

列出预配模板版本

以下 list-provisioning-template-versions 示例列出指定预配模板的可用版本。

```
aws iot list-provisioning-template-versions \
    --template-name "widget-template"
```

输出：

```
{
    "versions": [
        {
            "versionId": 1,
            "creationDate": 1574800471.339,
            "isDefaultVersion": true
        },
        {
            "versionId": 2,
            "creationDate": 1574801192.317,
            "isDefaultVersion": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProvisioningTemplateVersions。

list-provisioning-templates

以下代码示例演示了如何使用 list-provisioning-templates。

Amazon CLI

列出预配模板

以下 `list-provisioning-templates` 示例列出您的 Amazon 账户中的所有预配模板。

```
aws iot list-provisioning-templates
```

输出：

```
{
    "templates": [
        {
            "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/
widget-template",
            "templateName": "widget-template",
            "description": "A provisioning template for widgets",
            "creationDate": 1574800471.367,
            "lastModifiedDate": 1574801192.324,
            "enabled": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProvisioningTemplates。

`list-role-aliases`

以下代码示例演示了如何使用 `list-role-aliases`。

Amazon CLI

列出您的 Amazon 账户中的 Amazon IoT 角色别名

以下 `list-role-aliases` 示例列出您的 Amazon 账户中的 Amazon IoT 角色别名。

```
aws iot list-role-aliases
```

输出：

```
{
    "roleAliases": [
        "ResidentAlias",
        "ElectricianAlias"
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListRoleAliases。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoleAliases。

list-scheduled-audits

以下代码示例演示了如何使用 list-scheduled-audits。

Amazon CLI

列出您的 Amazon 账户的预定审计

以下 list-scheduled-audits 示例列出为您的 Amazon 账户安排的所有审计。

```
aws iot list-scheduled-audits
```

输出：

```
{
    "scheduledAudits": [
        {
            "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",
            "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSIoTDeviceDefenderDailyAudit",
            "frequency": "DAILY"
        },
        {
            "scheduledAuditName": "AWSDeviceDefenderWeeklyAudit",
            "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSDeviceDefenderWeeklyAudit",
            "frequency": "WEEKLY",
            "dayOfWeek": "SUN"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListScheduledAudits。

`list-security-profiles-for-target`

以下代码示例演示了如何使用 `list-security-profiles-for-target`。

Amazon CLI

列出附加到某个目标的安全配置文件

以下 `list-security-profiles-for-target` 示例列出附加到未注册设备的 Amazon IoT Device Defender 安全配置文件。

```
aws iot list-security-profiles-for-target  \
    --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/
unregistered-things"
```

输出：

```
{
    "securityProfileTargetMappings": [
        {
            "securityProfileIdentifier": {
                "name": "Testprofile",
                "arn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
Testprofile"
            },
            "target": {
                "arn": "arn:aws:iot:us-west-2:123456789012:all/unregistered-things"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecurityProfilesForTarget。

`list-security-profiles`

以下代码示例演示了如何使用 `list-security-profiles`。

Amazon CLI

列出您的 Amazon 账户的安全配置文件

以下 `list-security-profiles` 示例列出在您的 Amazon 账户中定义的所有 Amazon IoT Device Defender 安全配置文件。

```
aws iot list-security-profiles
```

输出：

```
{
    "securityProfileIdentifiers": [
        {
            "name": "Testprofile",
            "arn": "arn:aws:iot:us-west-2:123456789012:securityprofile/Testprofile"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecurityProfiles。

`list-streams`

以下代码示例演示了如何使用 `list-streams`。

Amazon CLI

列出账户中的流

以下 `list-streams` 示例列出您的 Amazon 账户中的所有流。

```
aws iot list-streams
```

输出：

```
{
    "streams": [
        {
            "streamId": "stream12345",
```

```
            "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
            "streamVersion": 1,
            "description": "This stream is used for Amazon FreeRTOS OTA Update
 12345."
        },
        {
            "streamId": "stream54321",
            "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream54321",
            "streamVersion": 1,
            "description": "This stream is used for Amazon FreeRTOS OTA Update
 54321."
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListStreams。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreams。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

显示与资源关联的标签及其值

以下 list-tags-for-resource 示例显示与事物组 LightBulbs 关联的标签和值。

```
aws iot list-tags-for-resource \
    --resource-arn "arn:aws:iot:us-west-2:094249569039:thinggroup/LightBulbs"
```

输出：

```
{
    "tags": [
        {
            "Key": "Assembly",
            "Value": "Fact1NW"
        },
        {
            "Key": "MyTag",
            "Value": "777"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的标记您的 Amazon IoT 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-targets-for-policy`

以下代码示例演示了如何使用 `list-targets-for-policy`。

Amazon CLI

列出与 Amazon IoT 策略关联的主体

以下 `list-targets-for-policy` 示例列出附加了指定策略的设备证书。

```
aws iot list-targets-for-policy \
    --policy-name UpdateDeviceCertPolicy
```

输出：

```
{
    "targets": [
        "arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
        "arn:aws:iot:us-west-2:123456789012:cert/
d1eb269fb55a628552143c8f96eb3c258fcd5331ea113e766ba0c82bf225f0be"
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetsForPolicy。

`list-targets-for-security-profile`

以下代码示例演示了如何使用 `list-targets-for-security-profile`。

Amazon CLI

列出应用了安全配置文件的目标

以下 list-targets-for-security-profile 示例列出应用了名为 PossibleIssue 的 Amazon IoT Device Defender 安全配置文件的目标。

```
aws iot list-targets-for-security-profile \
    --security-profile-name Testprofile
```

输出：

```
{
    "securityProfileTargets": [
        {
            "arn": "arn:aws:iot:us-west-2:123456789012:all/unregistered-things"
        },
        {
            "arn": "arn:aws:iot:us-west-2:123456789012:all/registered-things"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetsForSecurityProfile。

list-thing-groups-for-thing

以下代码示例演示了如何使用 list-thing-groups-for-thing。

Amazon CLI

列出事物所属的组

以下 list-thing-groups-for-thing 示例列出指定事物所属的组。

```
aws iot list-thing-groups-for-thing \
    --thing-name MyLightBulb
```

输出：

```
{
    "thingGroups": [
        {
            "groupName": "DeadBulbs",
```

```
            "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/DeadBulbs"
        },
        {
            "groupName": "LightBulbs",
            "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingGroupsForThing。

`list-thing-groups`

以下代码示例演示了如何使用 `list-thing-groups`。

Amazon CLI

列出您的 Amazon 账户中定义的事物组

以下 `describe-thing-group` 示例列出您的 Amazon 账户中定义的所有事物组。

```
aws iot list-thing-groups
```

输出：

```
{
    "thingGroups": [
        {
            "groupName": "HalogenBulbs",
            "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs"
        },
        {
            "groupName": "LightBulbs",
            "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingGroups。

list-thing-principals

以下代码示例演示了如何使用 list-thing-principals。

Amazon CLI

列出与事物关联的主体

以下 list-thing-principals 示例列出与指定事物关联的主体（X.509 证书、IAM 用户、组、角色、Amazon Cognito 身份或联合身份）。

```
aws iot list-thing-principals \
    --thing-name MyRaspberryPi
```

输出：

```
{
    "principals": [
        "arn:aws:iot:us-
west-2:123456789012:cert/33475ac865079a5ffd5ecd44240640349293facc760642d7d8d5dbb6b4c86893"
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListThingPrincipals。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingPrincipals。

list-thing-types

以下代码示例演示了如何使用 list-thing-types。

Amazon CLI

列出已定义的事物类型

以下 list-thing-types 示例显示您的 Amazon 账户中定义的事物类型列表。

```
aws iot list-thing-types
```

输出：

```
{
```

```
    "thingTypes": [
        {
            "thingTypeName": "LightBulb",
            "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/
LightBulb",
            "thingTypeProperties": {
                "thingTypeDescription": "light bulb type",
                "searchableAttributes": [
                    "model",
                    "wattage"
                ]
            },
            "thingTypeMetadata": {
            "deprecated": false,
            "creationDate": 1559772562.498
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingTypes。

list-things-in-billing-group

以下代码示例演示了如何使用 list-things-in-billing-group。

Amazon CLI

列出账单组中的事物

以下 list-things-in-billing-group 示例列出指定账单组中的事物。

```
aws iot list-things-in-billing-group \
    --billing-group-name GroupOne
```

输出：

```
{
    "things": [
        "MyOtherLightBulb",
        "MyLightBulb"
```

```
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingsInBillingGroup。

`list-things-in-thing-group`

以下代码示例演示了如何使用 `list-things-in-thing-group`。

Amazon CLI

列出属于某个组的事物

以下 `list-things-in-thing-group` 示例列出属于指定事物组的事物。

```
aws iot list-things-in-thing-group \
    --thing-group-name LightBulbs
```

输出：

```
{
    "things": [
        "MyLightBulb"
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThingsInThingGroup。

`list-things`

以下代码示例演示了如何使用 `list-things`。

Amazon CLI

示例 1：列出注册表中的所有事物

以下 `list-things` 示例列出在您的 Amazon 账户的 Amazon物联网注册表中定义的事物（设备）。

```
aws iot list-things
```

输出：

```
{
    "things": [
        {
            "thingName": "ThirdBulb",
            "thingTypeName": "LightBulb",
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/ThirdBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "version": 2
        },
        {
            "thingName": "MyOtherLightBulb",
            "thingTypeName": "LightBulb",
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "version": 3
        },
        {
            "thingName": "MyLightBulb",
            "thingTypeName": "LightBulb",
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "version": 1
        },
        {
        "thingName": "SampleIoTThing",
        "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/SampleIoTThing",
        "attributes": {},
        "version": 1
        }
    ]
```

```
}
```

示例 2：列出具有特定属性的已定义事物

以下 list-things 示例显示了具有名为 wattage 的属性的事物列表。

```
aws iot list-things \
    --attribute-name wattage
```

输出：

```
{
    "things": [
        {
            "thingName": "MyLightBulb",
            "thingTypeName": "LightBulb",
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "version": 1
        },
        {
            "thingName": "MyOtherLightBulb",
            "thingTypeName": "LightBulb",
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "version": 3
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的如何使用注册表管理事物。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListThings。

list-topic-rule-destinations

以下代码示例演示了如何使用 list-topic-rule-destinations。

Amazon CLI

列出您的主题规则目标

以下 `list-topic-rule-destinations` 示例列出您在当前 Amazon 区域中定义的所有主题规则目标。

```
aws iot list-topic-rule-destinations
```

输出：

```
{
    "destinationSummaries": [
        {
            "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "status": "ENABLED",
            "httpUrlSummary": {
                "confirmationUrl": "https://example.com"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用主题规则目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTopicRuleDestinations。

`list-topic-rules`

以下代码示例演示了如何使用 `list-topic-rules`。

Amazon CLI

列出您的规则

以下 `list-topic-rules` 示例列出您定义的所有规则。

```
aws iot list-topic-rules
```

输出：

```
{
    "rules": [
        {
            "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/
MyRPiLowMoistureAlertRule",
            "ruleName": "MyRPiLowMoistureAlertRule",
            "topicPattern": "$aws/things/MyRPi/shadow/update/accepted",
            "createdAt": 1558624363.0,
            "ruleDisabled": false
        },
        {
            "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/
MyPlantPiMoistureAlertRule",
            "ruleName": "MyPlantPiMoistureAlertRule",
            "topicPattern": "$aws/things/MyPlantPi/shadow/update/accepted",
            "createdAt": 1541458459.0,
            "ruleDisabled": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的查看您的规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTopicRules。

list-v2-logging-levels

以下代码示例演示了如何使用 list-v2-logging-levels。

Amazon CLI

列出日志级别

以下 list-v2-logging-levels 示例列出配置的日志级别。如果未设置日志级别，则运行此命令时会出现 NotConfiguredException。

```
aws iot list-v2-logging-levels
```

输出：

```
{
    "logTargetConfigurations": [
```

```
        {
            "logTarget": {
                "targetType": "DEFAULT"
            },
            "logLevel": "ERROR"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListV2LoggingLevels。

list-violation-events

以下代码示例演示了如何使用 list-violation-events。

Amazon CLI

列出一段时间内安全配置文件违规情况

以下 list-violation-events 示例列出在 2019 年 6 月 5 日至 2019 年 6 月 12 日期间，当前 Amazon 账户和 Amazon 区域的所有 Amazon IoT Device Defender 安全配置文件违规情况。

```
aws iot list-violation-events \
    --start-time 1559747125 \
    --end-time 1560351925
```

输出：

```
{
    "violationEvents": [
        {
            "violationId": "174db59167fa474c80a652ad1583fd44",
            "thingName": "iotconsole-1560269126751-1",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "Authorization",
                "metric": "aws:num-authorization-failures",
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 10
                    },
```

```
                "durationSeconds": 300,
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        },
        "metricValue": {
            "count": 0
        },
        "violationEventType": "in-alarm",
        "violationEventTime": 1560279000.0
    },
    {
        "violationId": "c8a9466a093d3b7b35cd44ca58bdbeab",
        "thingName": "TvnQoEoU",
        "securityProfileName": "Testprofile",
        "behavior": {
            "name": "CellularBandwidth",
            "metric": "aws:message-byte-size",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 128
                },
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        },
        "metricValue": {
            "count": 110
        },
        "violationEventType": "in-alarm",
        "violationEventTime": 1560276600.0
    },
    {
        "violationId": "74aa393adea02e6648f3ac362beed55e",
        "thingName": "iotconsole-1560269232412-2",
        "securityProfileName": "Testprofile",
        "behavior": {
            "name": "Authorization",
            "metric": "aws:num-authorization-failures",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 10
```

```
                    },
                    "durationSeconds": 300,
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "metricValue": {
                "count": 0
            },
            "violationEventType": "in-alarm",
            "violationEventTime": 1560276600.0
        },
        {
            "violationId": "1e6ab5f7cf39a1466fcd154e1377e406",
            "thingName": "TvnQoEoU",
            "securityProfileName": "Testprofile",
            "behavior": {
                "name": "Authorization",
                "metric": "aws:num-authorization-failures",
                "criteria": {
                    "comparisonOperator": "greater-than",
                    "value": {
                        "count": 10
                    },
                    "durationSeconds": 300,
                    "consecutiveDatapointsToAlarm": 1,
                    "consecutiveDatapointsToClear": 1
                }
            },
            "metricValue": {
                "count": 0
            },
            "violationEventType": "in-alarm",
            "violationEventTime": 1560276600.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListViolationEvents。

register-ca-certificate

以下代码示例演示了如何使用 register-ca-certificate。

Amazon CLI

注册证书颁发机构（CA）证书

以下 register-ca-certificate 示例注册了 CA 证书。该命令提供 CA 证书和一个密钥验证证书，后者证明您拥有与 CA 证书关联的私钥。

```
aws iot register-ca-certificate \
    --ca-certificate file://rootCA.pem \
    --verification-cert file://verificationCert.pem
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/
f4efed62c0142f16af278166f61962501165c4f05362952074264600 58cd1467",
    "certificateId":
 "f4efed62c0142f16af278166f61962501165c4f05362952074264600 58cd1467"
 }
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 RegisterCACertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterCaCertificate。

register-certificate

以下代码示例演示了如何使用 register-certificate。

Amazon CLI

注册自签名设备证书

以下 register-certificate 示例注册由 rootCA.pem CA 证书签名的 deviceCert.pem 设备证书。必须先注册 CA 证书，然后才能使用它来注册自签名设备证书。自签名证书必须由您传递给此命令的相同 CA 证书签名。

```
aws iot register-certificate \
    --certificate-pem file://deviceCert.pem \
```

```
    --ca-certificate-pem file://rootCA.pem
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
    "certificateId":
 "488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142"
 }
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 RegisterCertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterCertificate。

register-thing

以下代码示例演示了如何使用 register-thing。

Amazon CLI

注册事物

以下 register-thing 示例使用预配模板注册事物。

```
aws iot register-thing \
    --template-body '{"Parameters":{"ThingName":
{"Type":"String"},"AWS::IoT::Certificate::Id":{"Type":"String"}},"Resources":
 {"certificate":{"Properties":{"CertificateId":
{"Ref":"AWS::IoT::Certificate::Id"},"Status":"Active"},"Type":"AWS::IoT::Certificate"},"poli
{"Properties":{"PolicyName":"MyIotPolicy"},"Type":"AWS::IoT::Policy"},"thing":
{"OverrideSettings":
{"AttributePayload":"MERGE","ThingGroups":"DO_NOTHING","ThingTypeName":"REPLACE"},"Propertie
{"AttributePayload":{},"ThingGroups":[],"ThingName":
{"Ref":"ThingName"},"ThingTypeName":"VirtualThings"},"Type":"AWS::IoT::Thing"}}' \
    --parameters '{"ThingName":"Register-thing-
trial-1","AWS::IoT::Certificate::Id":"799a9ea048a1e6aea42b55EXAMPLEf8697b4bafcd77a318a3068e3
```

输出：

```
{
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCCAkGgAwIBAgIUYLk81I35cIppobpw
```

```
HiOJ2jNjboIwDQYJKoZIhvcNAQEL
\nBQAwTTFLMEkGA1UECwxCQW1hem9uIFdlYiBTZXJ2aWNlcyBPPUFtYXpvbi
5jb20g\nSW5jLiBMPVNlYXR0bGUgU1Q9V2FzaGluZ3RvbiBDPVVTMB4XDTIwMDcyMzE2NDUw
\nOVoXDTQ5MTIzMT
IzNTk1OVowHjEcMBoGA1UEAwwTQVdTIElvVCBDZXJ0aWZpY2F0\nZTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAO71uADhdBajqTmgrMV5\nmCFfBZQRMo1MdtVoZr2X+M4MzL
+RARrtUzH9a2SMAckeX8KeblIOTKzORI
RDXnyE
\n6lVOwjgAsd0ku22rFxex4eG2ikha7pYYkvuToqA7L3TxItRvfKrxRI4ZfJoFPip4\nKqiuBJVNOGKTcQ
Hd1RNOrddwwu6kFJLeKDmEXAMPLEdUF0N+qfR9yKnZQkm
+g6Q2\nGXu7u0W3hn6nlRN8qVoka0uW12p53xM7oHVz
Gf+cxKBxlbOhGkp6yCfTSkUBm3Sp\n9zLw35kiHXVm4EVpwgNlnk6XcIGIkw8a/iy4pzmvuGAANY1/uU/
zgCjymw
ZT5S30\nBV0CAwEAAaNgMF4wHwYDVR0jBBgwFoAUGx0tCcU3q2n1WXAuUCv6hugXjKswHQYD
\nVR0OBBYEFOVtvZ
9Aj2RYFnkX7Iu01XTRUdxgMAwGA1UdEwEB/wQCMAAwDgYDVR0P\nAQH/
BAQDAgeAMA0GCSqGSIb3DQEBCwUAA4IB
AQCXCQcpOtubS5ftOsDMTcpP/jNX
\nDHyArxmjpSc2aCdmm7WX59lTKWyAdxGAvqaDVWqTo0oXI7tZ8w7aINlGi5
pXnifx\n3SBebMUoBbTktrC97yUaeL025mCFv8emDnTR/fE7PTsBKjW0g/rrfpwBxZLXDFwN
\nnqkQjy3EDfifj2
6j0xYIqqWMPogyn4srOCKynS5wMJuQZlHQOnabVwnwK4Y0Mflp
\np9+4susFUR9aT3BT1AcIwqSpzhlKhh4Iz7ND
kRn4amsUT210jg/zOO1Ow+BTHcVQ\nJly8XDu0CWSu04q6SnaBzHmlySIajxuRTP/AdfRouP1OXe
+qlbPOBcvVvF
8o\n-----END CERTIFICATE-----\n",
    "resourceArns": {
        "certificate": "arn:aws:iot:us-
west-2:571032923833:cert/799a9ea048a1e6aea42b55EXAMPLEf8697b4bafcd77a318a3068e30404b9233c",
        "thing": "arn:aws:iot:us-west-2:571032923833:thing/Register-thing-trial-1"
    }
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的通过可信用户预配。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterThing。

reject-certificate-transfer

以下代码示例演示了如何使用 reject-certificate-transfer。

Amazon CLI

拒绝证书转移

以下 reject-certificate-transfer 示例拒绝从另一个 Amazon 账户转移指定的设备证书。

```
aws iot reject-certificate-transfer \
    --certificate-
id f0f33678c7c9a046e5cc87b2b1a58dfa0beec26db78addd5e605d630e05c7fc8
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的将证书转移到另一个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectCertificateTransfer。

remove-thing-from-billing-group

以下代码示例演示了如何使用 remove-thing-from-billing-group。

Amazon CLI

从账单组移除事物

以下 remove-thing-from-billing-group 示例从一个账单组移除指定事物。

```
aws iot remove-thing-from-billing-group \
    --billing-group-name GroupOne \
    --thing-name MyOtherLightBulb
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveThingFromBillingGroup。

remove-thing-from-thing-group

以下代码示例演示了如何使用 remove-thing-from-thing-group。

Amazon CLI

从事物组中移除事物

以下 remove-thing-from-thing-group 示例从一个事物组移除指定事物。

```
aws iot remove-thing-from-thing-group \
```

```
    --thing-name bulb7 \
    --thing-group-name DeadBulbs
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的“事物组”<https://docs.aws.amazon.com/iot/latest/developerguide/thing-groups.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveThingFromThingGroup。

replace-topic-rule

以下代码示例演示了如何使用 replace-topic-rule。

Amazon CLI

更新主题的规则定义

以下 replace-topic-rule 示例更新了指定规则，以便在土壤湿度读数过低时发送 SNS 警报。

```
aws iot replace-topic-rule \
    --rule-name MyRPiLowMoistureAlertRule \
    --topic-rule-payload "{\"sql\": \"SELECT * FROM '$aws/things/MyRPi/shadow/
update/accepted' WHERE state.reported.moisture = 'low'\", \"description\": \"Sends
 an alert when soil moisture level readings are too low.\",\"actions\": [{\"sns
\":{\"targetArn\":\"arn:aws:sns:us-west-2:123456789012:MyRPiLowMoistureTopic\",
\"roleArn\":\"arn:aws:iam::123456789012:role/service-role/MyRPiLowMoistureTopicRole
\",\"messageFormat\": \"RAW\"}}],\"ruleDisabled\": false,\"awsIotSqlVersion\":
\"2016-03-23\"}"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建 Amazon IoT 规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplaceTopicRule。

search-index

以下代码示例演示了如何使用 search-index。

Amazon CLI

查询事物索引

以下 search-index 示例在 AWS_Things 索引中查询类型为 LightBulb 的事物。

```
aws iot search-index \
    --index-name "AWS_Things" \
    --query-string "thingTypeName:LightBulb"
```

输出：

```
{
    "things": [
        {
            "thingName": "MyLightBulb",
            "thingId": "40da2e73-c6af-406e-b415-15acae538797",
            "thingTypeName": "LightBulb",
            "thingGroupNames": [
                "LightBulbs",
                "DeadBulbs"
            ],
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "connectivity": {
                "connected": false
            }
        },
        {
            "thingName": "ThirdBulb",
            "thingId": "615c8455-33d5-40e8-95fd-3ee8b24490af",
            "thingTypeName": "LightBulb",
            "attributes": {
                "model": "123",
                "wattage": "75"
            },
            "connectivity": {
                "connected": false
            }
        },
        {
            "thingName": "MyOtherLightBulb",
            "thingId": "6dae0d3f-40c1-476a-80c4-1ed24ba6aa11",
            "thingTypeName": "LightBulb",
            "attributes": {
```

```
                "model": "123",
                "wattage": "75"
            },
            "connectivity": {
                "connected": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon物联网开发人员指南》中的管理事物索引。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchIndex。

set-default-authorizer

以下代码示例演示了如何使用 set-default-authorizer。

Amazon CLI

设置默认授权方

以下 set-default-authorizer 示例将名为 CustomAuthorizer 的自定义授权方设置为默认授权方。

```
aws iot set-default-authorizer \
    --authorizer-name CustomAuthorizer
```

输出：

```
{
    "authorizerName": "CustomAuthorizer",
    "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 CreateDefaultAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetDefaultAuthorizer。

set-default-policy-version

以下代码示例演示了如何使用 set-default-policy-version。

Amazon CLI

为策略设置默认版本

以下 set-default-policy-version 示例对于名为 UpdateDeviceCertPolicy 的策略将默认版本设置为 2。

```
aws iot set-default-policy-version \
    --policy-name UpdateDeviceCertPolicy \
    --policy-version-id 2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetDefaultPolicyVersion。

set-v2-logging-level

以下代码示例演示了如何使用 set-v2-logging-level。

Amazon CLI

为事物组设置日志级别

以下 set-v2-logging-level 示例对于指定事物组将日志级别设置为日志警告。

```
aws iot set-v2-logging-level \
    --log-target "{\"targetType\":\"THING_GROUP\",\"targetName\":\"LightBulbs\"}" \
    --log-level WARN
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetV2LoggingLevel。

set-v2-logging-options

以下代码示例演示了如何使用 set-v2-logging-options。

Amazon CLI

设置日志选项

以下 set-v2-logging-options 示例将默认的日志详细级别设置为 ERROR，并指定 ARN 用于日志记录。

```
aws iot set-v2-logging-options \
    --default-log-level ERROR \
    --role-arn "arn:aws:iam::094249569039:role/service-role/iotLoggingRole"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetV2LoggingOptions。

start-audit-mitigation-actions-task

以下代码示例演示了如何使用 start-audit-mitigation-actions-task。

Amazon CLI

对审计的调查发现应用缓解操作

以下 start-audit-mitigation-actions-task 示例将 ResetPolicyVersionAction 操作（清除策略）应用于指定的单个调查发现。

```
aws iot start-audit-mitigation-actions-task \
    --task-id "myActionsTaskId" \
    --target "findingIds=[\"0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464\"]" \
    --audit-check-to-actions-mapping
 "IOT_POLICY_OVERLY_PERMISSIVE_CHECK=[\"ResetPolicyVersionAction\"]" \
    --client-request-token "adhadhahda"
```

输出：

```
{
    "taskId": "myActionsTaskId"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 StartAuditMitigationActionsTask（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartAuditMitigationActionsTask。

start-on-demand-audit-task

以下代码示例演示了如何使用 start-on-demand-audit-task。

Amazon CLI

立即启动审计

以下 start-on-demand-audit-task 示例启动 Amazon IoT Device Defender 审计并执行三次证书检查。

```
aws iot start-on-demand-audit-task \
    --target-check-
names CA_CERTIFICATE_EXPIRING_CHECK DEVICE_CERTIFICATE_EXPIRING_CHECK REVOKED_CA_CERTIFICATE
```

输出：

```
{
    "taskId": "a3aea009955e501a31b764abe1bebd3d"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartOnDemandAuditTask。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源指定标签键和值

以下 tag-resource 示例将键为 Assembly 且值为 Fact1NW 的标签应用于事物组 LightBulbs。

```
aws iot tag-resource \
    --tags Key=Assembly,Value="Fact1NW" \
    --resource-arn "arn:aws:iot:us-west-2:094249569039:thinggroup/LightBulbs"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的标记您的 Amazon IoT 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`test-authorization`

以下代码示例演示了如何使用 `test-authorization`。

Amazon CLI

测试您的 Amazon IoT 策略

以下 `test-authorization` 示例测试与指定主体关联的 Amazon IoT 策略。

```
aws iot test-authorization \
    --auth-infos actionType=CONNECT,resources=arn:aws:iot:us-
east-1:123456789012:client/client1 \
    --principal arn:aws:iot:us-west-2:123456789012:cert/
aab1068f7f43ac3e3cae4b3a8aa3f308d2a750e6350507962e32c1eb465d9775
```

输出：

```
{
    "authResults": [
        {
            "authInfo": {
                "actionType": "CONNECT",
                "resources": [
                    "arn:aws:iot:us-east-1:123456789012:client/client1"
                ]
            },
            "allowed": {
                "policies": [
                    {
                        "policyName": "TestPolicyAllowed",
                        "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyAllowed"
                    }
                ]
            },
            "denied": {
                "implicitDeny": {
                    "policies": [
                        {
                            "policyName": "TestPolicyDenied",
```

```
                            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyDenied"
                        }
                    ]
                },
                "explicitDeny": {
                    "policies": [
                        {
                            "policyName": "TestPolicyExplicitDenied",
                            "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyExplicitDenied"
                        }
                    ]
                }
            },
            "authDecision": "IMPLICIT_DENY",
            "missingContextValues": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 TestAuthorization。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestAuthorization。

test-invoke-authorizer

以下代码示例演示了如何使用 test-invoke-authorizer。

Amazon CLI

测试您的自定义授权方

以下 test-invoke-authorizer 示例测试您的自定义授权方。

```
aws iot test-invoke-authorizer \
    --authorizer-name IoTAuthorizer \
    --token allow \
    --token-signature "mE0GvaHqy9nER/
FdgtJX5lXYEJ3b3vE7t1gEszc0TKGgLKWXTnPkb2AbKnOAZ8lGyoN5dVtWDWVmr25m7+
+zjbYIMk2TBvyGXhOmvKFBPkdgyA43KL6SiZy0cTqlPMcQDsP7VX2rXr7CTowCxSNKphGXdQe0/
I5dQ+JO6KUaHwCmupt0/MejKtaNwiia064j6wprOAUwG5S1IYFuRd0X
+wfo8pb0DubAIX1Ua705kuhRUcTx4SxUShEYKmN4IDEvLB6FsIr0B2wvB7y4iPmcajxzGl02ExvyCUNctCV9dYlRRGJj
```

输出：

```
{
    "isAuthenticated": true,
    "principalId": "principalId",
    "policyDocuments": [
        "{"Version":"2012-10-17",       "Statement":
[{"Action":"iot:Publish","Effect":"Allow","Resource":"arn:aws:iot:us-
west-2:123456789012:topic/customauthtesting"}]}"
    ],
    "refreshAfterInSeconds": 600,
    "disconnectAfterInSeconds": 3600
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 TestInvokeAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestInvokeAuthorizer。

`transfer-certificate`

以下代码示例演示了如何使用 `transfer-certificate`。

Amazon CLI

将设备证书转移到其他 Amazon 账户

以下 `transfer-certificate` 示例将设备证书转移到另一个 Amazon 账户。证书和 Amazon 账户通过 ID 来标识。

```
aws iot transfer-certificate \
    --certificate-
id 488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142 \
    --target-aws-account 030714055129
```

输出：

```
{
    "transferredCertificateArn": "arn:aws:iot:us-
west-2:030714055129:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的将证书转移到另一个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TransferCertificate。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签键

以下 untag-resource 示例从事物组 LightBulbs 中移除标签 MyTag 及其值。

```
command
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的标记您的 Amazon IoT 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-account-audit-configuration

以下代码示例演示了如何使用 update-account-audit-configuration。

Amazon CLI

示例 1：为审计通知启用 Amazon SNS 通知

以下 update-account-audit-configuration 示例为 Amazon IoT Device Defender 审计通知启用 Amazon SNS 通知，指定目标和用于写入该目标的角色。

```
aws iot update-account-audit-configuration \
    --audit-notification-target-configurations "SNS={targetArn=\"arn:aws:sns:us-west-2:123456789012:ddaudits\",roleArn=\"arn:aws:iam::123456789012:role/service-role/AWSIoTDeviceDefenderAudit\",enabled=true}"
```

此命令不生成任何输出。

示例 2：启用审计检查

以下 update-account-audit-configuration 示例启用名为 AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK 的 Amazon IoT Device

Defender 审计检查。如果审计检查是 Amazon 账户一个或多个预定审计的 targetCheckNames 的一部分，则不能将其禁用。

```
aws iot update-account-audit-configuration \
    --audit-check-configurations
 "{\"AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK\":{\"enabled\":true}}"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccountAuditConfiguration。

update-audit-suppression

以下代码示例演示了如何使用 update-audit-suppression。

Amazon CLI

更新审计调查发现抑制

以下 update-audit-suppression 示例将审计调查发现抑制的到期日期更新为 2020-09-21。

```
aws iot update-audit-suppression \
    --check-name DEVICE_CERTIFICATE_EXPIRING_CHECK \
    --resource-identifier deviceCertificateId=c7691e<shortened> \
    --no-suppress-indefinitely \
    --expiration-date 2020-09-21
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计调查发现抑制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAuditSuppression。

update-authorizer

以下代码示例演示了如何使用 update-authorizer。

Amazon CLI

更新自定义授权方

以下 update-authorizer 示例将 CustomAuthorizer2 状态更新为 INACTIVE。

```
aws iot update-authorizer \
    --authorizer-name CustomAuthorizer2 \
    --status INACTIVE
```

输出：

```
{
    "authorizerName": "CustomAuthorizer2",
    "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer2"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateAuthorizer。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAuthorizer。

update-billing-group

以下代码示例演示了如何使用 update-billing-group。

Amazon CLI

更新有关账单组的信息

以下 update-billing-group 示例更新指定账单组的描述。

```
aws iot update-billing-group \
    --billing-group-name GroupOne \
    --billing-group-properties "billingGroupDescription=\"Primary bulb billing group
\""
```

输出：

```
{
    "version": 2
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的账单组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBillingGroup。

update-ca-certificate

以下代码示例演示了如何使用 update-ca-certificate。

Amazon CLI

更新证书颁发机构（CA）证书

以下 update-ca-certificate 示例将指定的 CA 证书设置为“ACTIVE”状态。

```
aws iot update-ca-certificate \
    --certificate-
id f4efed62c0142f16af278166f61962501165c4f053629520742646005 8cd1467 \
    --new-status ACTIVE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateCACertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCaCertificate。

update-certificate

以下代码示例演示了如何使用 update-certificate。

Amazon CLI

更新设备证书

以下 update-certificate 示例将指定的设备证书设置为“INACTIVE”状态。

```
aws iot update-certificate \
    --certificate-
id d1eb269fb55a628552143c8f96eb3c258fcd5331ea113e766ba0c82bf225f0be \
    --new-status INACTIVE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateCertificate。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCertificate。

update-custom-metric

以下代码示例演示了如何使用 update-custom-metric。

Amazon CLI

更新自定义指标

以下 update-custom-metric 示例更新一个自定义指标以使用新的 display-name。

```
aws iot update-custom-metric \
    --metric-name batteryPercentage \
    --display-name 'remaining battery percentage on device' \
    --region us-east-1
```

输出：

```
{
    "metricName": "batteryPercentage",
    "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/
batteryPercentage",
    "metricType": "number",
    "displayName": "remaining battery percentage on device",
    "creationDate": "2020-11-17T23:01:35.110000-08:00",
    "lastModifiedDate": "2020-11-17T23:02:12.879000-08:00"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的自定义指标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCustomMetric。

update-dimension

以下代码示例演示了如何使用 update-dimension。

Amazon CLI

更新维度

以下 update-dimension 示例更新维度。

```
aws iot update-dimension \
    --name TopicFilterForAuthMessages \
```

```
    --string-values device/${iot:ClientId}/auth
```

输出：

```
{
    "name": "TopicFilterForAuthMessages",
    "lastModifiedDate": 1585866222.317,
    "stringValues": [
        "device/${iot:ClientId}/auth"
    ],
    "creationDate": 1585854500.474,
    "type": "TOPIC_FILTER",
    "arn": "arn:aws:iot:us-west-2:1234564789012:dimension/
TopicFilterForAuthMessages"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的使用维度确定指标在安全配置文件中的范围。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDimension。

update-domain-configuration

以下代码示例演示了如何使用 update-domain-configuration。

Amazon CLI

更新域配置

以下 update-domain-configuration 示例禁用指定域配置。

```
aws iot update-domain-configuration \
    --domain-configuration-name "additionalDataDomain" \
    --domain-configuration-status "DISABLED"
```

输出：

```
{
    "domainConfigurationName": "additionalDataDomain",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的可配置的端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainConfiguration。

update-dynamic-thing-group

以下代码示例演示了如何使用 update-dynamic-thing-group。

Amazon CLI

更新动态事物组

以下 update-dynamic-thing-group 示例更新指定动态事物组。它提供描述并更新查询字符串以更改组成员资格标准。

```
aws iot update-dynamic-thing-group \
    --thing-group-name "RoomTooWarm" 
    --thing-group-properties "thingGroupDescription=\"This thing group contains
 rooms warmer than 65F.\"" \
    --query-string "attributes.temperature>65"
```

输出：

```
{
    "version": 2
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的动态事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDynamicThingGroup。

update-event-configurations

以下代码示例演示了如何使用 update-event-configurations。

Amazon CLI

显示已发布的事件类型

以下 update-event-configurations 示例更新配置以在添加、更新或删除 CA 证书时启用消息。

```
aws iot update-event-configurations \
    --event-configurations "{\"CA_CERTIFICATE\":{\"Enabled\":true}}"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事件消息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEventConfigurations。

update-indexing-configuration

以下代码示例演示了如何使用 update-indexing-configuration。

Amazon CLI

启用事物索引

以下 update-indexing-configuration 示例启用事物索引，以支持使用 Amazon_Things 索引来搜索注册表数据、阴影数据和事物连接状态。

```
aws iot update-indexing-configuration
    --thing-indexing-
configuration thingIndexingMode=REGISTRY_AND_SHADOW,thingConnectivityIndexingMode=STATUS
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon物联网开发人员指南》中的管理事物索引。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIndexingConfiguration。

update-job

以下代码示例演示了如何使用 update-job。

Amazon CLI

获取作业的详细状态

以下 update-job 示例获取 ID 为 example-job-01 的作业的详细状态。

```
aws iot describe-job \
```

```
    --job-id "example-job-01"
```

输出：

```
{
    "job": {
        "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
        "jobId": "example-job-01",
        "targetSelection": "SNAPSHOT",
        "status": "IN_PROGRESS",
        "targets": [
            "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi"
        ],
        "description": "example job test",
        "presignedUrlConfig": {},
        "jobExecutionsRolloutConfig": {},
        "createdAt": 1560787022.733,
        "lastUpdatedAt": 1560787026.294,
        "jobProcessDetails": {
            "numberOfCanceledThings": 0,
            "numberOfSucceededThings": 0,
            "numberOfFailedThings": 0,
            "numberOfRejectedThings": 0,
            "numberOfQueuedThings": 1,
            "numberOfInProgressThings": 0,
            "numberOfRemovedThings": 0,
            "numberOfTimedOutThings": 0
        },
        "timeoutConfig": {}
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的创建和管理作业（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJob。

update-mitigation-action

以下代码示例演示了如何使用 update-mitigation-action。

Amazon CLI

更新缓解操作

以下 update-mitigation-action 示例更新名为 AddThingsToQuarantineGroupAction 的指定缓解操作，更改事物组名称并设置 overrideDynamicGroups 为 false。您可以通过使用 describe-mitigation-action 命令来验证更改。

```
aws iot update-mitigation-action \
    --cli-input-json "{ \"actionName\": \"AddThingsToQuarantineGroupAction\",
 \"actionParams\": { \"addThingsToThingGroupParams\": {\"thingGroupNames\":
[\"QuarantineGroup2\"],\"overrideDynamicGroups\": false}}}"
```

输出：

```
{
    "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
    "actionId": "2fd2726d-98e1-4abf-b10f-09465ccd6bfa"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 UpdateMitigationAction（缓解操作命令）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMitigationAction。

update-provisioning-template

以下代码示例演示了如何使用 update-provisioning-template。

Amazon CLI

更新预配模板

以下 update-provisioning-template 示例修改了指定预配模板的描述和角色 arn，并启用了该模板。

```
aws iot update-provisioning-template \
    --template-name widget-template \
    --enabled \
    --description "An updated provisioning template for widgets" \
    --provisioning-role-arn arn:aws:iam::504350838278:role/Provision_role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的 Amazon IoT 安全隧道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProvisioningTemplate。

update-role-alias

以下代码示例演示了如何使用 update-role-alias。

Amazon CLI

更新角色别名

以下 update-role-alias 示例更新 LightBulbRole 角色别名。

```
aws iot update-role-alias \
    --role-alias LightBulbRole \
    --role-arn arn:aws:iam::123456789012:role/lightbulbrole-001
```

输出：

```
{
    "roleAlias": "LightBulbRole",
    "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/LightBulbRole"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateRoleAlias。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoleAlias。

update-scheduled-audit

以下代码示例演示了如何使用 update-scheduled-audit。

Amazon CLI

更新预定审计定义

以下 update-scheduled-audit 示例更改了 Amazon IoT Device Defender 预定审计的目标检查名称。

```
aws iot update-scheduled-audit \
    --scheduled-audit-name WednesdayCertCheck \
    --target-check-
names CA_CERTIFICATE_EXPIRING_CHECK DEVICE_CERTIFICATE_EXPIRING_CHECK REVOKED_CA_CERTIFICATE
```

输出：

```
{
    "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
WednesdayCertCheck"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的审计命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateScheduledAudit。

update-security-profile

以下代码示例演示了如何使用 update-security-profile。

Amazon CLI

更改安全配置文件

以下 update-security-profile 示例更新了 Amazon IoT Device Defender 安全配置文件的描述和行为。

```
aws iot update-security-profile \
    --security-profile-name PossibleIssue \
    --security-profile-description "Check to see if authorization fails 12 times in
 5 minutes or if cellular bandwidth exceeds 128"  \
    --behaviors "[{\"name\":\"CellularBandwidth\",\"metric\":\"aws:message-byte-size
\",\"criteria\":{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":128},
\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}},{\"name
\":\"Authorization\",\"metric\":\"aws:num-authorization-failures\",\"criteria\":
{\"comparisonOperator\":\"less-than\",\"value\":{\"count\":12},\"durationSeconds
\":300,\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}}]"
```

输出：

```
{
    "securityProfileName": "PossibleIssue",
    "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
PossibleIssue",
    "securityProfileDescription": "check to see if authorization fails 12 times in 5
 minutes or if cellular bandwidth exceeds 128",
    "behaviors": [
```

```
        {
            "name": "CellularBandwidth",
            "metric": "aws:message-byte-size",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 128
                },
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        },
        {
            "name": "Authorization",
            "metric": "aws:num-authorization-failures",
            "criteria": {
                "comparisonOperator": "less-than",
                "value": {
                    "count": 12
                },
                "durationSeconds": 300,
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        }
    ],
    "version": 2,
    "creationDate": 1560278102.528,
    "lastModifiedDate": 1560352711.207
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecurityProfile。

update-stream

以下代码示例演示了如何使用 update-stream。

Amazon CLI

更新流

以下 update-stream 示例更新一个现有流。流版本每次递增 1。

```
aws iot update-stream \
    --cli-input-json file://update-stream.json
```

的内容update-stream.json：

```
{
    "streamId": "stream12345",
    "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
    "files": [
        {
            "fileId": 123,
            "s3Location": {
                "bucket":"codesign-ota-bucket",
                "key":"48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
            }
        }
    ]
    "roleArn": "arn:aws:iam:us-west-2:123456789012:role/service-role/
my_ota_stream_role"
}
```

输出：

```
{
    "streamId": "stream12345",
    "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
    "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
    "streamVersion": 2
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateStream。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStream。

update-thing-group

以下代码示例演示了如何使用 update-thing-group。

Amazon CLI

更新事物组的定义

以下 update-thing-group 示例更新了指定事物组的定义，更改了描述和两个属性。

```
aws iot update-thing-group \
    --thing-group-name HalogenBulbs \
    --thing-group-properties "thingGroupDescription=\"Halogen bulb group\",
 attributePayload={attributes={Manufacturer=AnyCompany,wattage=60}}"
```

输出：

```
{
    "version": 2
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateThingGroup。

update-thing-groups-for-thing

以下代码示例演示了如何使用 update-thing-groups-for-thing。

Amazon CLI

更改事物所属的组

以下 update-thing-groups-for-thing 示例将名为 MyLightBulb 的事物从名为 DeadBulbs 的组中移除的同时，将其添加到名为 replaceableItems 的组中。

```
aws iot update-thing-groups-for-thing \
    --thing-name MyLightBulb \
    --thing-groups-to-add "replaceableItems" \
    --thing-groups-to-remove "DeadBulbs"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的事物组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateThingGroupsForThing。

update-thing

以下代码示例演示了如何使用 update-thing。

Amazon CLI

将事物与事物类型关联

以下 update-thing 示例将 Amazon物联网注册表中的一个事物与一个事物类型关联。在建立这种关联时，需要为事物类型定义的属性提供值。

```
aws iot update-thing \
    --thing-name "MyOtherLightBulb" \
    --thing-type-name "LightBulb" \
    --attribute-payload "{"attributes": {"wattage":"75", "model":"123"}}"
```

此命令不生成任何输出。使用 describe-thing 命令查看结果。

有关更多信息，请参阅《Amazon物联网开发人员指南》中的事物类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateThing。

update-topic-rule-destination

以下代码示例演示了如何使用 update-topic-rule-destination。

Amazon CLI

示例 1：启用主题规则目标

以下 update-topic-rule-destination 示例启用流向主题规则目标的流量。

```
aws iot update-topic-rule-destination \
    --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \
    --status ENABLED
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的启用主题规则目标。

示例 2：禁用主题规则目标

以下 update-topic-rule-destination 示例禁用流向主题规则目标的流量。

```
aws iot update-topic-rule-destination \
    --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \
```

```
    --status DISABLED
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的禁用主题规则目标。

示例 3：发送新的确认消息

以下 update-topic-rule-destination 示例为主题规则目标发送了一条新的确认消息。

```
aws iot update-topic-rule-destination \
    --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \
    --status IN_PROGRESS
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的发送新的确认消息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTopicRuleDestination。

validate-security-profile-behaviors

以下代码示例演示了如何使用 validate-security-profile-behaviors。

Amazon CLI

示例 1：验证安全配置文件的行为参数

以下 validate-security-profile-behaviors 示例验证一个 Amazon IoT Device Defender 安全配置文件的一组格式良好且正确的行为。

```
aws iot validate-security-profile-behaviors \
    --behaviors "[{\"name\":\"CellularBandwidth\",\"metric\":\"aws:message-byte-size
\",\"criteria\":{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":128},
\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}},{\"name
\":\"Authorization\",\"metric\":\"aws:num-authorization-failures\",\"criteria\":
{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":12},\"durationSeconds
\":300,\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}}]"
```

输出：

```
{
```

```
    "valid": true,
    "validationErrors": []
}
```

示例 2：验证安全配置文件的不正确行为参数

以下 `validate-security-profile-behaviors` 示例验证 Amazon IoT Device Defender 安全配置文件的一组包含错误的行为。

```
aws iot validate-security-profile-behaviors \
    --behaviors "[{\"name\":\"CellularBandwidth\",\"metric\":\"aws:message-byte-size
\",\"criteria\":{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":128},
\"consecutiveDatapointsToAlarm\":1,\"consecutiveDatapointsToClear\":1}},{\"name
\":\"Authorization\",\"metric\":\"aws:num-authorization-failures\",\"criteria\":
{\"comparisonOperator\":\"greater-than\",\"value\":{\"count\":12},\"durationSeconds
\":300,\"consecutiveDatapointsToAlarm\":100000,\"consecutiveDatapointsToClear
\":1}}]"
```

输出：

```
{
    "valid": false,
    "validationErrors": [
        {
            "errorMessage": "Behavior Authorization is malformed.
 consecutiveDatapointsToAlarm 100000 should be in range[1,10]"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的检测命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateSecurityProfileBehaviors。

使用 Amazon CLI 的 Amazon IoT Analytics 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Analytics 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-put-message

以下代码示例演示了如何使用 batch-put-message。

Amazon CLI

将消息发送到通道

以下 batch-put-message 示例向指定通道发送消息。

```
aws iotanalytics batch-put-message \
    --cli-binary-format raw-in-base64-out \
    --cli-input-json file://batch-put-message.json
```

的内容batch-put-message.json：

```
{
    "channelName": "mychannel",
    "messages": [
        {
            "messageId": "0001",
            "payload": "eyAidGVtcGVyYXR1cmUiOiAyMCB9"
        }
    ]
}
```

输出：

```
{
    "batchPutMessageErrorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 BatchPutMessage。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchPutMessage。

cancel-pipeline-reprocessing

以下代码示例演示了如何使用 cancel-pipeline-reprocessing。

Amazon CLI

取消通过管道重新处理数据

以下 cancel-pipeline-reprocessing 示例取消了通过指定管道重新处理数据。

```
aws iotanalytics cancel-pipeline-reprocessing \
    --pipeline-name mypipeline \
    --reprocessing-id "6ad2764f-fb13-4de3-b101-4e74af03b043"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CancelPipelineReprocessing。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelPipelineReprocessing。

create-channel

以下代码示例演示了如何使用 create-channel。

Amazon CLI

创建通道

以下 create-channel 示例使用指定配置创建了一个通道。通道收集来自 MQTT 主题的数据，并在将数据发布到管道之前将原始未处理消息归档。

```
aws iotanalytics create-channel \
    --cli-input-json file://create-channel.json
```

的内容create-channel.json：

```
{
    "channelName": "mychannel",
    "retentionPeriod": {
        "unlimited": true
```

```
    },
    "tags": [
        {
            "key": "Environment",
            "value": "Production"
        }
    ]
}
```

输出：

```
{
    "channelArn": "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel",
    "channelName": "mychannel",
    "retentionPeriod": {
        "unlimited": true
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CreateChannel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChannel。

create-dataset-content

以下代码示例演示了如何使用 create-dataset-content。

Amazon CLI

创建数据集的内容

以下 create-dataset-content 示例通过应用 queryAction（SQL 查询）或 containerAction（执行容器化应用程序）来创建指定数据集的内容。

```
aws iotanalytics create-dataset-content \
    --dataset-name mydataset
```

输出：

```
{
    "versionId": "d494b416-9850-4670-b885-ca22f1e89d62"
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CreateDatasetContent。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDatasetContent。

create-dataset

以下代码示例演示了如何使用 create-dataset。

Amazon CLI

创建数据集

以下 create-dataset 示例创建一个数据集。数据集存储通过应用 queryAction（SQL 查询）或 containerAction（执行容器化应用程序）从数据存储中检索的数据。此操作创建数据集的骨架。可以通过调用 CreateDatasetContent 手动填充数据集，也可以根据您指定的 trigger 自动填充数据集。

```
aws iotanalytics create-dataset \
    --cli-input-json file://create-dataset.json
```

的内容create-dataset.json：

```
{
    "datasetName": "mydataset",
    "actions": [
        {
            "actionName": "myDatasetAction",
            "queryAction": {
                "sqlQuery": "SELECT * FROM mydatastore"
            }
        }
    ],
    "retentionPeriod": {
        "unlimited": true
    },
    "tags": [
        {
            "key": "Environment",
            "value": "Production"
        }
    ]
}
```

输出：

```
{
    "datasetName": "mydataset",
    "retentionPeriod": {
        "unlimited": true
    },
    "datasetArn": "arn:aws:iotanalytics:us-west-2:123456789012:dataset/mydataset"
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CreateDataset。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataset。

create-datastore

以下代码示例演示了如何使用 create-datastore。

Amazon CLI

创建数据存储

以下 create-datastore 示例创建了一个数据存储，它是消息的存储库。

```
aws iotanalytics create-datastore \
    --cli-input-json file://create-datastore.json
```

的内容create-datastore.json：

```
{
    "datastoreName": "mydatastore",
    "retentionPeriod": {
        "numberOfDays": 90
    },
    "tags": [
        {
            "key": "Environment",
            "value": "Production"
        }
    ]
}
```

输出：

```
{
    "datastoreName": "mydatastore",
    "datastoreArn": "arn:aws:iotanalytics:us-west-2:123456789012:datastore/
mydatastore",
    "retentionPeriod": {
        "numberOfDays": 90,
        "unlimited": false
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CreateDatastore。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDatastore。

create-pipeline

以下代码示例演示了如何使用 create-pipeline。

Amazon CLI

创建 IoT Analytics 管道

以下 create-pipeline 示例创建一个管道。管道使用来自通道的消息，并允许您在将消息存储在数据存储之前处理消息。您必须同时指定一个通道和一个数据存储活动，以及可选择性地指定 pipelineActivities 数组中最多 23 个额外活动。

```
aws iotanalytics create-pipeline \
    --cli-input-json file://create-pipeline.json
```

的内容create-pipeline.json：

```
{
    "pipelineName": "mypipeline",
    "pipelineActivities": [
        {
            "channel": {
                "name": "myChannelActivity",
                "channelName": "mychannel",
                "next": "myMathActivity"
            }
```

```
        },
        {
            "datastore": {
                "name": "myDatastoreActivity",
                "datastoreName": "mydatastore"
            }
        },
        {
            "math": {
                "name": "myMathActivity",
                "math": "((temp - 32) * 5.0) / 9.0",
                "attribute": "tempC",
                "next": "myDatastoreActivity"
            }
        }
    ],
    "tags": [
        {
            "key": "Environment",
            "value": "Beta"
        }
    ]
}
```

输出：

```
{
    "pipelineArn": "arn:aws:iotanalytics:us-west-2:123456789012:pipeline/
mypipeline",
    "pipelineName": "mypipeline"
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 CreatePipeline。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePipeline。

`delete-channel`

以下代码示例演示了如何使用 `delete-channel`。

Amazon CLI

删除 IoT Analytics 通道

以下 delete-channel 示例删除指定通道。

```
aws iotanalytics delete-channel \
    --channel-name mychannel
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DeleteChannel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteChannel。

delete-dataset-content

以下代码示例演示了如何使用 delete-dataset-content。

Amazon CLI

删除数据集内容

以下 delete-dataset-content 示例删除指定数据集的内容。

```
aws iotanalytics delete-dataset-content \
    --dataset-name mydataset
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DeleteDatasetContent。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDatasetContent。

delete-dataset

以下代码示例演示了如何使用 delete-dataset。

Amazon CLI

删除数据集

以下 delete-dataset 示例删除指定数据集。在执行此操作之前，您不必删除数据集的内容。

```
aws iotanalytics delete-dataset \
    --dataset-name mydataset
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DeleteDataset。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDataset。

delete-datastore

以下代码示例演示了如何使用 delete-datastore。

Amazon CLI

删除数据存储

以下 delete-datastore 示例删除指定数据存储。

```
aws iotanalytics delete-datastore \
    --datastore-name mydatastore
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DeleteDatastore。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDatastore。

delete-pipeline

以下代码示例演示了如何使用 delete-pipeline。

Amazon CLI

删除管道

以下 delete-pipeline 示例删除指定管道。

```
aws iotanalytics delete-pipeline \
    --pipeline-name mypipeline
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DeletePipeline。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePipeline。

describe-channel

以下代码示例演示了如何使用 describe-channel。

Amazon CLI

检索有关通道的信息

以下 describe-channel 示例显示指定通道的详细信息，包括统计信息。

```
aws iotanalytics describe-channel \
    --channel-name mychannel \
    --include-statistics
```

输出：

```
{
    "statistics": {
        "size": {
            "estimatedSizeInBytes": 402.0,
            "estimatedOn": 1561504380.0
        }
    },
    "channel": {
        "status": "ACTIVE",
        "name": "mychannel",
        "lastUpdateTime": 1557860351.001,
        "creationTime": 1557860351.001,
        "retentionPeriod": {
            "unlimited": true
        },
        "arn": "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel"
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DescribeChannel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeChannel。

describe-dataset

以下代码示例演示了如何使用 describe-dataset。

Amazon CLI

检索有关数据集的信息

以下 describe-dataset 示例显示指定数据集的详细信息。

```
aws iotanalytics describe-dataset \
    --dataset-name mydataset
```

输出：

```
{
    "dataset": {
        "status": "ACTIVE",
        "contentDeliveryRules": [],
        "name": "mydataset",
        "lastUpdateTime": 1557859240.658,
        "triggers": [],
        "creationTime": 1557859240.658,
        "actions": [
            {
                "actionName": "query_32",
                "queryAction": {
                    "sqlQuery": "SELECT * FROM mydatastore",
                    "filters": []
                }
            }
        ],
        "retentionPeriod": {
            "numberOfDays": 90,
            "unlimited": false
        },
        "arn": "arn:aws:iotanalytics:us-west-2:123456789012:dataset/mydataset"
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DescribeDataset。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDataset。

describe-datastore

以下代码示例演示了如何使用 describe-datastore。

Amazon CLI

检索有关数据存储的信息

以下 describe-datastore 示例显示指定数据存储的详细信息，包括统计信息。

```
aws iotanalytics describe-datastore \
    --datastore-name mydatastore \
    --include-statistics
```

输出：

```
{
    "datastore": {
        "status": "ACTIVE",
        "name": "mydatastore",
        "lastUpdateTime": 1557858971.02,
        "creationTime": 1557858971.02,
        "retentionPeriod": {
            "unlimited": true
        },
        "arn": "arn:aws:iotanalytics:us-west-2:123456789012:datastore/mydatastore"
    },
    "statistics": {
        "size": {
            "estimatedSizeInBytes": 397.0,
            "estimatedOn": 1561592040.0
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DescribeDatastore。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDatastore。

describe-logging-options

以下代码示例演示了如何使用 describe-logging-options。

Amazon CLI

检索当前的日志选项

以下 describe-logging-options 示例显示当前 Amazon IoT Analytics 日志选项。

```
aws iotanalytics describe-logging-options
```

此命令不生成任何输出。输出：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/service-role/myIoTAnalyticsRole",
        "enabled": true,
        "level": "ERROR"
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DescribeLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoggingOptions。

describe-pipeline

以下代码示例演示了如何使用 describe-pipeline。

Amazon CLI

检索有关管道的信息

以下 describe-pipeline 示例显示指定管道的详细信息。

```
aws iotanalytics describe-pipeline \
    --pipeline-name mypipeline
```

输出：

```
{
    "pipeline": {
        "activities": [
            {
                "channel": {
                    "channelName": "mychannel",
                    "name": "mychannel_28",
                    "next": "mydatastore_29"
                }
```

```
            },
            {
                "datastore": {
                    "datastoreName": "mydatastore",
                    "name": "mydatastore_29"
                }
            }
        ],
        "name": "mypipeline",
        "lastUpdateTime": 1561676362.515,
        "creationTime": 1557859124.432,
        "reprocessingSummaries": [
            {
                "status": "SUCCEEDED",
                "creationTime": 1561676362.189,
                "id": "6ad2764f-fb13-4de3-b101-4e74af03b043"
            }
        ],
        "arn": "arn:aws:iotanalytics:us-west-2:123456789012:pipeline/mypipeline"
    }
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 DescribePipeline。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePipeline。

get-dataset-content

以下代码示例演示了如何使用 get-dataset-content。

Amazon CLI

检索数据集的内容

以下 get-dataset-content 示例以预签名 URI 的形式检索数据集的内容。

```
aws iotanalytics get-dataset-content --dataset-name mydataset
```

输出：

```
{
    "status": {
        "state": "SUCCEEDED"
```

```
    },
    "timestamp": 1557863215.995,
    "entries": [
        {
            "dataURI": "https://aws-radiant-
dataset-12345678-1234-1234-1234-123456789012.s3.us-west-2.amazonaws.com/
results/12345678-e8b3-46ba-b2dd-efe8d86cf385.csv?X-Amz-Security-Token=...-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20190628T173437Z&X-Amz-SignedHeaders=host&X-
Amz-Expires=7200&X-Amz-Credential=...F20190628%2Fus-west-2%2Fs3%2Faws4_request&X-
Amz-Signature=..."
        }
    ]
}
```

有关更多信息，请参阅《指南》中的 GetDatasetContent。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDatasetContent。

`list-channels`

以下代码示例演示了如何使用 `list-channels`。

Amazon CLI

检索通道的列表

以下 `list-channels` 示例显示可用通道的摘要信息。

```
aws iotanalytics list-channels
```

输出：

```
{
    "channelSummaries": [
        {
            "status": "ACTIVE",
            "channelName": "mychannel",
            "creationTime": 1557860351.001,
            "lastUpdateTime": 1557860351.001
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListChannels。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChannels。

`list-dataset-contents`

以下代码示例演示了如何使用 `list-dataset-contents`。

Amazon CLI

列出有关数据集内容的信息

以下 `list-dataset-contents` 示例列出有关已创建的数据集内容的信息。

```
aws iotanalytics list-dataset-contents \
    --dataset-name mydataset
```

输出：

```
{
    "datasetContentSummaries": [
        {
            "status": {
                "state": "SUCCEEDED"
            },
            "scheduleTime": 1557863215.995,
            "version": "b10ea2a9-66c1-4d99-8d1f-518113b738d0",
            "creationTime": 1557863215.995
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListDatasetContents。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatasetContents。

`list-datasets`

以下代码示例演示了如何使用 `list-datasets`。

Amazon CLI

检索有关数据集的信息

以下 `list-datasets` 示例列出有关可用数据集的摘要信息。

```
aws iotanalytics list-datasets
```

输出：

```
{
    "datasetSummaries": [
        {
            "status": "ACTIVE",
            "datasetName": "mydataset",
            "lastUpdateTime": 1557859240.658,
            "triggers": [],
            "creationTime": 1557859240.658,
            "actions": [
                {
                    "actionName": "query_32",
                    "actionType": "QUERY"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListDatasets。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatasets。

list-datastores

以下代码示例演示了如何使用 `list-datastores`。

Amazon CLI

检索数据存储的列表

以下 `list-datastores` 示例显示有关可用数据存储的摘要信息。

```
aws iotanalytics list-datastores
```

输出：

```
{
    "datastoreSummaries": [
        {
            "status": "ACTIVE",
            "datastoreName": "mydatastore",
            "creationTime": 1557858971.02,
            "lastUpdateTime": 1557858971.02
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListDatastores。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatastores。

`list-pipelines`

以下代码示例演示了如何使用 `list-pipelines`。

Amazon CLI

检索管道的列表

以下 `list-pipelines` 示例显示可用管道列表。

```
aws iotanalytics list-pipelines
```

输出：

```
{
    "pipelineSummaries": [
        {
            "pipelineName": "mypipeline",
            "creationTime": 1557859124.432,
            "lastUpdateTime": 1557859124.432,
            "reprocessingSummaries": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListPipelines。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPipelines。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出资源标签

以下 `list-tags-for-resource` 示例列出您附加到指定资源的标签：

```
aws iotanalytics list-tags-for-resource \
    --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel"
```

输出：

```
{
    "tags": [
        {
            "value": "bar",
            "key": "foo"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 ListTagsForResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`put-logging-options`

以下代码示例演示了如何使用 `put-logging-options`。

Amazon CLI

设置或更新日志选项

以下 `put-logging-options` 示例设置或更新 Amazon IoT Analytics 日志选项。如果您更新了任何 `loggingOptions` 字段的值，则最多需要一分钟，更改就能生效。此外，如果您更改附加到

您在“roleArn”字段中所指定角色的策略（例如，更正无效策略），则最多需要 5 分钟，更改就能生效。

```
aws iotanalytics put-logging-options \
    --cli-input-json file://put-logging-options.json
```

的内容put-logging-options.json：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/service-role/myIoTAnalyticsRole",
        "level": "ERROR",
        "enabled": true
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 PutLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingOptions。

run-pipeline-activity

以下代码示例演示了如何使用 run-pipeline-activity。

Amazon CLI

模拟管道活动

以下 run-pipeline-activity 示例模拟对消息有效载荷运行管道活动的结果。

```
aws iotanalytics run-pipeline-activity \
    --pipeline-activity file://maths.json \
    --payloads file://payloads.json
```

的内容maths.json：

```
{
    "math": {
        "name": "MyMathActivity",
```

```
        "math": "((temp - 32) * 5.0) / 9.0",
        "attribute": "tempC"
    }
}
```

的内容payloads.json：

```
[
    "{\"humidity\": 52, \"temp\": 68 }",
    "{\"humidity\": 52, \"temp\": 32 }"
]
```

输出：

```
{
    "logResult": "",
    "payloads": [
        "eyJodW1pZGl0eSI6NTIsInRlbXAiOjY4LCJ0ZW1wQyI6MjB9",
        "eyJodW1pZGl0eSI6NTIsInRlbXAiOjMyLCJ0ZW1wQyI6MH0="
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 RunPipelineActivity。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RunPipelineActivity。

sample-channel-data

以下代码示例演示了如何使用 sample-channel-data。

Amazon CLI

从通道检索样本消息

以下 sample-channel-data 示例从指定时间段内接收的指定通道中检索消息样本。您最多可以检索 10 条消息。

```
aws iotanalytics sample-channel-data \
    --channel-name mychannel
```

输出：

```
{
    "payloads": [
        "eyAidGVtcGVyYXR1cmUiOiAyMCB9",
        "eyAiZm9vIjogImJhciIgfQ=="
    ]
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 SampleChannelData。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SampleChannelData。

`start-pipeline-reprocessing`

以下代码示例演示了如何使用 `start-pipeline-reprocessing`。

Amazon CLI

启动管道重新处理

以下 `start-pipeline-reprocessing` 示例启动通过指定管道重新处理原始消息数据。

```
aws iotanalytics start-pipeline-reprocessing \
    --pipeline-name mypipeline
```

输出：

```
{
    "reprocessingId": "6ad2764f-fb13-4de3-b101-4e74af03b043"
}
```

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 StartPipelineReprocessing。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPipelineReprocessing。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

添加或修改资源的标签

以下 tag-resource 示例添加或修改附加到指定资源的标签。

```
aws iotanalytics tag-resource \
    --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel" \
    --tags "[{\"key\": \"Environment\", \"value\": \"Production\"}]"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 TagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从指定资源移除具有指定键名的标签。

```
aws iotanalytics untag-resource \
    --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel" \
    --tag-keys "[\"Environment\"]"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的“UntagResource”<https://docs.aws.amazon.com/iotanalytics/latest/APIReference/API_UntagResource.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-channel

以下代码示例演示了如何使用 update-channel。

Amazon CLI

修改通道

以下 update-channel 示例修改了指定通道的设置。

```
aws iotanalytics update-channel \
    --cli-input-json file://update-channel.json
```

的内容update-channel.json：

```
{
    "channelName": "mychannel",
    "retentionPeriod": {
        "numberOfDays": 92
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 UpdateChannel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateChannel。

update-dataset

以下代码示例演示了如何使用 update-dataset。

Amazon CLI

更新数据集

以下 update-dataset 示例修改指定数据集的设置。

```
aws iotanalytics update-dataset \
    --cli-input-json file://update-dataset.json
```

的内容update-dataset.json：

```
{
    "datasetName": "mydataset",
    "actions": [
        {
            "actionName": "myDatasetUpdateAction",
            "queryAction": {
                "sqlQuery": "SELECT * FROM mydatastore"
```

```
            }
        }
    ],
    "retentionPeriod": {
        "numberOfDays": 92
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的“UpdateDataset”<https://docs.aws.amazon.com/iotanalytics/latest/APIReference/API_UpdateDataset.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDataset。

update-datastore

以下代码示例演示了如何使用 update-datastore。

Amazon CLI

更新数据存储

以下 update-datastore 示例修改指定数据存储的设置。

```
aws iotanalytics update-datastore \
    --cli-input-json file://update-datastore.json
```

update-datastore.json 的内容：

```
{
    "datastoreName": "mydatastore",
    "retentionPeriod": {
        "numberOfDays": 93
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 UpdateDatastore。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDatastore。

update-pipeline

以下代码示例演示了如何使用 update-pipeline。

Amazon CLI

更新管道

以下 update-pipeline 示例修改指定管道的设置。您必须同时指定一个通道和一个数据存储活动，以及可选择性地指定 pipelineActivities 数组中最多 23 个额外活动。

```
aws iotanalytics update-pipeline \
    --cli-input-json file://update-pipeline.json
```

update-pipeline.json 的内容：

```
{
    "pipelineName": "mypipeline",
    "pipelineActivities": [
        {
            "channel": {
                "name": "myChannelActivity",
                "channelName": "mychannel",
                "next": "myMathActivity"
            }
        },
        {
            "datastore": {
                "name": "myDatastoreActivity",
                "datastoreName": "mydatastore"
            }
        },
        {
            "math": {
                "name": "myMathActivity",
                "math": "(((temp - 32) * 5.0) / 9.0) + 273.15",
                "attribute": "tempK",
                "next": "myDatastoreActivity"
            }
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Analytics API 参考》中的 UpdatePipeline。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePipeline。

使用 Amazon CLI 的 Device Advisor 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Device Advisor 结合使用，执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-suite-definition`

以下代码示例演示了如何使用 `create-suite-definition`。

Amazon CLI

示例 1：创建 IoT Device Advisor 测试套件

以下 `create-suite-definition` 示例使用指定套件定义配置在 Amazon IoT 中创建 Device Advisor 测试套件。

```
aws iotdeviceadvisor create-suite-definition \
    --suite-definition-configuration '{ \
        "suiteDefinitionName": "TestSuiteName", \
        "devices": [{"thingArn":"arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
        "intendedForQualification": false, \
        "rootGroup": "{\"configuration\":{},\"tests\":[{\"name\":\"MQTT Connect\",
\"configuration\":{\"EXECUTION_TIMEOUT\":120},\"tests\":[{\"name\":\"MQTT_Connect\",
```

```
\"configuration\":{},\"test\":{\"id\":\"MQTT_Connect\",\"testCase\":null,\"version
\":\"0.0.0\"}}]}]}", \
        "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"}'
```

输出：

```
{
    "suiteDefinitionId": "0jtsgio7yenu",
    "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/0jtsgio7yenu",
    "suiteDefinitionName": "TestSuiteName",
    "createdAt": "2022-12-02T11:38:13.263000-05:00"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的创建测试套件定义。

示例 2：创建 IoT Device Advisor 最新合格测试套件

以下 `create-suite-definition` 示例使用指定套件定义配置在 Amazon IoT 中创建包含最新版本的 Device Advisor 合格测试套件。

```
aws iotdeviceadvisor create-suite-definition \
    --suite-definition-configuration '{ \
        "suiteDefinitionName": "TestSuiteName", \
        "devices": [{"thingArn":"arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
        "intendedForQualification": true, \
        "rootGroup": "", \
        "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"}'
```

输出：

```
{
    "suiteDefinitionId": "txgsuolk2myj",
    "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/txgsuolk2myj",
    "suiteDefinitionName": "TestSuiteName",
    "createdAt": "2022-12-02T11:38:13.263000-05:00"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的创建测试套件定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSuiteDefinition。

delete-suite-definition

以下代码示例演示了如何使用 delete-suite-definition。

Amazon CLI

删除 IoT Device Advisor 测试套件

以下 delete-suite-definition 示例删除具有指定套件定义 ID 的 Device Advisor 测试套件。

```
aws iotdeviceadvisor delete-suite-definition \
    --suite-definition-id 0jtsgio7yenu
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 DeleteSuiteDefinition。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSuiteDefinition。

get-endpoint

以下代码示例演示了如何使用 get-endpoint。

Amazon CLI

示例 1：获取有关 IoT Device Advisor 账户级端点的信息

以下 get-endpoint 示例获取有关 Device Advisor 账户级测试端点的信息。

```
aws iotdeviceadvisor get-endpoint
```

输出：

```
{
    "endpoint": "t6y4c143x9sfo.deviceadvisor.iot.us-east-1.amazonaws.com"
}
```

示例 2：获取有关 IoT Device Advisor 设备级端点的信息

以下 get-endpoint 示例获取有关具有指定 thing-arn 或 certificate-arn 的 Device Advisor 设备级测试端点的信息。

```
aws iotdeviceadvisor get-endpoint \
    --thing-arn arn:aws:iot:us-east-1:123456789012:thing/MyIotThing
```

输出：

```
{
    "endpoint": "tdb7719be5t6y4c143x9sfo.deviceadvisor.iot.us-east-1.amazonaws.com"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的获取测试端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEndpoint。

get-suite-definition

以下代码示例演示了如何使用 get-suite-definition。

Amazon CLI

获取有关 IoT Device Advisor 测试套件的信息

以下 get-suite-definition 示例获取有关具有指定套件定义 ID 的 Device Advisor 测试套件的信息。

```
aws iotdeviceadvisor get-suite-definition \
    --suite-definition-id qqcsmtyyjabl
```

输出：

```
{
    "suiteDefinitionId": "qqcsmtyyjabl",
    "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/qqcsmtyyjabl",
    "suiteDefinitionVersion": "v1",
    "latestVersion": "v1",
    "suiteDefinitionConfiguration": {
        "suiteDefinitionName": "MQTT connection",
        "devices": [],
```

```
        "intendedForQualification": false,
        "isLongDurationTest": false,
        "rootGroup": "{\"configuration\":{},\"tests\":[{\"id\":\"uta5d9j1kvwc\",
\"name\":\"Test group 1\",\"configuration\":{},\"tests\":[{\"id\":\"awr8pq5vc9yp\",
\"name\":\"MQTT Connect\",\"configuration\":{},\"test\":{\"id\":\"MQTT_Connect\",
\"testCase\":null,\"version\":\"0.0.0\"}}]}]}",
        "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole",
        "protocol": "MqttV3_1_1"
    },
    "createdAt": "2022-11-11T22:28:52.389000-05:00",
    "lastModifiedAt": "2022-11-11T22:28:52.389000-05:00",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的获取测试套件定义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSuiteDefinition。

`get-suite-run-report`

以下代码示例演示了如何使用 `get-suite-run-report`。

Amazon CLI

获取有关 IoT Device Advisor 合格测试套件运行报告的信息

以下 `get-suite-run-report` 示例获取具有指定套件定义 ID 和套件运行 ID 的成功 Device Advisor 合格测试套件运行的报告下载链接。

```
aws iotdeviceadvisor get-suite-run-report \
    --suite-definition-id ztvb5aek4w4x \
    --suite-run-id p6awv83nre6v
```

输出：

```
{
    "qualificationReportDownloadUrl": "https://senate-apn-reports-us-east-1-
prod.s3.amazonaws.com/report.downloadlink"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的获取成功合格测试套件运行的合格报告。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSuiteRunReport。

get-suite-run

以下代码示例演示了如何使用 get-suite-run。

Amazon CLI

获取有关 IoT Device Advisor 测试套件运行状态的信息

以下 get-suite-run 示例获取具有指定套件定义 ID 和套件运行 ID 的 Device Advisor 测试套件运行状态的信息。

```
aws iotdeviceadvisor get-suite-run \
    --suite-definition-id qqcsmtyyjabl \
    --suite-run-id nzlfyhaa18oa
```

输出：

```
{
    "suiteDefinitionId": "qqcsmtyyjabl",
    "suiteDefinitionVersion": "v1",
    "suiteRunId": "nzlfyhaa18oa",
    "suiteRunArn": "arn:aws:iotdeviceadvisor:us-east-1:123456789012:suiterun/
qqcsmtyyjabl/nzlfyhaa18oa",
    "suiteRunConfiguration": {
        "primaryDevice": {
            "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/MyIotThing",
            "certificateArn": "arn:aws:iot:us-east-1:123456789012:cert/certFile"
        },
        "parallelRun": false
    },
    "testResult": {
        "groups": [
            {
                "groupId": "uta5d9j1kvwc",
                "groupName": "Test group 1",
                "tests": [
                    {
                        "testCaseRunId": "2ve2twrqyr0s",
                        "testCaseDefinitionId": "awr8pq5vc9yp",
                        "testCaseDefinitionName": "MQTT Connect",
```

```
                        "status": "PASS",
                        "startTime": "2022-11-12T00:01:53.693000-05:00",
                        "endTime": "2022-11-12T00:02:15.443000-05:00",
                        "logUrl": "https://console.aws.amazon.com/
cloudwatch/home?region=us-east-1#logEventViewer:group=/aws/iot/deviceadvisor/
qqcsmtyyjabl;stream=nzlfyhaa18oa_2ve2twrqyr0s",
                        "warnings": "null",
                        "failure": "null"
                    }
                ]
            }
        ]
    },
    "startTime": "2022-11-12T00:01:52.673000-05:00",
    "endTime": "2022-11-12T00:02:16.496000-05:00",
    "status": "PASS",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的获取测试套件运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSuiteRun。

`list-suite-definitions`

以下代码示例演示了如何使用 `list-suite-definitions`。

Amazon CLI

示例 1：列出您创建的 IoT Device Advisor 测试套件

以下 `list-suite-definitions` 示例列出您在 Amazon IoT 中创建的最多 25 个 Device Advisor 测试套件。如果您有超过 25 个测试套件，则输出中将显示“nextToken”。您可以使用这个“nextToken”来显示创建的其余测试套件。

```
aws iotdeviceadvisor list-suite-definitions
```

输出：

```
{
    "suiteDefinitionInformationList": [
        {
```

```
            "suiteDefinitionId": "3hsn88h4p2g5",
            "suiteDefinitionName": "TestSuite1",
            "defaultDevices": [
                {
                    "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"
                }
            ],
            "intendedForQualification": false,
            "isLongDurationTest": false,
            "protocol": "MqttV3_1_1",
            "createdAt": "2022-11-17T14:15:56.830000-05:00"
        },
        {
            ......
        }
    ],
    "nextToken": "nextTokenValue"
}
```

示例 2：列出您使用指定设置创建的 IoT Device Advisor 测试套件

以下 `list-suite-definitions` 示例列出您在 Amazon IoT 中创建的具有指定 max-result 数值的 Device Advisor 测试套件。如果您有超过此最大数值的测试套件，则输出中将显示“nextToken”。如果您有“nextToken”，则可以使用“nextToken”来显示您创建的在前面未显示的测试套件。

```
aws iotdeviceadvisor list-suite-definitions \
    --max-result 1 \
    --next-token "nextTokenValue"
```

输出：

```
{
    "suiteDefinitionInformationList": [
        {
            "suiteDefinitionId": "ztvb5aew4w4x",
            "suiteDefinitionName": "TestSuite2",
            "defaultDevices": [],
            "intendedForQualification": true,
            "isLongDurationTest": false,
            "protocol": "MqttV3_1_1",
```

```
            "createdAt": "2022-11-17T14:15:56.830000-05:00"
        }
    ],
    "nextToken": "nextTokenValue"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListSuiteDefinitions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSuiteDefinitions。

`list-suite-runs`

以下代码示例演示了如何使用 `list-suite-runs`。

Amazon CLI

示例 1：列出有关指定 IoT Device Advisor 测试套件运行状态的所有信息

以下 `list-suite-runs` 示例列出有关具有指定套件定义 ID 的 Device Advisor 测试套件运行状态的所有信息。如果您有超过 25 个测试套件运行，则输出中将显示“nextToken”。您可以使用这个“nextToken”来显示其余测试套件运行。

```
aws iotdeviceadvisor list-suite-runs \
    --suite-definition-id ztvb5aew4w4x
```

输出：

```
{
    "suiteRunsList": [
        {
            "suiteDefinitionId": "ztvb5aew4w4x",
            "suiteDefinitionVersion": "v1",
            "suiteDefinitionName": "TestSuite",
            "suiteRunId": "p6awv89nre6v",
            "createdAt": "2022-12-01T16:33:14.212000-05:00",
            "startedAt": "2022-12-01T16:33:15.710000-05:00",
            "endAt": "2022-12-01T16:42:03.323000-05:00",
            "status": "PASS",
            "passed": 6,
            "failed": 0
        }
    ]
```

```
}
```

示例 2：列出具有指定设置的指定 IoT Device Advisor 测试套件运行状态的信息

以下 `list-suite-runs` 示例列出具有指定套件定义 ID 和指定 max-result 数值的 Device Advisor 测试套件运行状态的信息。如果您有超过此最大数值的测试套件运行，则输出中将显示“nextToken”。如果您有“nextToken”，则可以使用“nextToken”来显示在前面未显示的测试套件运行。

```
aws iotdeviceadvisor list-suite-runs \
    --suite-definition-id qqcsmtyyjaml \
    --max-result 1 \
    --next-token "nextTokenValue"
```

输出：

```
{
    "suiteRunsList": [
        {
            "suiteDefinitionId": "qqcsmtyyjaml",
            "suiteDefinitionVersion": "v1",
            "suiteDefinitionName": "MQTT connection",
            "suiteRunId": "gz9vm2s6d2jy",
            "createdAt": "2022-12-01T20:10:27.079000-05:00",
            "startedAt": "2022-12-01T20:10:28.003000-05:00",
            "endAt": "2022-12-01T20:10:45.084000-05:00",
            "status": "STOPPED",
            "passed": 0,
            "failed": 0
        }
    ],
    "nextToken": "nextTokenValue"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListSuiteRuns。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSuiteRuns。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出附加到 IoT Device Advisor 资源的标签

以下 `list-tags-for-resource` 示例列出附加到 Device Advisor 资源的标签。Device Advisor 资源可以是 Suitedefinition-Arn 或 Suiterun-Arn。

```
aws iotdeviceadvisor list-tags-for-resource \
    --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/ba0uyjpg38ny
```

输出：

```
{
    "tags": {
        "TestTagKey": "TestTagValue"
    }
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 ListTagsForResource，或《服务授权参考》中的 Amazon IoT Core Device Advisor 定义的资源类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-suite-run

以下代码示例演示了如何使用 `start-suite-run`。

Amazon CLI

启动 IoT Device Advisor 测试套件运行

以下 `start-suite-run` 示例列出您的 Amazon 账户中可用的微件。

```
aws iotdeviceadvisor start-suite-run \
    --suite-definition-id qqcsmtyyjabl \
    --suite-definition-version v1 \
    --suite-run-configuration '{"primaryDevice":{"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/MyIotThing","certificateArn":"arn:aws:iot:us-east-1:123456789012:cert/certFile"}}'
```

输出：

```
{
    "suiteRunId": "pwmucgw7lt9s",
    "suiteRunArn": "arn:aws:iotdeviceadvisor:us-east-1:123456789012:suiterun/
qqcsmtyyjabl/pwmucgw7lk9s",
    "createdAt": "2022-12-02T15:43:05.581000-05:00"
}
```

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的启动测试套件运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSuiteRun。

`stop-suite-run`

以下代码示例演示了如何使用 `stop-suite-run`。

Amazon CLI

停止当前正在运行的 IoT Device Advisor 测试套件

以下 `stop-suite-run` 示例停止当前正在运行的具有指定套件定义 ID 和套件运行 ID 的 Device Advisor 测试套件。

```
aws iotdeviceadvisor stop-suite-run \
    --suite-definition-id qqcsmtyyjabl \
    --suite-run-id nzlfyhaa18oa
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Core 开发人员指南》中的停止测试套件运行。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopSuiteRun。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

添加和修改 IoT Device Advisor 资源的现有标签

以下 `tag-resource` 示例添加和修改具有指定资源 arn 和标签的 Device Advisor 资源的现有标签。Device Advisor 资源可以是 Suitedefinition-Arn 或 Suiterun-Arn。

```
aws iotdeviceadvisor tag-resource \
    --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/ba0uyjpg38ny \
    --tags '{"TagKey": "TagValue"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 TagResource，或《服务授权参考》中的 Amazon IoT Core Device Advisor 定义的资源类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从 IoT Device Advisor 资源中移除现有标签

以下 `untag-resource` 示例从具有指定资源 arn 和标签键的 Device Advisor 资源中移除现有标签。Device Advisor 资源可以是 Suitedefinition-Arn 或 Suiterun-Arn。

```
aws iotdeviceadvisor untag-resource \
    --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/ba0uyjpg38ny \
    --tag-keys "TagKey"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT API 参考》中的 UntagResource，或《服务授权参考》中的 Amazon IoT Core Device Advisor 定义的资源类型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

`update-suite-definition`

以下代码示例演示了如何使用 `update-suite-definition`。

Amazon CLI

示例 1：更新 IoT Device Advisor 测试套件

以下 update-suite-definition 示例更新 Amazon IoT 中的具有指定套件定义 ID 和套件定义配置的 Device Advisor 测试套件。

```
aws iotdeviceadvisor update-suite-definition \
    --suite-definition-id 3hsn88h4p2g5 \
    --suite-definition-configuration '{ \
        "suiteDefinitionName": "TestSuiteName", \
        "devices": [{"thingArn":"arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
        "intendedForQualification": false, \
        "rootGroup": "{\"configuration\":{},\"tests\":[{\"name\":\"MQTT Connect\",
\"configuration\":{\"EXECUTION_TIMEOUT\":120},\"tests\":[{\"name\":\"MQTT_Connect\",
\"configuration\":{},\"test\":{\"id\":\"MQTT_Connect\",\"testCase\":null,\"version
\":\"0.0.0\"}}]}]}", \
        "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"}'
```

输出：

```
{
    "suiteDefinitionId": "3hsn88h4p2g5",
    "suiteDefinitionName": "TestSuiteName",
    "suiteDefinitionVersion": "v3",
    "createdAt": "2022-11-17T14:15:56.830000-05:00",
    "lastUpdatedAt": "2022-12-02T16:02:45.857000-05:00"
}
```

示例 2：更新 IoT Device Advisor 合格测试套件

以下 update-suite-definition 示例更新 Amazon IoT 中的具有指定套件定义 ID 和套件定义配置的 Device Advisor 合格测试套件。

```
aws iotdeviceadvisor update-suite-definition \
    --suite-definition-id txgsuolk2myj \
    --suite-definition-configuration '{
        "suiteDefinitionName": "TestSuiteName", \
        "devices": [{"thingArn":"arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
        "intendedForQualification": true, \
```

```
        "rootGroup": "", \
        "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"}'
```

输出：

```
{
    "suiteDefinitionId": "txgsuolk2myj",
    "suiteDefinitionName": "TestSuiteName",
    "suiteDefinitionVersion": "v3",
    "createdAt": "2022-11-17T14:15:56.830000-05:00",
    "lastUpdatedAt": "2022-12-02T16:02:45.857000-05:00"
}
```

有关更多信息，请参阅《Amazon IoT API 参考》中的 UpdateSuiteDefinition。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSuiteDefinition。

使用 Amazon CLI 的 Amazon IoT data示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT data 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-thing-shadow`

以下代码示例演示了如何使用 `delete-thing-shadow`。

Amazon CLI

删除设备的影子文档

以下 delete-thing-shadow 示例删除名为 MyRPi 的设备的整个影子文档。

```
aws iot-data delete-thing-shadow \
    --thing-name MyRPi \
    "output.txt"
```

该命令不会在显示屏上生成任何输出，但 output.txt 中包含确认您删除的影子文档的版本和时间戳的信息。

```
{"version":2,"timestamp":1560270384}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的使用影子。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteThingShadow。

get-thing-shadow

以下代码示例演示了如何使用 get-thing-shadow。

Amazon CLI

获取事物影子文档

以下 get-thing-shadow 示例获取指定 IoT 事物的事物影子文档。

```
aws iot-data get-thing-shadow \
    --thing-name MyRPi \
    output.txt
```

该命令不会在显示屏上生成任何输出，但以下显示了 output.txt 的内容：

```
{
  "state":{
    "reported":{
    "moisture":"low"
    }
  },
  "metadata":{
    "reported":{
      "moisture":{
```

```
        "timestamp":1560269319
      }
    }
  },
  "version":1,"timestamp":1560269405
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备影子服务数据流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetThingShadow。

update-thing-shadow

以下代码示例演示了如何使用 update-thing-shadow。

Amazon CLI

更新事物影子

以下 update-thing-shadow 示例修改了指定事物的设备影子的当前状态并将其保存到文件 output.txt 中。

```
aws iot-data update-thing-shadow \
    --thing-name MyRPi \
    --payload "{"state":{"reported":{"moisture":"okay"}}}" \
    "output.txt"
```

该命令不会在显示屏上生成任何输出，但以下显示了 output.txt 的内容：

```
{
    "state": {
        "reported": {
            "moisture": "okay"
        }
    },
    "metadata": {
        "reported": {
            "moisture": {
                "timestamp": 1560270036
            }
        }
    },
```

```
    "version": 2,
    "timestamp": 1560270036
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备影子服务数据流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateThingShadow。

使用 Amazon CLI 的 Amazon IoT Events示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Events 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-put-message`

以下代码示例演示了如何使用 `batch-put-message`。

Amazon CLI

向 Amazon IoT Events 发送消息（输入）

以下 `batch-put-message` 示例向 Amazon IoT Events 系统发送一组消息。每个消息有效载荷都将转换为您指定的输入 (`inputName`)，并提取到任何监视该输入的检测器中。如果发送多条消息，不能保证按顺序处理消息。为确保按序处理，您必须一次发送一条消息，等收到成功回应后，再发送下一条。

```
aws iotevents-data batch-put-message \
    --cli-input-json file://highPressureMessage.json
```

的内容highPressureMessage.json：

```
{
    "messages": [
        {
            "messageId": "00001",
            "inputName": "PressureInput",
            "payload": "{\"motorid\": \"Fulton-A32\", \"sensorData\": {\"pressure\":
 80, \"temperature\": 39} }"
        }
    ]
}
```

输出：

```
{
    "BatchPutMessageErrorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 BatchPutMessage。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchPutMessage。

batch-update-detector

以下代码示例演示了如何使用 batch-update-detector。

Amazon CLI

更新检测器（实例）

以下 batch-update-detector 示例更新指定检测器模型的一个或多个检测器（实例）的状态、变量值和计时器设置。

```
aws iotevents-data batch-update-detector \
    --cli-input-json file://budFulton-A32.json
```

的内容budFulton-A32.json：

```
{
```

```
    "detectors": [
        {
            "messageId": "00001",
            "detectorModelName": "motorDetectorModel",
            "keyValue": "Fulton-A32",
            "state": {
                "stateName": "Normal",
                "variables": [
                    {
                        "name": "pressureThresholdBreached",
                        "value": "0"
                    }
                ],
                "timers": [
                ]
            }
        }
    ]
}
```

输出：

```
{
    "batchUpdateDetectorErrorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 BatchUpdateDetector。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateDetector。

create-detector-model

以下代码示例演示了如何使用 create-detector-model。

Amazon CLI

创建检测器模型

以下 create-detector-model 示例创建了一个检测器模型，其配置由参数文件指定。

```
aws iotevents create-detector-model  \
    --cli-input-json file://motorDetectorModel.json
```

的内容motorDetectorModel.json：

```
{
    "detectorModelName": "motorDetectorModel",
    "detectorModelDefinition": {
        "states": [
            {
                "stateName": "Normal",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "init",
                            "condition": "true",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "0"
                                    }
                                }
                            ]
                        }
                    ]
                },
                "onInput": {
                    "transitionEvents": [
                        {
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure
 &gt; 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
 "$variable.pressureThresholdBreached + 3"
                                    }
                                }
                            ],
                            "nextState": "Dangerous"
                        }
                    ]
                }
            },
```

```
            {
                "stateName": "Dangerous",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "Pressure Threshold Breached",
                            "condition": "$variable.pressureThresholdBreached &gt;
 1",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                                    }
                                }
                            ]
                        }
                    ]
                },
                "onInput": {
                    "events": [
                        {
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure
 &gt; 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "3"
                                    }
                                }
                            ]
                        },
                        {
                            "eventName": "Pressure Okay",
                            "condition": "$input.PressureInput.sensorData.pressure
 &lt;= 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
 "$variable.pressureThresholdBreached - 1"
```

```
                                    }
                                }
                            ]
                        }
                    ],
                    "transitionEvents": [
                        {
                            "eventName": "BackToNormal",
                            "condition": "$input.PressureInput.sensorData.pressure
 &lt;= 70 &amp;&amp; $variable.pressureThresholdBreached &lt;= 1",
                            "nextState": "Normal"
                        }
                    ]
                },
                "onExit": {
                    "events": [
                        {
                            "eventName": "Normal Pressure Restored",
                            "condition": "true",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                                    }
                                }
                            ]
                        }
                    ]
                }
            }
        ],
        "initialStateName": "Normal"
    },
    "key": "motorid",
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

输出：

```
{
    "detectorModelConfiguration": {
        "status": "ACTIVATING",
```

```
        "lastUpdateTime": 1560796816.077,
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "creationTime": 1560796816.077,
        "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
        "key": "motorid",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "1"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 CreateDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDetectorModel。

create-input

以下代码示例演示了如何使用 create-input。

Amazon CLI

创建输入

以下 create-input 示例创建了一个输入。

```
aws iotevents create-input  \
    --cli-input-json file://pressureInput.json
```

的内容pressureInput.json：

```
{
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor",
    "inputDefinition": {
        "attributes": [
            { "jsonPath": "sensorData.pressure" },
            { "jsonPath": "motorid" }
        ]
    }
}
```

输出：

```
{
    "inputConfiguration": {
        "status": "ACTIVE",
        "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
        "lastUpdateTime": 1560795312.542,
        "creationTime": 1560795312.542,
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 CreateInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInput。

delete-detector-model

以下代码示例演示了如何使用 delete-detector-model。

Amazon CLI

删除检测器模型

以下 delete-detector-model 示例删除了指定的检测器模型。该检测器模型的所有活动实例也被删除。

```
aws iotevents delete-detector-model \
    --detector-model-name motorDetectorModel
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DeleteDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDetectorModel。

delete-input

以下代码示例演示了如何使用 delete-input。

Amazon CLI

删除输入

以下 delete-input 示例删除了指定输入。

```
aws iotevents delete-input \
    --input-name PressureInput
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DeleteInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInput。

describe-detector-model

以下代码示例演示了如何使用 describe-detector-model。

Amazon CLI

获取有关检测器模型的信息

以下 describe-detector-model 示例显示指定检测器模型的详细信息。由于未指定 version 参数，所以返回的是有关最新版本的信息。

```
aws iotevents describe-detector-model \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorModel": {
        "detectorModelConfiguration": {
            "status": "ACTIVE",
            "lastUpdateTime": 1560796816.077,
            "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
            "creationTime": 1560796816.077,
            "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
            "key": "motorid",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        },
        "detectorModelDefinition": {
            "states": [
                {
```

```
                    "onInput": {
                        "transitionEvents": [
                            {
                                "eventName": "Overpressurized",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
                                            "value":
"$variable.pressureThresholdBreached + 3"
                                        }
                                    }
                                ],
                                "condition":
"$input.PressureInput.sensorData.pressure > 70",
                                "nextState": "Dangerous"
                            }
                        ],
                        "events": []
                    },
                    "stateName": "Normal",
                    "onEnter": {
                        "events": [
                            {
                                "eventName": "init",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
                                            "value": "0"
                                        }
                                    }
                                ],
                                "condition": "true"
                            }
                        ]
                    },
                    "onExit": {
                        "events": []
                    }
                },
                {
```

```
                    "onInput": {
                        "transitionEvents": [
                            {
                                "eventName": "BackToNormal",
                                "actions": [],
                                "condition":
"$input.PressureInput.sensorData.pressure <= 70 &&
$variable.pressureThresholdBreached <= 1",
                                "nextState": "Normal"
                            }
                        ],
                        "events": [
                            {
                                "eventName": "Overpressurized",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
                                            "value": "3"
                                        }
                                    }
                                ],
                                "condition":
"$input.PressureInput.sensorData.pressure > 70"
                            },
                            {
                                "eventName": "Pressure Okay",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
                                            "value":
"$variable.pressureThresholdBreached - 1"
                                        }
                                    }
                                ],
                                "condition":
"$input.PressureInput.sensorData.pressure <= 70"
                            }
                        ]
                    },
                    "stateName": "Dangerous",
```

```
                    "onEnter": {
                        "events": [
                            {
                                "eventName": "Pressure Threshold Breached",
                                "actions": [
                                    {
                                        "sns": {
                                            "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                                        }
                                    }
                                ],
                                "condition": "$variable.pressureThresholdBreached >
 1"
                            }
                        ]
                    },
                    "onExit": {
                        "events": [
                            {
                                "eventName": "Normal Pressure Restored",
                                "actions": [
                                    {
                                        "sns": {
                                            "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                                        }
                                    }
                                ],
                                "condition": "true"
                            }
                        ]
                    }
                }
            ],
            "initialStateName": "Normal"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DescribeDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDetectorModel。

describe-detector

以下代码示例演示了如何使用 describe-detector。

Amazon CLI

获取有关检测器（实例）的信息

以下 describe-detector 示例显示指定检测器（实例）的详细信息。

```
aws iotevents-data describe-detector \
    --detector-model-name motorDetectorModel \
    --key-value "Fulton-A32"
```

输出：

```
{
    "detector": {
        "lastUpdateTime": 1560797852.776,
        "creationTime": 1560797852.775,
        "state": {
            "variables": [
                {
                    "name": "pressureThresholdBreached",
                    "value": "3"
                }
            ],
            "stateName": "Dangerous",
            "timers": []
        },
        "keyValue": "Fulton-A32",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "1"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DescribeDetector。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDetector。

describe-input

以下代码示例演示了如何使用 describe-input。

Amazon CLI

获取有关输入的信息

以下 describe-input 示例显示指定输入的详细信息。

```
aws iotevents describe-input \
    --input-name PressureInput
```

输出：

```
{
    "input": {
        "inputConfiguration": {
            "status": "ACTIVE",
            "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/
PressureInput",
            "lastUpdateTime": 1560795312.542,
            "creationTime": 1560795312.542,
            "inputName": "PressureInput",
            "inputDescription": "Pressure readings from a motor"
        },
        "inputDefinition": {
            "attributes": [
                {
                    "jsonPath": "sensorData.pressure"
                },
                {
                    "jsonPath": "motorid"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DescribeInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInput。

describe-logging-options

以下代码示例演示了如何使用 describe-logging-options。

Amazon CLI

获取有关日志设置的信息

以下 `describe-logging-options` 示例检索 Amazon IoT Events 日志记录选项的当前设置。

```
aws iotevents describe-logging-options
```

输出：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "enabled": false,
        "level": "ERROR"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 DescribeLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoggingOptions。

list-detector-model-versions

以下代码示例演示了如何使用 `list-detector-model-versions`。

Amazon CLI

获取有关检测器模型版本的信息

以下 `list-detector-model-versions` 示例列出某个检测器模型的所有版本。仅返回与每个检测器模型版本关联的元数据。

```
aws iotevents list-detector-model-versions \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorModelVersionSummaries": [
        {
            "status": "ACTIVE",
```

```
            "lastUpdateTime": 1560796816.077,
            "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
            "creationTime": 1560796816.077,
            "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 ListDetectorModelVersions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectorModelVersions。

`list-detector-models`

以下代码示例演示了如何使用 `list-detector-models`。

Amazon CLI

获取您的检测器模型列表

以下 `list-detector-models` 示例列出您创建的检测器模型。仅返回与每个检测器模型关联的元数据。

```
aws iotevents list-detector-models
```

输出：

```
{
    "detectorModelSummaries": [
        {
            "detectorModelName": "motorDetectorModel",
            "creationTime": 1552072424.212
            "detectorModelDescription": "Detect overpressure in a motor."
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 ListDetectorModels。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectorModels。

`list-detectors`

以下代码示例演示了如何使用 `list-detectors`。

Amazon CLI

获取检测器模型的检测器列表

以下 `list-detectors` 示例列出了您的账户中的检测器（检测器模型的实例）。

```
aws iotevents-data list-detectors \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorSummaries": [
        {
            "lastUpdateTime": 1558129925.2,
            "creationTime": 1552073155.527,
            "state": {
                "stateName": "Normal"
            },
            "keyValue": "Fulton-A32",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 ListDetectors。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectors。

`list-inputs`

以下代码示例演示了如何使用 `list-inputs`。

Amazon CLI

列出输入

以下 `list-inputs` 示例列出您在自己的账户中创建的输入。

```
aws iotevents list-inputs
```

此命令不生成任何输出。输出：

```
{
    {
        "status": "ACTIVE",
        "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
        "lastUpdateTime": 1551742986.768,
        "creationTime": 1551742986.768,
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 ListInputs。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInputs。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出分配给资源的标签

以下 list-tags-for-resource 示例列出您分配给资源的标签键名称和值。

```
aws iotevents list-tags-for-resource \
    --resource-arn "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput"
```

输出：

```
{
    "tags": [
        {
            "value": "motor",
            "key": "deviceType"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 ListTagsForResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-logging-options

以下代码示例演示了如何使用 put-logging-options。

Amazon CLI

设置日志选项

以下 put-logging-options 示例设置或更新 Amazon IoT Events 日志选项。如果您更新了任何 loggingOptions` field, it can take up to one minute for the change to take effect. Also, if you change the policy attached to the role you specified in the ``roleArn 字段的值（例如，更正无效策略），则最多需要五分钟，更改就能生效。

```
aws iotevents put-logging-options \
    --cli-input-json file://logging-options.json
```

的内容logging-options.json：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "level": "DEBUG",
        "enabled": true,
        "detectorDebugOptions": [
            {
                "detectorModelName": "motorDetectorModel",
                "keyValue": "Fulton-A32"
            }
        ]
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 PutLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingOptions。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源添加标签

以下 tag-resource 示例添加或修改（如果键 deviceType 已存在）附加到指定资源的标签。

```
aws iotevents tag-resource \
    --cli-input-json file://pressureInput.tag.json
```

的内容pressureInput.tag.json：

```
{
    "resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "tags": [
        {
            "key": "deviceType",
            "value": "motor"
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 TagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从指定资源中移除具有指定键名的标签。

```
aws iotevents untag-resource \
    --resource-arn arn:aws:iotevents:us-west-2:123456789012:input/PressureInput \
```

```
    --tagkeys deviceType
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-detector-model

以下代码示例演示了如何使用 update-detector-model。

Amazon CLI

更新检测器模型

以下 update-detector-model 示例更新了指定的检测器模型。先前版本生成的检测器（实例）会被删除，然后在新输入到来时重新创建。

```
aws iotevents update-detector-model \
    --cli-input-json file://motorDetectorModel.update.json
```

的内容motorDetectorModel.update.json：

```
{
    "detectorModelName": "motorDetectorModel",
    "detectorModelDefinition": {
        "states": [
            {
                "stateName": "Normal",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "init",
                            "condition": "true",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "0"
                                    }
                                }
                            ]
```

```
                        }
                    ]
                },
                "onInput": {
                    "transitionEvents": [
                        {
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure >
 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
 "$variable.pressureThresholdBreached + 3"
                                    }
                                }
                            ],
                            "nextState": "Dangerous"
                        }
                    ]
                }
            },
            {
                "stateName": "Dangerous",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "Pressure Threshold Breached",
                            "condition": "$variable.pressureThresholdBreached > 1",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                                    }
                                }
                            ]
                        }
                    ]
                },
                "onInput": {
                    "events": [
                        {
```

```
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure >
70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "3"
                                    }
                                }
                            ]
                        },
                        {
                            "eventName": "Pressure Okay",
                            "condition": "$input.PressureInput.sensorData.pressure
<= 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
"$variable.pressureThresholdBreached - 1"
                                    }
                                }
                            ]
                        }
                    ],
                    "transitionEvents": [
                        {
                            "eventName": "BackToNormal",
                            "condition": "$input.PressureInput.sensorData.pressure
<= 70 && $variable.pressureThresholdBreached <= 1",
                            "nextState": "Normal"
                        }
                    ]
                },
                "onExit": {
                    "events": [
                        {
                            "eventName": "Normal Pressure Restored",
                            "condition": "true",
                            "actions": [
                                {
                                    "sns": {
```

```
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                                    }
                                }
                            ]
                        }
                    ]
                }
            }
        ],
        "initialStateName": "Normal"
    },
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

输出：

```
{
    "detectorModelConfiguration": {
        "status": "ACTIVATING",
        "lastUpdateTime": 1560799387.719,
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "creationTime": 1560799387.719,
        "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
        "key": "motorid",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "2"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 UpdateDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDetectorModel。

update-input

以下代码示例演示了如何使用 update-input。

Amazon CLI

更新输入

以下 update-input 示例使用新的描述和定义更新了指定的输入。

```
aws iotevents update-input \
    --cli-input-json file://pressureInput.json
```

的内容pressureInput.json：

```
{
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor",
    "inputDefinition": {
        "attributes": [
            { "jsonPath": "sensorData.pressure" },
            { "jsonPath": "motorid" }
        ]
    }
}
```

输出：

```
{
    "inputConfiguration": {
        "status": "ACTIVE",
        "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
        "lastUpdateTime": 1560795976.458,
        "creationTime": 1560795312.542,
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events API 参考》中的 UpdateInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateInput。

使用 Amazon CLI 的 Amazon IoT Events-Data示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Events-Data 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-put-message

以下代码示例演示了如何使用 batch-put-message。

Amazon CLI

向 Amazon IoT Events 发送消息（输入）

以下 batch-put-message 示例向 Amazon IoT Events 系统发送一组消息。每个消息有效载荷都将转换为您指定的输入 (inputName)，并提取到任何监视该输入的检测器中。如果发送多条消息，不能保证按顺序处理消息。为确保按序处理，您必须一次发送一条消息，等收到成功回应后，再发送下一条。

```
aws iotevents-data batch-put-message \
    --cli-binary-format raw-in-base64-out \
    --cli-input-json file://highPressureMessage.json
```

的内容highPressureMessage.json：

```
{
    "messages": [
        {
            "messageId": "00001",
            "inputName": "PressureInput",
            "payload": "{\"motorid\": \"Fulton-A32\", \"sensorData\": {\"pressure\":
 80, \"temperature\": 39} }"
        }
    ]
}
```

输出：

```
{
```

```
    "BatchPutMessageErrorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 BatchPutMessage。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchPutMessage。

batch-update-detector

以下代码示例演示了如何使用 batch-update-detector。

Amazon CLI

更新检测器（实例）

以下 batch-update-detector 示例更新指定检测器模型的一个或多个检测器（实例）的状态、变量值和计时器设置。

```
aws iotevents-data batch-update-detector \
    --cli-input-json file://budFulton-A32.json
```

的内容budFulton-A32.json：

```
{
    "detectors": [
        {
            "messageId": "00001",
            "detectorModelName": "motorDetectorModel",
            "keyValue": "Fulton-A32",
            "state": {
                "stateName": "Normal",
                "variables": [
                    {
                        "name": "pressureThresholdBreached",
                        "value": "0"
                    }
                ],
                "timers": [
                ]
            }
        }
    ]
```

```
}
```

输出：

```
{
    "batchUpdateDetectorErrorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 BatchUpdateDetector。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateDetector。

create-detector-model

以下代码示例演示了如何使用 create-detector-model。

Amazon CLI

创建检测器模型

以下 create-detector-model 示例创建了一个检测器模型。

```
aws iotevents create-detector-model \
    --cli-input-json file://motorDetectorModel.json
```

的内容motorDetectorModel.json：

```
{
    "detectorModelName": "motorDetectorModel",
    "detectorModelDefinition": {
        "states": [
            {
                "stateName": "Normal",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "init",
                            "condition": "true",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "0"
```

```
                                    }
                                }
                            ]
                        }
                    ]
                },
                "onInput": {
                    "transitionEvents": [
                        {
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure
 &gt; 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
 "$variable.pressureThresholdBreached + 3"
                                    }
                                }
                            ],
                            "nextState": "Dangerous"
                        }
                    ]
                }
            },
            {
                "stateName": "Dangerous",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "Pressure Threshold Breached",
                            "condition": "$variable.pressureThresholdBreached &gt;
 1",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                                    }
                                }
                            ]
                        }
                    ]
```

```
                },
                "onInput": {
                    "events": [
                        {
                            "eventName": "Overpressurized",
                            "condition": "$input.PressureInput.sensorData.pressure
&gt; 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value": "3"
                                    }
                                }
                            ]
                        },
                        {
                            "eventName": "Pressure Okay",
                            "condition": "$input.PressureInput.sensorData.pressure
&lt;= 70",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName": "pressureThresholdBreached",
                                        "value":
"$variable.pressureThresholdBreached - 1"
                                    }
                                }
                            ]
                        }
                    ],
                    "transitionEvents": [
                        {
                            "eventName": "BackToNormal",
                            "condition": "$input.PressureInput.sensorData.pressure
&lt;= 70 &amp;&amp; $variable.pressureThresholdBreached &lt;= 1",
                            "nextState": "Normal"
                        }
                    ]
                },
                "onExit": {
                    "events": [
                        {
                            "eventName": "Normal Pressure Restored",
```

```
                            "condition": "true",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                                    }
                                }
                            ]
                        }
                    ]
                }
            }
        ],
        "initialStateName": "Normal"
    },
    "key": "motorid",
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

输出：

```
{
    "detectorModelConfiguration": {
        "status": "ACTIVATING",
        "lastUpdateTime": 1560796816.077,
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "creationTime": 1560796816.077,
        "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
        "key": "motorid",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "1"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 CreateDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDetectorModel。

create-input

以下代码示例演示了如何使用 create-input。

Amazon CLI

创建输入

以下 create-input 示例创建了一个输入。

```
aws iotevents create-input \
    --cli-input-json file://pressureInput.json
```

的内容pressureInput.json：

```
{
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor",
    "inputDefinition": {
        "attributes": [
                { "jsonPath": "sensorData.pressure" },
                { "jsonPath": "motorid" }
            ]
    }
}
```

输出：

```
{
    "inputConfiguration": {
        "status": "ACTIVE",
        "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
        "lastUpdateTime": 1560795312.542,
        "creationTime": 1560795312.542,
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 CreateInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInput。

delete-detector-model

以下代码示例演示了如何使用 delete-detector-model。

Amazon CLI

删除检测器模型

以下 delete-detector-model 示例删除了一个检测器模型。该检测器模型的所有活动实例也被删除。

```
aws iotevents delete-detector-model \
    --detector-model-name motorDetectorModel*
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DeleteDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDetectorModel。

delete-input

以下代码示例演示了如何使用 delete-input。

Amazon CLI

删除输入

以下 delete-input 示例删除了一个输入。

```
aws iotevents delete-input \
    --input-name PressureInput
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DeleteInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInput。

describe-detector-model

以下代码示例演示了如何使用 describe-detector-model。

Amazon CLI

获取有关检测器模型的信息

以下 describe-detector-model 示例描述了一个检测器模型。如果未指定 version 参数，该命令将返回有关最新版本的信息。

```
aws iotevents describe-detector-model \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorModel": {
        "detectorModelConfiguration": {
            "status": "ACTIVE",
            "lastUpdateTime": 1560796816.077,
            "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
            "creationTime": 1560796816.077,
            "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
            "key": "motorid",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        },
        "detectorModelDefinition": {
            "states": [
                {
                    "onInput": {
                        "transitionEvents": [
                            {
                                "eventName": "Overpressurized",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
 "pressureThresholdBreached",
                                            "value":
 "$variable.pressureThresholdBreached + 3"
                                        }
                                    }
                                ],
                                "condition":
 "$input.PressureInput.sensorData.pressure > 70",
                                "nextState": "Dangerous"
                            }
                        ],
```

```
                        "events": []
                    },
                    "stateName": "Normal",
                    "onEnter": {
                        "events": [
                            {
                                "eventName": "init",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
                                            "value": "0"
                                        }
                                    }
                                ],
                                "condition": "true"
                            }
                        ]
                    },
                    "onExit": {
                        "events": []
                    }
                },
                {
                    "onInput": {
                        "transitionEvents": [
                            {
                                "eventName": "BackToNormal",
                                "actions": [],
                                "condition":
"$input.PressureInput.sensorData.pressure <= 70 &&
$variable.pressureThresholdBreached <= 1",
                                "nextState": "Normal"
                            }
                        ],
                        "events": [
                            {
                                "eventName": "Overpressurized",
                                "actions": [
                                    {
                                        "setVariable": {
                                            "variableName":
"pressureThresholdBreached",
```

```
                                        "value": "3"
                                    }
                                }
                            ],
                            "condition":
 "$input.PressureInput.sensorData.pressure > 70"
                        },
                        {
                            "eventName": "Pressure Okay",
                            "actions": [
                                {
                                    "setVariable": {
                                        "variableName":
 "pressureThresholdBreached",
                                        "value":
 "$variable.pressureThresholdBreached - 1"
                                    }
                                }
                            ],
                            "condition":
 "$input.PressureInput.sensorData.pressure <= 70"
                        }
                    ]
                },
                "stateName": "Dangerous",
                "onEnter": {
                    "events": [
                        {
                            "eventName": "Pressure Threshold Breached",
                            "actions": [
                                {
                                    "sns": {
                                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                                    }
                                }
                            ],
                            "condition": "$variable.pressureThresholdBreached >
 1"
                        }
                    ]
                },
                "onExit": {
                    "events": [
```

```
                            {
                                "eventName": "Normal Pressure Restored",
                                "actions": [
                                    {
                                        "sns": {
                                            "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                                        }
                                    }
                                ],
                                "condition": "true"
                            }
                        ]
                    }
                }
            ],
            "initialStateName": "Normal"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DescribeDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDetectorModel。

describe-detector

以下代码示例演示了如何使用 describe-detector。

Amazon CLI

获取有关检测器（实例）的信息

以下 describe-detector 示例返回有关指定检测器（实例）的信息。

```
aws iotevents-data describe-detector \
    --detector-model-name motorDetectorModel \
    --key-value "Fulton-A32"
```

输出：

```
{
    "detector": {
```

```
        "lastUpdateTime": 1560797852.776,
        "creationTime": 1560797852.775,
        "state": {
            "variables": [
                {
                    "name": "pressureThresholdBreached",
                    "value": "3"
                }
            ],
            "stateName": "Dangerous",
            "timers": []
        },
        "keyValue": "Fulton-A32",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "1"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DescribeDetector。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDetector。

describe-input

以下代码示例演示了如何使用 describe-input。

Amazon CLI

获取有关输入的信息

以下 describe-input 示例检索某个输入的详细信息。

```
aws iotevents describe-input \
    --input-name PressureInput
```

输出：

```
{
    "input": {
        "inputConfiguration": {
            "status": "ACTIVE",
            "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/
PressureInput",
```

```
            "lastUpdateTime": 1560795312.542,
            "creationTime": 1560795312.542,
            "inputName": "PressureInput",
            "inputDescription": "Pressure readings from a motor"
        },
        "inputDefinition": {
            "attributes": [
                {
                    "jsonPath": "sensorData.pressure"
                },
                {
                    "jsonPath": "motorid"
                }
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DescribeInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInput。

describe-logging-options

以下代码示例演示了如何使用 describe-logging-options。

Amazon CLI

获取有关日志设置的信息

以下 describe-logging-options 示例检索当前 Amazon IoT Events 日志选项。

```
aws iotevents describe-logging-options
```

输出：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "enabled": false,
        "level": "ERROR"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 DescribeLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoggingOptions。

list-detector-model-versions

以下代码示例演示了如何使用 list-detector-model-versions。

Amazon CLI

获取有关检测器模型版本的信息

以下 list-detector-model-versions 示例列出某个检测器模型的所有版本。仅返回与每个检测器模型版本关联的元数据。

```
aws iotevents list-detector-model-versions \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorModelVersionSummaries": [
        {
            "status": "ACTIVE",
            "lastUpdateTime": 1560796816.077,
            "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
            "creationTime": 1560796816.077,
            "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 ListDetectorModelVersions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectorModelVersions。

list-detector-models

以下代码示例演示了如何使用 list-detector-models。

Amazon CLI

获取您的检测器模型列表

以下 `list-detector-models` 示例列出您创建的检测器模型。仅返回与每个检测器模型关联的元数据。

```
aws iotevents list-detector-models
```

输出：

```
{
    "detectorModelSummaries": [
        {
            "detectorModelName": "motorDetectorModel",
            "creationTime": 1552072424.212
            "detectorModelDescription": "Detect overpressure in a motor."
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 ListDetectorModels。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectorModels。

`list-detectors`

以下代码示例演示了如何使用 `list-detectors`。

Amazon CLI

获取检测器模型的检测器列表

以下 `list-detectors` 示例列出检测器（某个检测器模型的实例）。

```
aws iotevents-data list-detectors \
    --detector-model-name motorDetectorModel
```

输出：

```
{
    "detectorSummaries": [
```

```
        {
            "lastUpdateTime": 1558129925.2,
            "creationTime": 1552073155.527,
            "state": {
                "stateName": "Normal"
            },
            "keyValue": "Fulton-A32",
            "detectorModelName": "motorDetectorModel",
            "detectorModelVersion": "1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 ListDetectors。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDetectors。

`list-inputs`

以下代码示例演示了如何使用 `list-inputs`。

Amazon CLI

列出输入

以下 `list-inputs` 示例列出您创建的输入。

```
aws iotevents list-inputs
```

输出：

```
{
    "status": "ACTIVE",
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "lastUpdateTime": 1551742986.768,
    "creationTime": 1551742986.768,
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor"
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 ListInputs。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInputs。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出分配给资源的标签

以下 list-tags-for-resource 示例列出您分配给资源的标签（元数据）。

```
aws iotevents list-tags-for-resource \
    --resource-arn "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput"
```

输出：

```
{
    "tags": [
        {
            "value": "motor",
            "key": "deviceType"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 ListTagsForResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-logging-options

以下代码示例演示了如何使用 put-logging-options。

Amazon CLI

设置日志选项

以下 list-tags-for-resource 示例设置或更新 Amazon IoT Events 日志选项。如果您更新了任何 loggingOptions 字段的值，则最多需要一分钟，更改就能生效。此外，如果您更改了附加到您在 roleArn 字段中所指定角色的策略（例如，更正无效策略），则最多需要 5 分钟，更改就能生效。

```
aws iotevents put-logging-options \
```

```
    --cli-input-json file://logging-options.json
```

的内容logging-options.json：

```
{
    "loggingOptions": {
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "level": "DEBUG",
        "enabled": true,
        "detectorDebugOptions": [
            {
                "detectorModelName": "motorDetectorModel",
                "keyValue": "Fulton-A32"
            }
        ]
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 PutLoggingOptions。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingOptions。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源添加标签

以下 tag-resource 示例添加或修改给定资源的标签。标签是可用于管理资源的元数据。

```
aws iotevents tag-resource \
    --cli-input-json file://pressureInput.tag.json
```

的内容pressureInput.tag.json：

```
{
    "resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "tags": [
        {
```

```
            "key": "deviceType",
            "value": "motor"
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 TagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从资源中移除指定标签。

```
aws iotevents untag-resource \
    --cli-input-json file://pressureInput.untag.json
```

的内容pressureInput.untag.json：

```
{
    "resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "tagKeys": [
            "deviceType"
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-detector-model

以下代码示例演示了如何使用 update-detector-model。

Amazon CLI

更新检测器模型

以下 update-detector-model 示例更新一个检测器模型。先前版本生成的检测器（实例）会被删除，然后在新输入到来时重新创建。

```
aws iotevents update-detector-model \
    --cli-input-json file://motorDetectorModel.update.json
```

motorDetectorModel.update.json 的内容：

```
{
  "detectorModelName": "motorDetectorModel",
  "detectorModelDefinition": {
    "states": [
      {
        "stateName": "Normal",
        "onEnter": {
          "events": [
            {
              "eventName": "init",
              "condition": "true",
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "0"
                  }
                }
              ]
            }
          ]
        },
        "onInput": {
          "transitionEvents": [
            {
              "eventName": "Overpressurized",
              "condition": "$input.PressureInput.sensorData.pressure > 70",
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreached",
```

```
                    "value": "$variable.pressureThresholdBreached + 3"
                  }
                }
              ],
              "nextState": "Dangerous"
            }
          ]
        }
      },
      {
        "stateName": "Dangerous",
        "onEnter": {
          "events": [
            {
              "eventName": "Pressure Threshold Breached",
              "condition": "$variable.pressureThresholdBreached > 1",
              "actions": [
                {
                  "sns": {
                    "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                  }
                }
              ]
            }
          ]
        },
        "onInput": {
          "events": [
            {
              "eventName": "Overpressurized",
              "condition": "$input.PressureInput.sensorData.pressure > 70",
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "3"
                  }
                }
              ]
            },
            {
              "eventName": "Pressure Okay",
              "condition": "$input.PressureInput.sensorData.pressure <= 70",
```

```
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "$variable.pressureThresholdBreached - 1"
                  }
                }
              ]
            }
          ],
          "transitionEvents": [
            {
              "eventName": "BackToNormal",
              "condition": "$input.PressureInput.sensorData.pressure <= 70 &&
 $variable.pressureThresholdBreached <= 1",
              "nextState": "Normal"
            }
          ]
        },
        "onExit": {
          "events": [
            {
              "eventName": "Normal Pressure Restored",
              "condition": "true",
              "actions": [
                {
                  "sns": {
                    "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                  }
                }
              ]
            }
          ]
        }
      }
    ],
    "initialStateName": "Normal"
  },
  "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

输出：

```
{
    "detectorModelConfiguration": {
        "status": "ACTIVATING",
        "lastUpdateTime": 1560799387.719,
        "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
        "creationTime": 1560799387.719,
        "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
        "key": "motorid",
        "detectorModelName": "motorDetectorModel",
        "detectorModelVersion": "2"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 UpdateDetectorModel。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDetectorModel。

update-input

以下代码示例演示了如何使用 update-input。

Amazon CLI

更新输入

以下 update-input 示例更新了一个输入。

```
aws iotevents update-input \
    --cli-input-json file://pressureInput.json
```

的内容pressureInput.json：

```
{
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor",
    "inputDefinition": {
        "attributes": [
            { "jsonPath": "sensorData.pressure" },
            { "jsonPath": "motorid" }
        ]
```

```
    }
}
```

输出：

```
{
    "inputConfiguration": {
        "status": "ACTIVE",
        "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
        "lastUpdateTime": 1560795976.458,
        "creationTime": 1560795312.542,
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    }
}
```

有关更多信息，请参阅《Amazon IoT Events 开发人员指南》*中的 UpdateInput。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateInput。

Amazon IoT Greengrass使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Greengrass 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-role-to-group`

以下代码示例演示了如何使用 `associate-role-to-group`。

Amazon CLI

将角色与 Greengrass 组相关联

以下 `associate-role-to-group` 示例将指定的 IAM 角色与 Greengrass 组相关联。本地 Lambda 函数和连接器使用该组角色来访问 Amazon 服务。例如，您的组角色可能会授予 CloudWatch Logs 集成所需的权限。

```
aws greengrass associate-role-to-group \
    --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b \
    --role-arn arn:aws:iam::123456789012:role/GG-Group-Role
```

输出：

```
{
    "AssociatedAt": "2019-09-10T20:03:30Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置组角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateRoleToGroup。

associate-service-role-to-account

以下代码示例演示了如何使用 `associate-service-role-to-account`。

Amazon CLI

将服务角色与 Amazon 账户相关联

以下 `associate-service-role-to-account` 示例将 IAM 服务角色（由其 ARN 指定）与 Amazon 账户中的 Amazon IoT Greengrass 相关联。您之前必须在 IAM 中创建服务角色，并且必须将允许 Amazon IoT Greengrass 代入此角色的策略文档与该角色相关联。

```
aws greengrass associate-service-role-to-account \
    --role-arn "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
```

输出：

```
{
```

```
    "AssociatedAt": "2019-06-25T18:12:45Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateServiceRoleToAccount。

`create-connector-definition-version`

以下代码示例演示了如何使用 `create-connector-definition-version`。

Amazon CLI

创建连接器定义版本

以下 `create-connector-definition-version` 示例创建一个连接器定义版本，并将其与指定的连接器定义相关联。版本中的所有连接器均定义其参数的值。

```
aws greengrass create-connector-definition-version \
    --connector-definition-id "55d0052b-0d7d-44d6-b56f-21867215e118" \
    --connectors "[{\"Id\": \"MyTwilioNotificationsConnector\",
 \"ConnectorArn\": \"arn:aws:greengrass:us-west-2::/connectors/
TwilioNotifications/versions/2\", \"Parameters\": {\"TWILIO_ACCOUNT_SID
\": \"AC1a8d4204890840d7fc482aab38090d57\", \"TwilioAuthTokenSecretArn\":
 \"arn:aws:secretsmanager:us-west-2:123456789012:secret:greengrass-TwilioAuthToken-
ntSlp6\", \"TwilioAuthTokenSecretArn-ResourceId\": \"TwilioAuthToken\",
 \"DefaultFromPhoneNumber\": \"4254492999\"}}]"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/55d0052b-0d7d-44d6-b56f-21867215e118/versions/33f709a0-c825-49cb-9eea-
dc8964fbd635",
    "CreationTimestamp": "2019-06-24T20:46:30.134Z",
    "Id": "55d0052b-0d7d-44d6-b56f-21867215e118",
    "Version": "33f709a0-c825-49cb-9eea-dc8964fbd635"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnectorDefinitionVersion。

create-connector-definition

以下代码示例演示了如何使用 create-connector-definition。

Amazon CLI

创建连接器定义

以下 create-connector-definition 示例创建一个连接器定义和一个初始连接器定义版本。该初始版本包含一个连接器。版本中的所有连接器均定义其参数的值。

```
aws greengrass create-connector-definition \
    --name MySNSConnector \
    --initial-version "{\"Connectors\": [{\"Id\":\"MySNSConnector\",\"ConnectorArn
\":\"arn:aws:greengrass:us-west-2::/connectors/SNS/versions/1\",\"Parameters\":
 {\"DefaultSNSArn\":\"arn:aws:sns:us-west-2:123456789012:GGConnectorTopic\"}}]}"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "CreationTimestamp": "2019-06-19T19:30:01.300Z",
    "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
    "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-
c7c2-4a26-a7e2-7bf478ea2623",
    "Name": "MySNSConnector"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Greengrass 连接器入门（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConnectorDefinition。

create-core-definition-version

以下代码示例演示了如何使用 create-core-definition-version。

Amazon CLI

创建核心定义版本

以下 create-core-definition-version 示例创建一个核心定义版本，并将其与指定的核心定义相关联。该版本只能包含一个核心。在创建核心之前，您必须先创建和预置相应的 Amazon IoT 物品。此过程包括以下 iot 命令，这些命令返回 create-core-definition-version 命令所需的 ThingArn 和 CertificateArn。

创建与核心设备对应的 Amazon IoT 物品：

```
aws iot create-thing \
    --thing-name "MyCoreDevice"
```

输出：

```
{
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice",
    "thingName": "MyCoreDevice",
    "thingId": "cb419a19-9099-4515-9cec-e9b0e760608a"
}
```

为该物品创建公有和私有密钥以及核心设备证书。此示例使用 create-keys-and-certificate 命令并需要对当前目录的写入权限。或者，您可以使用 create-certificate-from-csr 命令。

```
aws iot create-keys-and-certificate \
    --set-as-active \
    --certificate-pem-outfile "myCore.cert.pem" \
    --public-key-outfile "myCore.public.key" \
    --private-key-outfile "myCore.private.key"
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz",
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
 CERTIFICATE-----\n",
```

```
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
 PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbMXDcNOK4rMCxDR...fvY4+te\n-----END
 RSA PRIVATE KEY-----\n"
    },
    "certificateId":
 "123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
}
```

创建允许执行 iot 和 greengrass 操作的 Amazon IoT 策略。为简单起见，以下策略允许对所有资源执行操作，但您的策略应更具限制性。

```
aws iot create-policy \
    --policy-name "Core_Devices" \
    --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}"
```

输出：

```
{
    "policyName": "Core_Devices",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/Core_Devices",
    "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}",
    "policyVersionId": "1"
}
```

将策略附加到证书：

```
aws iot attach-policy \
    --policy-name "Core_Devices" \
```

```
    --target "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

此命令不生成任何输出。

将物品附加到证书：

```
aws iot attach-thing-principal \
    --thing-name "MyCoreDevice" \
    --principal "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

此命令不生成任何输出。

创建核心定义版本：

```
aws greengrass create-core-definition-version \
    --core-definition-id "582efe12-b05a-409e-9a24-a2ba1bcc4a12" \
    --cores "[{\"Id\":\"MyCoreDevice\",\"ThingArn\":\"arn:aws:iot:us-
west-2:123456789012:thing/MyCoreDevice\",\"CertificateArn\":\"arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz
\",\"SyncShadow\":true}]"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12/versions/3fdc1190-2ce5-44de-b98b-
eec8f9571014",
    "Version": "3fdc1190-2ce5-44de-b98b-eec8f9571014",
    "CreationTimestamp": "2019-09-18T00:15:09.838Z",
    "Id": "582efe12-b05a-409e-9a24-a2ba1bcc4a12"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置 Amazon IoT Greengrass 核心。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCoreDefinitionVersion。

create-core-definition

以下代码示例演示了如何使用 create-core-definition。

Amazon CLI

示例 1：创建空核心定义

以下 create-core-definition 示例创建了一个空的（没有初始版本）Greengrass 核心定义。在核心可用之前，必须使用 create-core-definition-version 命令为核心提供其他参数。

```
aws greengrass create-core-definition \
    --name cliGroup_Core
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/
b5c08008-54cb-44bd-9eec-c121b04283b5",
    "CreationTimestamp": "2019-06-25T18:23:22.106Z",
    "Id": "b5c08008-54cb-44bd-9eec-c121b04283b5",
    "LastUpdatedTimestamp": "2019-06-25T18:23:22.106Z",
    "Name": "cliGroup_Core"
}
```

示例 2：使用初始版本创建核心定义

以下 create-core-definition 示例创建包含初始核心定义版本的核心定义。该版本只能包含一个核心。在创建核心之前，您必须先创建和预置相应的 Amazon IoT 物品。此过程包括以下 iot 命令，这些命令返回 create-core-definition 命令所需的 ThingArn 和 CertificateArn。

创建与核心设备对应的 Amazon IoT 物品：

```
aws iot create-thing \
    --thing-name "MyCoreDevice"
```

输出：

```
{
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice",
    "thingName": "MyCoreDevice",
    "thingId": "cb419a19-9099-4515-9cec-e9b0e760608a"
}
```

为该物品创建公有和私有密钥以及核心设备证书。此示例使用 create-keys-and-certificate 命令并需要对当前目录的写入权限。或者，您可以使用 create-certificate-from-csr 命令。

```
aws iot create-keys-and-certificate \
    --set-as-active \
    --certificate-pem-outfile "myCore.cert.pem" \
    --public-key-outfile "myCore.public.key" \
    --private-key-outfile "myCore.private.key"
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz",
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
 CERTIFICATE-----\n",
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
 PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbMXDcNOK4rMCxDR...fvY4+te\n-----END
 RSA PRIVATE KEY-----\n"
    },
    "certificateId":
 "123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
}
```

创建允许执行 iot 和 greengrass 操作的 Amazon IoT 策略。为简单起见，以下策略允许对所有资源执行操作，但您的策略应更具限制性。

```
aws iot create-policy \
    --policy-name "Core_Devices" \
    --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}"
```

输出：

```
{
    "policyName": "Core_Devices",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/Core_Devices",
    "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}",
    "policyVersionId": "1"
}
```

将策略附加到证书：

```
aws iot attach-policy \
    --policy-name "Core_Devices" \
    --target "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

此命令不生成任何输出。

将物品附加到证书：

```
aws iot attach-thing-principal \
    --thing-name "MyCoreDevice" \
    --principal "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

此命令不生成任何输出。

创建核心定义：

```
aws greengrass create-core-definition \
    --name "MyCores" \
    --initial-version "{\"Cores\":[{\"Id\":\"MyCoreDevice\",\"ThingArn\":
\"arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice\",\"CertificateArn\":
\"arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz
\",\"SyncShadow\":true}]}"
```

输出：

```
{
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12/versions/
cc87b5b3-8f4b-465d-944c-1d6de5dbfcdb",
    "Name": "MyCores",
    "LastUpdatedTimestamp": "2019-09-18T00:11:06.197Z",
    "LatestVersion": "cc87b5b3-8f4b-465d-944c-1d6de5dbfcdb",
    "CreationTimestamp": "2019-09-18T00:11:06.197Z",
    "Id": "582efe12-b05a-409e-9a24-a2ba1bcc4a12",
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置 Amazon IoT Greengrass 核心。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCoreDefinition。

`create-deployment`

以下代码示例演示了如何使用 `create-deployment`。

Amazon CLI

为 Greengrass 组的某个版本创建部署

以下 `create-deployment` 示例部署指定版本的 Greengrass 组。

```
aws greengrass create-deployment \
    --deployment-type NewDeployment \
    --group-id "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca" \
    --group-version-id "dc40c1e9-e8c8-4d28-a84d-a9cad5f599c9"
```

输出：

```
{
    "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca/deployments/bfceb608-4e97-45bc-
af5c-460144270308",
    "DeploymentId": "bfceb608-4e97-45bc-af5c-460144270308"
```

```
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的连接器入门（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeployment。

`create-device-definition-version`

以下代码示例演示了如何使用 `create-device-definition-version`。

Amazon CLI

创建设备定义版本

以下 `create-device-definition-version` 示例创建一个设备定义版本，并将其与指定的设备定义相关联。该版本定义两个设备。在可创建 Greengrass 设备之前，您必须先创建并预置相应的 Amazon IoT 物品。此过程包括以下 `iot` 命令，您必须运行这些命令才能获取 Greengrass 命令所需的信息：

创建与设备对应的 Amazon IoT 物品：

```
aws iot create-thing \
    --thing-name "InteriorTherm"
```

输出：

```
{
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm",
    "thingName": "InteriorTherm",
    "thingId": "01d4763c-78a6-46c6-92be-7add080394bf"
}
```

为该物品创建公有和私有密钥以及设备证书。此示例使用 `create-keys-and-certificate` 命令并需要对当前目录的写入权限。或者，您可以使用 `create-certificate-from-csr` 命令：

```
aws iot create-keys-and-certificate \
    --set-as-active \
    --certificate-pem-outfile "myDevice.cert.pem" \
    --public-key-outfile "myDevice.public.key" \
    --private-key-outfile "myDevice.private.key"
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
 CERTIFICATE-----\n",
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
 PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbMXDcNOK4rMCxDR...fvY4+te\n-----END
 RSA PRIVATE KEY-----\n"
    },
    "certificateId":
 "66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
}
```

创建允许执行 iot 和 greengrass 操作的 Amazon IoT 策略。为简单起见，以下策略允许对所有资源执行操作，但您的策略可能更具限制性：

```
aws iot create-policy \
    --policy-name "GG_Devices" \
    --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}"
```

输出：

```
{
    "policyName": "GG_Devices",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/GG_Devices",
    "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}",
    "policyVersionId": "1"
```

```
}
```

将策略附加到证书：

```
aws iot attach-policy \
    --policy-name "GG_Devices" \
    --target "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

将物品附加到证书

```
aws iot attach-thing-principal \
    --thing-name "InteriorTherm" \
    --principal "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

创建并配置如上所示的 IoT 物品后，在以下示例中使用前两个命令中的 ThingArn 和 CertificateArn。

```
aws greengrass create-device-definition-version \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \
    --devices "[{\"Id\":\"InteriorTherm\",\"ThingArn\":\"arn:aws:iot:us-
west-2:123456789012:thing/InteriorTherm\",\"CertificateArn\":\"arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92\",
\"SyncShadow\":true},{\"Id\":\"ExteriorTherm\",\"ThingArn\":\"arn:aws:iot:us-
west-2:123456789012:thing/ExteriorTherm\",\"CertificateArn\":\"arn:aws:iot:us-
west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02\",
\"SyncShadow\":true}]"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",
    "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
    "CreationTimestamp": "2019-09-11T00:15:09.838Z",
    "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeviceDefinitionVersion。

create-device-definition

以下代码示例演示了如何使用 create-device-definition。

Amazon CLI

创建设备定义

以下 create-device-definition 示例创建包含初始设备定义版本的设备定义。初始版本定义两个设备。在可创建 Greengrass 设备之前，您必须先创建并预置相应的 Amazon IoT 物品。此过程包括以下 iot 命令，您必须运行这些命令才能获取 Greengrass 命令所需的信息：

创建与设备对应的 Amazon IoT 物品：

```
aws iot create-thing \
    --thing-name "InteriorTherm"
```

输出：

```
{
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm",
    "thingName": "InteriorTherm",
    "thingId": "01d4763c-78a6-46c6-92be-7add080394bf"
}
```

为该物品创建公有和私有密钥以及设备证书。此示例使用 create-keys-and-certificate 命令并需要对当前目录的写入权限。或者，您可以使用 create-certificate-from-csr 命令：

```
aws iot create-keys-and-certificate \
    --set-as-active \
    --certificate-pem-outfile "myDevice.cert.pem" \
    --public-key-outfile "myDevice.public.key" \
    --private-key-outfile "myDevice.private.key"
```

输出：

```
{
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
 CERTIFICATE-----\n",
```

```
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
 PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbMXDcNOK4rMCxDR...fvY4+te\n-----END
 RSA PRIVATE KEY-----\n"
    },
    "certificateId":
 "66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
}
```

创建允许执行 iot 和 greengrass 操作的 Amazon IoT 策略。为简单起见，以下策略允许对所有资源执行操作，但您的策略可能更具限制性：

```
aws iot create-policy \
    --policy-name "GG_Devices" \
    --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}"
```

输出：

```
{
    "policyName": "GG_Devices",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/GG_Devices",
    "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":
[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
\":[\"*\"]}]}",
    "policyVersionId": "1"
}
```

将策略附加到证书：

```
aws iot attach-policy \
```

```
    --policy-name "GG_Devices" \
    --target "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

将物品附加到证书

```
aws iot attach-thing-principal \
    --thing-name "InteriorTherm" \
    --principal "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

创建并配置如上所示的 IoT 物品后，在以下示例中使用前两个命令中的 ThingArn 和 CertificateArn。

```
aws greengrass create-device-definition \
    --name "Sensors" \
    --initial-version "{\"Devices\":[{\"Id\":\"InteriorTherm
\",\"ThingArn\":\"arn:aws:iot:us-west-2:123456789012:thing/
InteriorTherm\",\"CertificateArn\":\"arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92\",
\"SyncShadow\":true},{\"Id\":\"ExteriorTherm\",\"ThingArn\":\"arn:aws:iot:us-
west-2:123456789012:thing/ExteriorTherm\",\"CertificateArn\":\"arn:aws:iot:us-
west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02\",
\"SyncShadow\":true}]}"
```

输出：

```
{
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/3b5cc510-58c1-44b5-9d98-4ad858ffa795",
    "Name": "Sensors",
    "LastUpdatedTimestamp": "2019-09-11T00:11:06.197Z",
    "LatestVersion": "3b5cc510-58c1-44b5-9d98-4ad858ffa795",
    "CreationTimestamp": "2019-09-11T00:11:06.197Z",
    "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeviceDefinition。

create-function-definition-version

以下代码示例演示了如何使用 create-function-definition-version。

Amazon CLI

创建函数定义的版本

以下 create-function-definition-version 示例创建指定函数定义的新版本。该版本指定单个函数，该函数的 ID 为 Hello-World-function，允许访问文件系统，并指定最大内存大小和超时期限。

```
aws greengrass create-function-definition-version \
    --cli-input-json "{\"FunctionDefinitionId\": \"e626e8c9-3b8f-4bf3-9cdc-
d26ecdeb9fa3\",\"Functions\": [{\"Id\": \"Hello-World-function\", \"FunctionArn\":
 \""arn:aws:lambda:us-
west-2:123456789012:function:Greengrass_HelloWorld_Counter:gghw-alias"\",
\"FunctionConfiguration\": {\"Environment\": {\"AccessSysfs\": true},\"Executable\":
 \"greengrassHelloWorldCounter.function_handler\",\"MemorySize\": 16000,\"Pinned\":
 false,\"Timeout\": 25}}]}"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/e626e8c9-3b8f-4bf3-9cdc-d26ecdeb9fa3/
versions/74abd1cc-637e-4abe-8684-9a67890f4043",
    "CreationTimestamp": "2019-06-25T22:03:43.376Z",
    "Id": "e626e8c9-3b8f-4bf3-9cdc-d26ecdeb9fa3",
    "Version": "74abd1cc-637e-4abe-8684-9a67890f4043"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFunctionDefinitionVersion。

create-function-definition

以下代码示例演示了如何使用 create-function-definition。

Amazon CLI

创建 Lambda 函数定义

以下 create-function-definition 示例通过提供 Lambda 函数列表（在本例中为仅包含一个名为 TempMonitorFunction 的函数的列表）及其配置来创建 Lambda 函数定义和初始版本。在创建函数定义之前，您需要 Lambda 函数 ARN。要创建函数及其别名，请使用 Lambda 的 create-function 和 publish-version 命令。Lambda 的 create-function 命令需要执行角色的 ARN，即使 Amazon IoT Greengrass 不使用该角色也是如此，因为权限是在 Greengrass 组角色中指定的。您可以使用 IAM create-role 命令创建空角色来获取 ARN 以用于 Lambda 的 create-function，也可以使用现有的执行角色。

```
aws greengrass create-function-definition \
    --name MyGreengrassFunctions \
    --initial-version "{\"Functions\": [{\"Id\": \"TempMonitorFunction\",
 \"FunctionArn\": \"arn:aws:lambda:us-
west-2:123456789012:function:TempMonitor:GG_TempMonitor\", \"FunctionConfiguration
\": {\"Executable\": \"temp_monitor.function_handler\", \"MemorySize\": 16000,
\"Timeout\": 5}}]}"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
functions/3b0d0080-87e7-48c6-b182-503ec743a08b",
    "CreationTimestamp": "2019-06-19T22:24:44.585Z",
    "Id": "3b0d0080-87e7-48c6-b182-503ec743a08b",
    "LastUpdatedTimestamp": "2019-06-19T22:24:44.585Z",
    "LatestVersion": "67f918b9-efb4-40b0-b87c-de8c9faf085b",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/3b0d0080-87e7-48c6-b182-503ec743a08b/versions/67f918b9-
efb4-40b0-b87c-de8c9faf085b",
    "Name": "MyGreengrassFunctions"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的如何使用 Amazon 命令行界面配置本地资源访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFunctionDefinition。

create-group-certificate-authority

以下代码示例演示了如何使用 create-group-certificate-authority。

Amazon CLI

为组创建证书颁发机构（CA）

以下 create-group-certificate-authority 示例为指定组创建或轮换 CA。

```
aws greengrass create-group-certificate-authority \
    --group-id "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1"
```

输出：

```
{
    "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/8eaadd72-ce4b-4f15-892a-0cc4f3a343f1/certificateauthorities/
d31630d674c4437f6c5dbc0dca56312a902171ce2d086c38e509c8EXAMPLEcc5"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Amazon IoT Greengrass 安全性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroupCertificateAuthority。

create-group-version

以下代码示例演示了如何使用 create-group-version。

Amazon CLI

创建 Greengrass 组的版本

以下 create-group-version 示例创建一个组版本并将其与指定的组相关联。该版本引用了核心、资源、连接器、函数和订阅版本，这些版本包含要包括在此组版本中的实体。必须先创建这些实体，然后才能创建组版本。

要创建包含初始版本的资源定义，请使用 create-resource-definition 命令。要创建包含初始版本的连接器定义，请使用 create-connector-definition 命令。要创建包含初始版本的函数定义，请使用 create-function-definition 命令。要创建包含初始版本的订阅定义，请使用 create-subscription-definition 命令。要检索最新核心定义版本的 ARN，请使用 get-group-version 命令并指定最新组版本的 ID。

```
aws greengrass create-group-version \
```

```
    --group-id "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca" \
    --core-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/6a630442-8708-4838-ad36-eb98849d975e/
versions/6c87151b-1fb4-4cb2-8b31-6ee715d8f8ba" \
    --resource-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/
a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1" \
    --connector-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/connectors/55d0052b-0d7d-44d6-b56f-21867215e118/
versions/78a3331b-895d-489b-8823-17b4f9f418a0" \
    --function-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/3b0d0080-87e7-48c6-b182-503ec743a08b/
versions/67f918b9-efb4-40b0-b87c-de8c9faf085b" \
    --subscription-definition-version-arn "arn:aws:greengrass:us-
west-2:123456789012:/greengrass/definition/subscriptions/9d611d57-5d5d-44bd-a3b4-
feccbdd69112/versions/aa645c47-ac90-420d-9091-8c7ffa4f103f"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/
ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca/versions/e10b0459-4345-4a09-88a4-1af1f5d34638",
    "CreationTimestamp": "2019-06-20T18:42:47.020Z",
    "Id": "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca",
    "Version": "e10b0459-4345-4a09-88a4-1af1f5d34638"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Amazon IoT Greengrass 组对象模型概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroupVersion。

`create-group`

以下代码示例演示了如何使用 `create-group`。

Amazon CLI

创建 Greeengrass 组

以下 `create-group` 示例创建名为 `cli-created-group` 的组。

```
aws greengrass create-group \
```

```
    --name cli-created-group
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/4e22bd92-898c-436b-ade5-434d883ff749",
    "CreationTimestamp": "2019-06-25T18:07:17.688Z",
    "Id": "4e22bd92-898c-436b-ade5-434d883ff749",
    "LastUpdatedTimestamp": "2019-06-25T18:07:17.688Z",
    "Name": "cli-created-group"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Amazon IoT Greengrass 组对象模型概述。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-logger-definition-version

以下代码示例演示了如何使用 create-logger-definition-version。

Amazon CLI

创建记录器定义版本

以下 create-logger-definition-version 示例创建一个记录器定义版本并将其与记录器定义相关联。该版本定义了四个日志记录配置：1) 核心设备文件系统上的系统组件日志，2) 核心设备文件系统上用户定义的 Lambda 函数日志，3) Amazon CloudWatch Logs 中的系统组件日志，以及 4) Amazon CloudWatch Logs 中用户定义的 Lambda 函数日志。注意：对于 CloudWatch Logs 集成，您的组角色必须授予相应的权限。

```
aws greengrass create-logger-definition-version \
    --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0" \
    --loggers "[{\"Id\":\"1\",\"Component\":\"GreengrassSystem\",\"Level\":\"ERROR
\",\"Space\":10240,\"Type\":\"FileSystem\"},{\"Id\":\"2\",\"Component\":\"Lambda
\",\"Level\":\"INFO\",\"Space\":10240,\"Type\":\"FileSystem\"},{\"Id\":\"3\",
\"Component\":\"GreengrassSystem\",\"Level\":\"WARN\",\"Type\":\"AWSCloudWatch\"},
{\"Id\":\"4\",\"Component\":\"Lambda\",\"Level\":\"INFO\",\"Type\":\"AWSCloudWatch
\"}]"
```

输出：

```
{
 "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/loggers/
a454b62a-5d56-4ca9-bdc4-8254e1662cb0/versions/49aedb1e-01a3-4d39-9871-3a052573f1ea",
 "Version": "49aedb1e-01a3-4d39-9871-3a052573f1ea",
 "CreationTimestamp": "2019-07-24T00:04:48.523Z",
 "Id": "a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Amazon IoT Greengrass 日志进行监控。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoggerDefinitionVersion。

`create-logger-definition`

以下代码示例演示了如何使用 `create-logger-definition`。

Amazon CLI

创建记录器定义

以下 `create-logger-definition` 示例创建包含初始记录器定义版本的记录器定义。该初始版本定义了三个日志记录配置：1) 核心设备文件系统上的系统组件日志，2) 核心设备文件系统上用户定义的 Lambda 函数日志，以及 3) Amazon CloudWatch Logs 中用户定义的 Lambda 函数日志。注意：对于 CloudWatch Logs 集成，您的组角色必须授予相应的权限。

```
aws greengrass create-logger-definition \
    --name "LoggingConfigs" \
    --initial-version "{\"Loggers\":[{\"Id\":\"1\",\"Component\":\"GreengrassSystem
\",\"Level\":\"ERROR\",\"Space\":10240,\"Type\":\"FileSystem\"},{\"Id\":
\"2\",\"Component\":\"Lambda\",\"Level\":\"INFO\",\"Space\":10240,\"Type\":
\"FileSystem\"},{\"Id\":\"3\",\"Component\":\"Lambda\",\"Level\":\"INFO\",\"Type\":
\"AWSCloudWatch\"}]}"
```

输出：

```
{
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/a454b62a-5d56-4ca9-bdc4-8254e1662cb0/versions/de1d9854-1588-4525-
b25e-b378f60f2322",
```

```
    "Name": "LoggingConfigs",
    "LastUpdatedTimestamp": "2019-07-23T23:52:17.165Z",
    "LatestVersion": "de1d9854-1588-4525-b25e-b378f60f2322",
    "CreationTimestamp": "2019-07-23T23:52:17.165Z",
    "Id": "a454b62a-5d56-4ca9-bdc4-8254e1662cb0",
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
loggers/a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Amazon IoT Greengrass 日志进行监控。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoggerDefinition。

`create-resource-definition-version`

以下代码示例演示了如何使用 `create-resource-definition-version`。

Amazon CLI

创建资源定义的版本

以下 `create-resource-definition-version` 示例创建 TwilioAuthToken 的新版本。

```
aws greengrass create-resource-definition-version \
    --resource-definition-id "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38" \
    --resources "[{\"Id\": \"TwilioAuthToken\",\"Name\": \"MyTwilioAuthToken
\",\"ResourceDataContainer\": {\"SecretsManagerSecretResourceData\": {\"ARN\":
 \"arn:aws:secretsmanager:us-west-2:123456789012:secret:greengrass-TwilioAuthToken-
ntSlp6\"}}}]"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/b3bcada0-5fb6-42df-
bf0b-1ee4f15e769e",
    "CreationTimestamp": "2019-06-24T21:17:25.623Z",
    "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
    "Version": "b3bcada0-5fb6-42df-bf0b-1ee4f15e769e"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResourceDefinitionVersion。

create-resource-definition

以下代码示例演示了如何使用 create-resource-definition。

Amazon CLI

创建资源定义

以下 create-resource-definition 示例创建资源定义，其中包含要在 Greengrass 组中使用的资源列表。在此示例中，通过提供资源列表来包含资源定义的初始版本。该列表包括一个用于 Twilio 授权令牌的资源和用于存储在 Amazon Secrets Manager 中的密钥的 ARN。必须先创建密钥，然后才能创建资源定义。

```
aws greengrass create-resource-definition \
    --name MyGreengrassResources \
    --initial-version "{\"Resources\": [{\"Id\": \"TwilioAuthToken
\",\"Name\": \"MyTwilioAuthToken\",\"ResourceDataContainer\":
 {\"SecretsManagerSecretResourceData\": {\"ARN\": \"arn:aws:secretsmanager:us-
west-2:123456789012:secret:greengrass-TwilioAuthToken-ntSlp6\"}}}]}"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
    "CreationTimestamp": "2019-06-19T21:51:28.212Z",
    "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
    "LastUpdatedTimestamp": "2019-06-19T21:51:28.212Z",
    "LatestVersion": "a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/a5f94d0b-
f6bc-40f4-bb78-7a1c5fe13ba1",
    "Name": "MyGreengrassResources"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的如何使用 Amazon 命令行界面配置本地资源访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResourceDefinition。

create-software-update-job

以下代码示例演示了如何使用 create-software-update-job。

Amazon CLI

为核心创建软件更新作业

以下 create-software-update-job 示例创建一个空中下载（OTA）更新作业，以更新名称为 MyFirstGroup_Core 的核心上的 Amazon IoT Greengrass 核心软件。此命令需要一个 IAM 角色，该角色允许访问 Amazon S3 中的软件更新程序包，并包含 iot.amazonaws.com 作为可信实体。

```
aws greengrass create-software-update-job \
    --update-targets-architecture armv7l \
    --update-targets [\"arn:aws:iot:us-west-2:123456789012:thing/MyFirstGroup_Core
\"] \
    --update-targets-operating-system raspbian \
    --software-to-update core \
    --s3-url-signer-role arn:aws:iam::123456789012:role/OTA_signer_role \
    --update-agent-log-level WARN
```

输出：

```
{
    "IotJobId": "GreengrassUpdateJob_30b353e3-3af7-4786-be25-4c446663c09e",
    "IotJobArn": "arn:aws:iot:us-west-2:123456789012:job/
GreengrassUpdateJob_30b353e3-3af7-4786-be25-4c446663c09e",
    "PlatformSoftwareVersion": "1.9.3"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Amazon IoT Greengrass 核心软件的 OTA 更新。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSoftwareUpdateJob。

create-subscription-definition-version

以下代码示例演示了如何使用 create-subscription-definition-version。

Amazon CLI

创建订阅定义的新版本

以下 create-subscription-definition-version 示例创建包含三个订阅的新版本的订阅定义：触发通知、温度输入和输出状态。

```
aws greengrass create-subscription-definition-version \
    --subscription-definition-id "9d611d57-5d5d-44bd-a3b4-feccbdd69112" \
    --subscriptions "[{\"Id\": \"TriggerNotification\", \"Source\":
 \"arn:aws:lambda:us-west-2:123456789012:function:TempMonitor:GG_TempMonitor
\", \"Subject\": \"twilio/txt\", \"Target\": \"arn:aws:greengrass:us-west-2::/
connectors/TwilioNotifications/versions/1\"},{\"Id\": \"TemperatureInput\", \"Source
\": \"cloud\", \"Subject\": \"temperature/input\", \"Target\": \"arn:aws:lambda:us-
west-2:123456789012:function:TempMonitor:GG_TempMonitor\"},{\"Id\": \"OutputStatus
\", \"Source\": \"arn:aws:greengrass:us-west-2::/connectors/TwilioNotifications/
versions/1\", \"Subject\": \"twilio/message/status\", \"Target\": \"cloud\"}]"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112/versions/7b65dfae-50b6-4d0f-
b3e0-27728bfb0620",
    "CreationTimestamp": "2019-06-24T21:21:33.837Z",
    "Id": "9d611d57-5d5d-44bd-a3b4-feccbdd69112",
    "Version": "7b65dfae-50b6-4d0f-b3e0-27728bfb0620"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscriptionDefinitionVersion。

create-subscription-definition

以下代码示例演示了如何使用 create-subscription-definition。

Amazon CLI

创建订阅定义

以下 create-subscription-definition 示例创建订阅定义并指定其初始版本。该初始版本包含三个订阅：一个用于连接器订阅的 MQTT 主题，一个用于允许函数从 Amazon IoT 接收温度读

数，另一个用于允许 Amazon IoT 从连接器接收状态信息。该示例提供了之前通过使用 Lambda 的 `create-alias` 命令创建的 Lambda 函数别名的 ARN。

```
aws greengrass create-subscription-definition \
    --initial-version "{\"Subscriptions\": [{\"Id\":
 \"TriggerNotification\", \"Source\": \"arn:aws:lambda:us-
west-2:123456789012:function:TempMonitor:GG_TempMonitor\", \"Subject\":
 \"twilio/txt\", \"Target\": \"arn:aws:greengrass:us-west-2::/connectors/
TwilioNotifications/versions/1\"},{\"Id\": \"TemperatureInput\", \"Source\":
 \"cloud\", \"Subject\": \"temperature/input\", \"Target\": \"arn:aws:lambda:us-
west-2:123456789012:function:TempMonitor:GG_TempMonitor\"},{\"Id\": \"OutputStatus
\", \"Source\": \"arn:aws:greengrass:us-west-2::/connectors/TwilioNotifications/
versions/1\", \"Subject\": \"twilio/message/status\", \"Target\": \"cloud\"}]}"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112",
    "CreationTimestamp": "2019-06-19T22:34:26.677Z",
    "Id": "9d611d57-5d5d-44bd-a3b4-feccbdd69112",
    "LastUpdatedTimestamp": "2019-06-19T22:34:26.677Z",
    "LatestVersion": "aa645c47-ac90-420d-9091-8c7ffa4f103f",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112/versions/aa645c47-
ac90-420d-9091-8c7ffa4f103f"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的连接器入门（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscriptionDefinition。

delete-connector-definition

以下代码示例演示了如何使用 `delete-connector-definition`。

Amazon CLI

删除连接器定义

以下 `delete-connector-definition` 示例删除指定的 Greengrass 连接器定义。如果删除组使用的连接器定义，则无法成功部署该组。

```
aws greengrass delete-connector-definition \
    --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConnectorDefinition。

delete-core-definition

以下代码示例演示了如何使用 delete-core-definition。

Amazon CLI

删除核心定义

以下 delete-core-definition 示例删除指定的 Greengrass 核心定义，包括所有版本。如果您删除与 Greengrass 组关联的核心，则将无法成功部署该组。

```
aws greengrass delete-core-definition \
    --core-definition-id "ff36cc5f-9f98-4994-b468-9d9b6dc52abd"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCoreDefinition。

delete-device-definition

以下代码示例演示了如何使用 delete-device-definition。

Amazon CLI

删除设备定义

以下 delete-device-definition 示例删除指定的设备定义，包括其所有版本。如果您删除组版本使用的设备定义版本，则无法成功部署该组版本。

```
aws greengrass delete-device-definition \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeviceDefinition。

delete-function-definition

以下代码示例演示了如何使用 delete-function-definition。

Amazon CLI

删除函数定义

以下 delete-function-definition 示例删除指定的 Greengrass 函数定义。如果您删除组使用的函数定义，则无法成功部署该组。

```
aws greengrass delete-function-definition \
    --function-definition-id "fd4b906a-dff3-4c1b-96eb-52ebfcfac06a"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFunctionDefinition。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除组

以下 delete-group 示例删除指定的 Greengrass 组。

```
aws greengrass delete-group \
    --group-id "4e22bd92-898c-436b-ade5-434d883ff749"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-logger-definition

以下代码示例演示了如何使用 delete-logger-definition。

Amazon CLI

删除记录器定义

以下 delete-logger-definition 示例删除指定的记录器定义，包括所有记录器定义版本。如果您删除组版本使用的记录器定义版本，则无法成功部署该组版本。

```
aws greengrass delete-logger-definition \
    --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Amazon IoT Greengrass 日志进行监控。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoggerDefinition。

delete-resource-definition

以下代码示例演示了如何使用 delete-resource-definition。

Amazon CLI

删除资源定义

以下 delete-resource-definition 示例删除指定的资源定义，包括所有资源版本。如果您删除组使用的资源定义，则无法成功部署该组。

```
aws greengrass delete-resource-definition \
    --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourceDefinition。

delete-subscription-definition

以下代码示例演示了如何使用 delete-subscription-definition。

Amazon CLI

删除订阅定义

以下 delete-subscription-definition 示例删除指定的 Greengrass 订阅定义。如果您删除组正在使用的订阅，则无法成功部署该组。

```
aws greengrass delete-subscription-definition \
    --subscription-definition-id "cd6f1c37-d9a4-4e90-be94-01a7404f5967"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubscriptionDefinition。

disassociate-role-from-group

以下代码示例演示了如何使用 disassociate-role-from-group。

Amazon CLI

取消角色与 Greengrass 组的关联

以下 disassociate-role-from-group 示例取消 IAM 角色与指定 Greengrass 组的关联。

```
aws greengrass disassociate-role-from-group \
    --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b
```

输出：

```
{
    "DisassociatedAt": "2019-09-10T20:05:49Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置组角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateRoleFromGroup。

disassociate-service-role-from-account

以下代码示例演示了如何使用 disassociate-service-role-from-account。

Amazon CLI

取消服务角色与您的 Amazon 账户的关联

以下 disassociate-service-role-from-account 示例删除与您的 Amazon 账户关联的服务角色。如果您未在任何 Amazon 区域中使用服务角色，请使用 delete-role-policy 命令将

AWSGreengrassResourceAccessRolePolicy 托管策略与角色分离，然后使用 delete-role 命令删除该角色。

```
aws greengrass disassociate-service-role-from-account
```

输出：

```
{
    "DisassociatedAt": "2019-06-25T22:12:55Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateServiceRoleFromAccount。

get-associated-role

以下代码示例演示了如何使用 get-associated-role。

Amazon CLI

获取与 Greengrass 组关联的角色

以下 get-associated-role 示例获取与指定 Greengrass 组关联的 IAM 角色。本地 Lambda 函数和连接器使用该组角色来访问 Amazon 服务。

```
aws greengrass get-associated-role \
    --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b
```

输出：

```
{
    "RoleArn": "arn:aws:iam::123456789012:role/GG-Group-Role",
    "AssociatedAt": "2019-09-10T20:03:30Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置组角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAssociatedRole。

get-bulk-deployment-status

以下代码示例演示了如何使用 get-bulk-deployment-status。

Amazon CLI

检查批量部署的状态

以下 get-bulk-deployment-status 示例检索指定批量部署操作的状态信息。在此示例中，指定要部署的组的文件具有无效的输入记录。

```
aws greengrass get-bulk-deployment-status \
    --bulk-deployment-id "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
```

输出：

```
{
    "BulkDeploymentMetrics": {
        "InvalidInputRecords": 1,
        "RecordsProcessed": 1,
        "RetryAttempts": 0
    },
    "BulkDeploymentStatus": "Completed",
    "CreatedAt": "2019-06-25T16:11:33.265Z",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的为组创建批量部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBulkDeploymentStatus。

get-connectivity-info

以下代码示例演示了如何使用 get-connectivity-info。

Amazon CLI

获取 Greengrass 核心的连接信息

以下 get-connectivity-info 示例显示设备可用于连接到指定 Greengrass 核心的端点。连接信息是 IP 地址或域名的列表，以及相应的端口号和可选的客户定义元数据。

```
aws greengrass get-connectivity-info \
```

```
    --thing-name "MyGroup_Core"
```

输出：

```
{
    "ConnectivityInfo": [
        {
            "Metadata": "",
            "PortNumber": 8883,
            "HostAddress": "127.0.0.1",
            "Id": "AUTOIP_127.0.0.1_0"
        },
        {
            "Metadata": "",
            "PortNumber": 8883,
            "HostAddress": "192.168.1.3",
            "Id": "AUTOIP_192.168.1.3_1"
        },
        {
            "Metadata": "",
            "PortNumber": 8883,
            "HostAddress": "::1",
            "Id": "AUTOIP_::1_2"
        },
        {
            "Metadata": "",
            "PortNumber": 8883,
            "HostAddress": "fe80::1e69:ed93:f5b:f6d",
            "Id": "AUTOIP_fe80::1e69:ed93:f5b:f6d_3"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnectivityInfo。

get-connector-definition-version

以下代码示例演示了如何使用 get-connector-definition-version。

Amazon CLI

检索有关连接器定义特定版本的信息

以下 get-connector-definition-version 示例检索有关指定连接器定义的指定版本的信息。要检索所有版本的连接器定义的 ID，请使用 list-connector-definition-versions 命令。要检索添加到连接器定义的上个版本的 ID，请使用 get-connector-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-connector-definition-version \
    --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8" \
    --connector-definition-version-id "63c57963-c7c2-4a26-a7e2-7bf478ea2623"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-c7c2-4a26-
a7e2-7bf478ea2623",
    "CreationTimestamp": "2019-06-19T19:30:01.300Z",
    "Definition": {
        "Connectors": [
            {
                "ConnectorArn": "arn:aws:greengrass:us-west-2::/connectors/SNS/
versions/1",
                "Id": "MySNSConnector",
                "Parameters": {
                    "DefaultSNSArn": "arn:aws:sns:us-
west-2:123456789012:GGConnectorTopic"
                }
            }
        ]
    },
    "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "Version": "63c57963-c7c2-4a26-a7e2-7bf478ea2623"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Greengrass 连接器与服务和协议集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnectorDefinitionVersion。

get-connector-definition

以下代码示例演示了如何使用 get-connector-definition。

Amazon CLI

检索有关连接器定义的信息

以下 get-connector-definition 示例检索有关指定连接器定义的信息。要检索连接器定义的 ID，请使用 list-connector-definitions 命令。

```
aws greengrass get-connector-definition \
    --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "CreationTimestamp": "2019-06-19T19:30:01.300Z",
    "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
    "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-
c7c2-4a26-a7e2-7bf478ea2623",
    "Name": "MySNSConnector",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Greengrass 连接器与服务和协议集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnectorDefinition。

get-core-definition-version

以下代码示例演示了如何使用 get-core-definition-version。

Amazon CLI

检索有关 Greengrass 核心定义特定版本的详细信息

以下 get-core-definition-version 示例检索有关指定核心定义的指定版本的信息。要检索所有版本的核心定义的 ID，请使用 list-core-definition-versions 命令。要检索添加到核心定义的上个版本的 ID，请使用 get-core-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-core-definition-version \
    --core-definition-id "c906ed39-a1e3-4822-a981-7b9bd57b4b46"  \
    --core-definition-version-id "42aeeac3-fd9d-4312-a8fd-ffa9404a20e0"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/
c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeeac3-fd9d-4312-a8fd-ffa9404a20e0",
    "CreationTimestamp": "2019-06-18T16:21:21.351Z",
    "Definition": {
        "Cores": [
            {
                "CertificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/928dea7b82331b47c3ff77b0e763fc5e64e2f7c884e6ef391baed9b6b8e21b45",
                "Id": "1a39aac7-0885-4417-91f6-23e4cea6c511",
                "SyncShadow": false,
                "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/
GGGroup4Pi3_Core"
            }
        ]
    },
    "Id": "c906ed39-a1e3-4822-a981-7b9bd57b4b46",
    "Version": "42aeeac3-fd9d-4312-a8fd-ffa9404a20e0"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCoreDefinitionVersion。

get-core-definition

以下代码示例演示了如何使用 get-core-definition。

Amazon CLI

检索 Greengrass 核心定义的详细信息

以下 get-core-definition 示例检索有关指定核心定义的信息。要检索核心定义的 ID，请使用 list-core-definitions 命令。

```
aws greengrass get-core-definition \
    --core-definition-id "c906ed39-a1e3-4822-a981-7b9bd57b4b46"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
cores/237d6916-27cf-457f-ba0c-e86cfb5d25cd",
    "CreationTimestamp": "2018-10-18T04:47:06.721Z",
    "Id": "237d6916-27cf-457f-ba0c-e86cfb5d25cd",
    "LastUpdatedTimestamp": "2018-10-18T04:47:06.721Z",
    "LatestVersion": "bd2cd6d4-2bc5-468a-8962-39e071e34b68",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/237d6916-27cf-457f-ba0c-e86cfb5d25cd/versions/
bd2cd6d4-2bc5-468a-8962-39e071e34b68",
    "tags": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCoreDefinition。

get-deployment-status

以下代码示例演示了如何使用 get-deployment-status。

Amazon CLI

检索部署的状态

以下 get-deployment-status 示例检索指定 Greengrass 组的指定部署的状态。要获取部署 ID，请使用 list-deployments 命令并指定组 ID。

```
aws greengrass get-deployment-status \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731" \
    --deployment-id "1065b8a0-812b-4f21-9d5d-e89b232a530f"
```

输出：

```
{
    "DeploymentStatus": "Success",
    "DeploymentType": "NewDeployment",
    "UpdatedAt": "2019-06-18T17:04:44.761Z"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeploymentStatus。

get-device-definition-version

以下代码示例演示了如何使用 get-device-definition-version。

Amazon CLI

获取设备定义版本

以下 get-device-definition-version 示例检索有关指定设备定义的指定版本的信息。要检索设备定义的所有版本的 ID，请使用 list-device-definition-versions 命令。要检索添加到设备定义的上个版本的 ID，请使用 get-device-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-device-definition-version \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \
    --device-definition-version-id "83c13984-6fed-447e-84d5-5b8aa45d5f71"
```

输出：

```
{
    "Definition": {
        "Devices": [
            {
                "CertificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02",
                "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/
ExteriorTherm",
                "SyncShadow": true,
                "Id": "ExteriorTherm"
            },
            {
                "CertificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
                "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/
InteriorTherm",
                "SyncShadow": true,
                "Id": "InteriorTherm"
            }
        ]
    },
    "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
    "CreationTimestamp": "2019-09-11T00:15:09.838Z",
    "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
```

```
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeviceDefinitionVersion。

get-device-definition

以下代码示例演示了如何使用 get-device-definition。

Amazon CLI

获取设备定义

以下 get-device-definition 示例检索有关指定设备定义的信息。要检索设备定义的 ID，请使用 list-device-definitions 命令。

```
aws greengrass get-device-definition \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

输出：

```
{
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",
    "Name": "TemperatureSensors",
    "tags": {},
    "LastUpdatedTimestamp": "2019-09-11T00:19:03.698Z",
    "LatestVersion": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
    "CreationTimestamp": "2019-09-11T00:11:06.197Z",
    "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeviceDefinition。

get-function-definition-version

以下代码示例演示了如何使用 get-function-definition-version。

Amazon CLI

检索有关 Lambda 函数特定版本的详细信息

以下 `get-function-definition-version` 检索有关指定函数定义的指定版本的信息。要检索所有版本的函数定义的 ID，请使用 `list-function-definition-versions` 命令。要检索添加到函数定义的上个版本的 ID，请使用 `get-function-definition` 命令并检查 `LatestVersion` 属性。

```
aws greengrass get-function-definition-version \
    --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85" \
    --function-definition-version-id "9748fda7-1589-4fcc-ac94-f5559e88678b"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/versions/9748fda7-1589-4fcc-ac94-
f5559e88678b",
    "CreationTimestamp": "2019-06-18T17:04:30.776Z",
    "Definition": {
        "Functions": [
            {
                "FunctionArn": "arn:aws:lambda:::function:GGIPDetector:1",
                "FunctionConfiguration": {
                    "Environment": {},
                    "MemorySize": 32768,
                    "Pinned": true,
                    "Timeout": 3
                },
                "Id": "26b69bdb-e547-46bc-9812-84ec04b6cc8c"
            },
            {
                "FunctionArn": "arn:aws:lambda:us-
west-2:123456789012:function:Greengrass_HelloWorld:GG_HelloWorld",
                "FunctionConfiguration": {
                    "EncodingType": "json",
                    "Environment": {
                        "Variables": {}
                    },
                    "MemorySize": 16384,
                    "Pinned": true,
                    "Timeout": 25
```

```
                },
                "Id": "384465a8-eedf-48c6-b793-4c35f7bfae9b"
            }
        ]
    },
    "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
    "Version": "9748fda7-1589-4fcc-ac94-f5559e88678b"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunctionDefinitionVersion。

get-function-definition

以下代码示例演示了如何使用 get-function-definition。

Amazon CLI

检索函数定义

以下 get-function-definition 示例显示指定函数定义的详细信息。要检索函数定义的 ID，请使用 list-function-definitions 命令。

```
aws greengrass get-function-definition \
    --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
    "CreationTimestamp": "2019-06-18T16:21:21.431Z",
    "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
    "LastUpdatedTimestamp": "2019-06-18T16:21:21.431Z",
    "LatestVersion": "9748fda7-1589-4fcc-ac94-f5559e88678b",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",
    "tags": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunctionDefinition。

get-group-certificate-authority

以下代码示例演示了如何使用 get-group-certificate-authority。

Amazon CLI

检索与 Greengrass 组关联的 CA

以下 get-group-certificate-authority 示例检索与指定 Greengrass 组关联的证书颁发机构（CA）。要获取证书颁发机构 ID，请使用 list-group-certificate-authorities 命令并指定组 ID。

```
aws greengrass get-group-certificate-authority \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731" \
    --certificate-authority-
id "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6"
```

输出：

```
{
    "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/certificateauthorities/
f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",
    "GroupCertificateAuthorityId":
 "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",
    "PemEncodedCertificate": "-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBWEXAMPLEGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDEXAMPLEEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAEXAMPLESBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jEXAMPLENMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0EXAMPLEBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWEXAMPLEDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWEXAMPLEgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5EXAMPLE8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CEXAMPLE93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswYEXAMPLEgpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKEXAMPLEAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----\n"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupCertificateAuthority。

get-group-certificate-configuration

以下代码示例演示了如何使用 get-group-certificate-configuration。

Amazon CLI

检索 Greengrass 组使用的证书颁发机构的配置

以下 get-group-certificate-configuration 示例检索指定 Greengrass 组使用的证书颁发机构（CA）的配置。

```
aws greengrass get-group-certificate-configuration \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

输出：

```
{
    "CertificateAuthorityExpiryInMilliseconds": 2524607999000,
    "CertificateExpiryInMilliseconds": 604800000,
    "GroupId": "1013db12-8b58-45ff-acc7-704248f66731"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupCertificateConfiguration。

get-group-version

以下代码示例演示了如何使用 get-group-version。

Amazon CLI

检索有关 Greengrass 组版本的信息

以下 get-group-version 示例检索有关指定组的指定版本的信息。要检索组的所有版本的 ID，请使用 list-group-versions 命令。要检索添加到组的上个版本的 ID，请使用 get-group 命令并检查 LatestVersion 属性。

```
aws greengrass get-group-version \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"  \
    --group-version-id "115136b3-cfd7-4462-b77f-8741a4b00e5e"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
    "CreationTimestamp": "2019-06-18T17:04:30.915Z",
    "Definition": {
        "CoreDefinitionVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeeac3-
fd9d-4312-a8fd-ffa9404a20e0",
        "FunctionDefinitionVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",
        "SubscriptionDefinitionVersionArn": "arn:aws:greengrass:us-
west-2:123456789012:/greengrass/definition/subscriptions/70e49321-83d5-45d2-
bc09-81f4917ae152/versions/88ae8699-12ac-4663-ba3f-4d7f0519140b"
    },
    "Id": "1013db12-8b58-45ff-acc7-704248f66731",
    "Version": "115136b3-cfd7-4462-b77f-8741a4b00e5e"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupVersion。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

检索有关 Greengrass 组的信息

以下 get-group 示例检索有关指定 Greengrass 组的信息。要检索组的 ID，请使用 list-groups 命令。

```
aws greengrass get-group \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

输出：

```
{
```

```
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731",
    "CreationTimestamp": "2019-06-18T16:21:21.457Z",
    "Id": "1013db12-8b58-45ff-acc7-704248f66731",
    "LastUpdatedTimestamp": "2019-06-18T16:21:21.457Z",
    "LatestVersion": "115136b3-cfd7-4462-b77f-8741a4b00e5e",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
    "Name": "GGGroup4Pi3",
    "tags": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

get-logger-definition-version

以下代码示例演示了如何使用 get-logger-definition-version。

Amazon CLI

检索有关记录器定义版本的信息

以下 get-logger-definition-version 示例检索有关指定记录器定义的指定版本的信息。要检索所有版本的记录器定义的 ID，请使用 list-logger-definition-versions 命令。要检索添加到记录器定义的上个版本的 ID，请使用 get-logger-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-logger-definition-version \
    --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23" \
    --logger-definition-version-id "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
    "CreationTimestamp": "2019-05-08T16:10:13.866Z",
    "Definition": {
        "Loggers": []
```

```
    },
    "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
    "Version": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoggerDefinitionVersion。

get-logger-definition

以下代码示例演示了如何使用 get-logger-definition。

Amazon CLI

检索有关记录器定义的信息

以下 get-logger-definition 示例检索有关指定记录器定义的信息。要检索记录器定义的 ID，请使用 list-logger-definitions 命令。

```
aws greengrass get-logger-definition \
    --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23",
    "CreationTimestamp": "2019-05-08T16:10:13.809Z",
    "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
    "LastUpdatedTimestamp": "2019-05-08T16:10:13.809Z",
    "LatestVersion": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
    "tags": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoggerDefinition。

get-resource-definition-version

以下代码示例演示了如何使用 get-resource-definition-version。

Amazon CLI

检索有关资源定义的特定版本的信息

以下 get-resource-definition-version 示例检索有关指定资源定义的指定版本的信息。要检索资源定义的所有版本的 ID，请使用 list-resource-definition-versions 命令。要检索添加到资源定义的上个版本的 ID，请使用 get-resource-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-resource-definition-version \
    --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658" \
    --resource-definition-version-id "26e8829a-491a-464d-9c87-664bf6f6f2be"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
    "CreationTimestamp": "2019-06-19T16:40:59.392Z",
    "Definition": {
        "Resources": [
            {
                "Id": "26ff3f7b-839a-4217-9fdc-a218308b3963",
                "Name": "usb-port",
                "ResourceDataContainer": {
                    "LocalDeviceResourceData": {
                        "GroupOwnerSetting": {
                            "AutoAddGroupOwner": false
                        },
                        "SourcePath": "/dev/bus/usb"
                    }
                }
            }
        ]
    },
    "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
    "Version": "26e8829a-491a-464d-9c87-664bf6f6f2be"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceDefinitionVersion。

get-resource-definition

以下代码示例演示了如何使用 get-resource-definition。

Amazon CLI

检索有关资源定义的信息

以下 get-resource-definition 示例检索有关指定资源定义的信息。要检索资源定义的 ID，请使用 list-resource-definitions 命令。

```
aws greengrass get-resource-definition \
    --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658",
    "CreationTimestamp": "2019-06-19T16:40:59.261Z",
    "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
    "LastUpdatedTimestamp": "2019-06-19T16:40:59.261Z",
    "LatestVersion": "26e8829a-491a-464d-9c87-664bf6f6f2be",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
    "tags": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceDefinition。

get-service-role-for-account

以下代码示例演示了如何使用 get-service-role-for-account。

Amazon CLI

检索附加到账户的服务角色的详细信息

以下 get-service-role-for-account 示例检索有关附加到 Amazon 账户的服务角色的信息。

```
aws greengrass get-service-role-for-account
```

输出：

```
{
    "AssociatedAt": "2018-10-18T15:59:20Z",
    "RoleArn": "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceRoleForAccount。

get-subscription-definition-version

以下代码示例演示了如何使用 get-subscription-definition-version。

Amazon CLI

检索有关订阅定义的特定版本的信息

以下 get-subscription-definition-version 示例检索有关指定订阅定义的指定版本的信息。要检索订阅定义的所有版本的 ID，请使用 list-subscription-definition-versions 命令。要检索添加到订阅定义的上个版本的 ID，请使用 get-subscription-definition 命令并检查 LatestVersion 属性。

```
aws greengrass get-subscription-definition-version \
    --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152" \
    --subscription-definition-version-id "88ae8699-12ac-4663-ba3f-4d7f0519140b"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/versions/88ae8699-12ac-4663-
ba3f-4d7f0519140b",
    "CreationTimestamp": "2019-06-18T17:03:52.499Z",
    "Definition": {
        "Subscriptions": [
            {
```

```
                "Id": "692c4484-d89f-4f64-8edd-1a041a65e5b6",
                "Source": "arn:aws:lambda:us-
west-2:123456789012:function:Greengrass_HelloWorld:GG_HelloWorld",
                "Subject": "hello/world",
                "Target": "cloud"
            }
        ]
    },
    "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
    "Version": "88ae8699-12ac-4663-ba3f-4d7f0519140b"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubscriptionDefinitionVersion。

get-subscription-definition

以下代码示例演示了如何使用 get-subscription-definition。

Amazon CLI

检索有关订阅定义的信息

以下 get-subscription-definition 示例检索有关指定订阅定义的信息。要检索订阅定义的 ID，请使用 list-subscription-definitions 命令。

```
aws greengrass get-subscription-definition \
    --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152"
```

输出：

```
{
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152",
    "CreationTimestamp": "2019-06-18T17:03:52.392Z",
    "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
    "LastUpdatedTimestamp": "2019-06-18T17:03:52.392Z",
    "LatestVersion": "88ae8699-12ac-4663-ba3f-4d7f0519140b",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b",
    "tags": {}
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubscriptionDefinition。

get-thing-runtime-configuration

以下代码示例演示了如何使用 get-thing-runtime-configuration。

Amazon CLI

检索 Greengrass 核心的运行时配置

以下 get-thing-runtime-configuration 示例检索 Greengrass 核心的运行时配置。在您能够检索运行时配置之前，必须使用 update-thing-runtime-configuration 命令为核心创建运行时配置。

```
aws greengrass get-thing-runtime-configuration \
    --thing-name SampleGreengrassCore
```

输出：

```
{
    "RuntimeConfiguration": {
        "TelemetryConfiguration": {
            "ConfigurationSyncStatus": "OutOfSync",
            "Telemetry": "On"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置遥测设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetThingRuntimeConfiguration。

list-bulk-deployment-detailed-reports

以下代码示例演示了如何使用 list-bulk-deployment-detailed-reports。

Amazon CLI

列出有关批量部署中各个部署的信息

以下 `list-bulk-deployment-detailed-reports` 示例显示有关批量部署操作中各个部署的信息，包括状态。

```
aws greengrass list-bulk-deployment-detailed-reports \
    --bulk-deployment-id 42ce9c42-489b-4ed4-b905-8996aa50ef9d
```

输出：

```
{
    "Deployments": [
        {
            "DeploymentType": "NewDeployment",
            "DeploymentStatus": "Success",
            "DeploymentId": "123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/
deployments/123456789012:123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/
versions/123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "CreatedAt": "2020-01-21T21:34:16.501Z"
        },
        {
            "DeploymentType": "NewDeployment",
            "DeploymentStatus": "InProgress",
            "DeploymentId": "123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE55555/
deployments/123456789012:123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE55555/versions/a1b2c3d4-5678-90ab-cdef-
EXAMPLE66666",
            "CreatedAt": "2020-01-21T21:34:16.486Z"
        },
        ...
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的为组创建批量部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBulkDeploymentDetailedReports。

list-bulk-deployments

以下代码示例演示了如何使用 list-bulk-deployments。

Amazon CLI

列出批量部署

以下 list-bulk-deployments 示例列出所有批量部署。

```
aws greengrass list-bulk-deployments
```

输出：

```
{
    "BulkDeployments": [
        {
            "BulkDeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/bulk/deployments/870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
            "BulkDeploymentId": "870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
            "CreatedAt": "2019-06-25T16:11:33.265Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的为组创建批量部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBulkDeployments。

list-connector-definition-versions

以下代码示例演示了如何使用 list-connector-definition-versions。

Amazon CLI

列出可用于连接器定义的版本

以下 list-connector-definition-versions 示例列出可用于指定连接器定义的版本。使用 list-connector-definitions 命令获取连接器定义 ID。

```
aws greengrass list-connector-definition-versions \
    --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-
c7c2-4a26-a7e2-7bf478ea2623",
            "CreationTimestamp": "2019-06-19T19:30:01.300Z",
            "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
            "Version": "63c57963-c7c2-4a26-a7e2-7bf478ea2623"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Greengrass 连接器与服务和协议集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConnectorDefinitionVersions。

`list-connector-definitions`

以下代码示例演示了如何使用 `list-connector-definitions`。

Amazon CLI

列出定义的 Greengrass 连接器

以下 `list-connector-definitions` 示例列出为您的 Amazon 账户定义的所有 Greengrass 连接器。

```
aws greengrass list-connector-definitions
```

输出：

```
{
    "Definitions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
            "CreationTimestamp": "2019-06-19T19:30:01.300Z",
            "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
```

```
            "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
            "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/
versions/63c57963-c7c2-4a26-a7e2-7bf478ea2623",
            "Name": "MySNSConnector"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Greengrass 连接器与服务和协议集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConnectorDefinitions。

`list-core-definition-versions`

以下代码示例演示了如何使用 `list-core-definition-versions`。

Amazon CLI

列出 Greengrass 核心定义的版本

以下 `list-core-definitions` 示例列出指定 Greengrass 核心定义的所有版本。您可以使用 `list-core-definitions` 命令获取版本 ID。

```
aws greengrass list-core-definition-versions \
    --core-definition-id "eaf280cb-138c-4d15-af36-6f681a1348f7"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7/versions/467c36e4-c5da-440c-
a97b-084e62593b4c",
            "CreationTimestamp": "2019-06-18T16:14:17.709Z",
            "Id": "eaf280cb-138c-4d15-af36-6f681a1348f7",
            "Version": "467c36e4-c5da-440c-a97b-084e62593b4c"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCoreDefinitionVersions。

list-core-definitions

以下代码示例演示了如何使用 list-core-definitions。

Amazon CLI

列出 Greengrass 核心定义

以下 list-core-definitions 示例列出您的 Amazon 账户的所有 Greengrass 核心定义。

```
aws greengrass list-core-definitions
```

输出：

```
{
    "Definitions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/0507843c-c1ef-4f06-b051-817030df7e7d",
            "CreationTimestamp": "2018-10-17T04:30:32.786Z",
            "Id": "0507843c-c1ef-4f06-b051-817030df7e7d",
            "LastUpdatedTimestamp": "2018-10-17T04:30:32.786Z",
            "LatestVersion": "bcdf9e86-3793-491e-93af-3cdfbf4e22b7",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/0507843c-c1ef-4f06-b051-817030df7e7d/versions/
bcdf9e86-3793-491e-93af-3cdfbf4e22b7"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/31c22500-3509-4271-bafd-cf0655cda438",
            "CreationTimestamp": "2019-06-18T16:24:16.064Z",
            "Id": "31c22500-3509-4271-bafd-cf0655cda438",
            "LastUpdatedTimestamp": "2019-06-18T16:24:16.064Z",
            "LatestVersion": "2f350395-6d09-4c8a-8336-9ae5b57ace84",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/31c22500-3509-4271-bafd-cf0655cda438/
versions/2f350395-6d09-4c8a-8336-9ae5b57ace84"
        },
```

```
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46",
            "CreationTimestamp": "2019-06-18T16:21:21.351Z",
            "Id": "c906ed39-a1e3-4822-a981-7b9bd57b4b46",
            "LastUpdatedTimestamp": "2019-06-18T16:21:21.351Z",
            "LatestVersion": "42aeeac3-fd9d-4312-a8fd-ffa9404a20e0",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeeac3-
fd9d-4312-a8fd-ffa9404a20e0"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7",
            "CreationTimestamp": "2019-06-18T16:14:17.709Z",
            "Id": "eaf280cb-138c-4d15-af36-6f681a1348f7",
            "LastUpdatedTimestamp": "2019-06-18T16:14:17.709Z",
            "LatestVersion": "467c36e4-c5da-440c-a97b-084e62593b4c",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7/versions/467c36e4-
c5da-440c-a97b-084e62593b4c"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCoreDefinitions。

`list-deployments`

以下代码示例演示了如何使用 `list-deployments`。

Amazon CLI

列出 Greengrass 组的部署

以下 `list-deployments` 示例列出指定 Greengrass 组的部署。您可以使用 `list-groups` 命令查找您的组 ID。

```
aws greengrass list-deployments \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

输出：

```
{
    "Deployments": [
        {
            "CreatedAt": "2019-06-18T17:04:32.702Z",
            "DeploymentId": "1065b8a0-812b-4f21-9d5d-e89b232a530f",
            "DeploymentType": "NewDeployment",
            "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeployments。

list-device-definition-versions

以下代码示例演示了如何使用 list-device-definition-versions。

Amazon CLI

列出设备定义的版本

以下 list-device-definition-versions 示例显示与指定设备定义关联的设备定义版本。

```
aws greengrass list-device-definition-versions \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

输出：

```
{
    "Versions": [
        {
            "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
            "CreationTimestamp": "2019-09-11T00:15:09.838Z",
            "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71"
        },
        {
```

```
            "Version": "3b5cc510-58c1-44b5-9d98-4ad858ffa795",
            "CreationTimestamp": "2019-09-11T00:11:06.197Z",
            "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/3b5cc510-58c1-44b5-9d98-4ad858ffa795"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeviceDefinitionVersions。

list-device-definitions

以下代码示例演示了如何使用 list-device-definitions。

Amazon CLI

列出设备定义

以下 list-device-definitions 示例显示有关指定 Amazon 区域中 Amazon 账户中的设备定义的详细信息。

```
aws greengrass list-device-definitions \
    --region us-west-2
```

输出：

```
{
    "Definitions": [
        {
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/50f3274c-3f0a-4f57-b114-6f46085281ab/versions/
c777b0f5-1059-449b-beaa-f003ebc56c34",
            "LastUpdatedTimestamp": "2019-06-14T15:42:09.059Z",
            "LatestVersion": "c777b0f5-1059-449b-beaa-f003ebc56c34",
            "CreationTimestamp": "2019-06-14T15:42:09.059Z",
            "Id": "50f3274c-3f0a-4f57-b114-6f46085281ab",
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/devices/50f3274c-3f0a-4f57-b114-6f46085281ab"
        },
        {
```

```
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/e01951c9-6134-479a-969a-1a15cac11c40/
versions/514d57aa-4ee6-401c-9fac-938a9f7a51e5",
            "Name": "TestDeviceDefinition",
            "LastUpdatedTimestamp": "2019-04-16T23:17:43.245Z",
            "LatestVersion": "514d57aa-4ee6-401c-9fac-938a9f7a51e5",
            "CreationTimestamp": "2019-04-16T23:17:43.245Z",
            "Id": "e01951c9-6134-479a-969a-1a15cac11c40",
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/devices/e01951c9-6134-479a-969a-1a15cac11c40"
        },
        {
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",
            "Name": "TemperatureSensors",
            "LastUpdatedTimestamp": "2019-09-10T00:19:03.698Z",
            "LatestVersion": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
            "CreationTimestamp": "2019-09-11T00:11:06.197Z",
            "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeviceDefinitions。

list-function-definition-versions

以下代码示例演示了如何使用 list-function-definition-versions。

Amazon CLI

列出 Lambda 函数的版本

以下 list-function-definition-versions 示例列出指定 Lambda 函数的所有版本。您可以使用 list-function-definitions 命令获取 ID。

```
aws greengrass list-function-definition-versions \
    --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",
            "CreationTimestamp": "2019-06-18T17:04:30.776Z",
            "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "Version": "9748fda7-1589-4fcc-ac94-f5559e88678b"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9b08df77-26f2-4c29-93d2-769715edcfec",
            "CreationTimestamp": "2019-06-18T17:02:44.087Z",
            "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "Version": "9b08df77-26f2-4c29-93d2-769715edcfec"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/4236239f-94f7-4b90-a2f8-2a24c829d21e",
            "CreationTimestamp": "2019-06-18T17:01:42.284Z",
            "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "Version": "4236239f-94f7-4b90-a2f8-2a24c829d21e"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/343408bb-549a-4fbe-b043-853643179a39",
            "CreationTimestamp": "2019-06-18T16:21:21.431Z",
            "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "Version": "343408bb-549a-4fbe-b043-853643179a39"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFunctionDefinitionVersions。

list-function-definitions

以下代码示例演示了如何使用 list-function-definitions。

Amazon CLI

列出 Lambda 函数

以下 `list-function-definitions` 示例列出为您的 Amazon 账户定义的所有 Lambda 函数。

```
aws greengrass list-function-definitions
```

输出：

```
{
    "Definitions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/017970a5-8952-46dd-b1c1-020b3ae8e960",
            "CreationTimestamp": "2018-10-17T04:30:32.884Z",
            "Id": "017970a5-8952-46dd-b1c1-020b3ae8e960",
            "LastUpdatedTimestamp": "2018-10-17T04:30:32.884Z",
            "LatestVersion": "4380b302-790d-4ed8-92bf-02e88afecb15",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/017970a5-8952-46dd-b1c1-020b3ae8e960/
versions/4380b302-790d-4ed8-92bf-02e88afecb15"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "CreationTimestamp": "2019-06-18T16:21:21.431Z",
            "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
            "LastUpdatedTimestamp": "2019-06-18T16:21:21.431Z",
            "LatestVersion": "9748fda7-1589-4fcc-ac94-f5559e88678b",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/6598e653-a262-440c-9967-e2697f64da7b",
            "CreationTimestamp": "2019-06-18T16:24:16.123Z",
            "Id": "6598e653-a262-440c-9967-e2697f64da7b",
            "LastUpdatedTimestamp": "2019-06-18T16:24:16.123Z",
            "LatestVersion": "38bc6ccd-98a2-4ce7-997e-16c84748fae4",
```

```
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/6598e653-a262-440c-9967-e2697f64da7b/
versions/38bc6ccd-98a2-4ce7-997e-16c84748fae4"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/c668df84-fad2-491b-95f4-655d2cad7885",
            "CreationTimestamp": "2019-06-18T16:14:17.784Z",
            "Id": "c668df84-fad2-491b-95f4-655d2cad7885",
            "LastUpdatedTimestamp": "2019-06-18T16:14:17.784Z",
            "LatestVersion": "37dd68c4-a64f-40ba-aa13-71fecc3ebded",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/c668df84-fad2-491b-95f4-655d2cad7885/
versions/37dd68c4-a64f-40ba-aa13-71fecc3ebded"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFunctionDefinitions。

list-group-certificate-authorities

以下代码示例演示了如何使用 list-group-certificate-authorities。

Amazon CLI

列出组的当前 CA

以下 list-group-certificate-authorities 示例列出指定 Greengrass 组的当前证书颁发机构（CA）。

```
aws greengrass list-group-certificate-authorities \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

输出：

```
{
    "GroupCertificateAuthorities": [
        {
            "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-
west-2:123456789012:/greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/
```

```
certificateauthorities/
f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",
            "GroupCertificateAuthorityId":
 "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupCertificateAuthorities。

`list-group-versions`

以下代码示例演示了如何使用 `list-group-versions`。

Amazon CLI

列出 Greengrass 组的版本

以下 `list-group-versions` 示例列出指定 Greengrass 组的版本。

```
aws greengrass list-group-versions \
    --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
            "CreationTimestamp": "2019-06-18T17:04:30.915Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "Version": "115136b3-cfd7-4462-b77f-8741a4b00e5e"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/versions/4340669d-
d14d-44e3-920c-46c928750750",
            "CreationTimestamp": "2019-06-18T17:03:52.663Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "Version": "4340669d-d14d-44e3-920c-46c928750750"
```

```
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/
versions/1b06e099-2d5b-4f10-91b9-78c4e060f5da",
            "CreationTimestamp": "2019-06-18T17:02:44.189Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "Version": "1b06e099-2d5b-4f10-91b9-78c4e060f5da"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/2d3f27f1-3b43-4554-
ab7a-73ec30477efe",
            "CreationTimestamp": "2019-06-18T17:01:42.401Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "Version": "2d3f27f1-3b43-4554-ab7a-73ec30477efe"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/d20f7ae9-3444-4c1c-b025-
e2ede23cdd31",
            "CreationTimestamp": "2019-06-18T16:21:21.457Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "Version": "d20f7ae9-3444-4c1c-b025-e2ede23cdd31"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupVersions。

`list-groups`

以下代码示例演示了如何使用 `list-groups`。

Amazon CLI

列出 Greengrass 组

以下 `list-groups` 示例列出您的 Amazon 账户中定义的所有 Greengrass 组。

```
aws greengrass list-groups
```

输出：

```
{
    "Groups": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731",
            "CreationTimestamp": "2019-06-18T16:21:21.457Z",
            "Id": "1013db12-8b58-45ff-acc7-704248f66731",
            "LastUpdatedTimestamp": "2019-06-18T16:21:21.457Z",
            "LatestVersion": "115136b3-cfd7-4462-b77f-8741a4b00e5e",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
            "Name": "GGGroup4Pi3"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8",
            "CreationTimestamp": "2018-10-31T21:52:46.603Z",
            "Id": "1402daf9-71cf-4cfe-8be0-d5e80526d0d8",
            "LastUpdatedTimestamp": "2018-10-31T21:52:46.603Z",
            "LatestVersion": "749af901-60ab-456f-a096-91b12d983c29",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8/versions/749af901-60ab-456f-
a096-91b12d983c29",
            "Name": "MyTestGroup"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/504b5c8d-bbed-4635-aff1-48ec5b586db5",
            "CreationTimestamp": "2018-12-31T21:39:36.771Z",
            "Id": "504b5c8d-bbed-4635-aff1-48ec5b586db5",
            "LastUpdatedTimestamp": "2018-12-31T21:39:36.771Z",
            "LatestVersion": "46911e8e-f9bc-4898-8b63-59c7653636ec",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/504b5c8d-bbed-4635-aff1-48ec5b586db5/versions/46911e8e-
f9bc-4898-8b63-59c7653636ec",
            "Name": "smp-ggrass-group"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

list-logger-definition-versions

以下代码示例演示了如何使用 list-logger-definition-versions。

Amazon CLI

获取记录器定义版本的列表

以下 list-logger-definition-versions 示例获取指定记录器定义的所有版本的列表。

```
aws greengrass list-logger-definition-versions \
    --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
            "CreationTimestamp": "2019-05-08T16:10:13.866Z",
            "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
            "Version": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/3ec6d3af-eb85-48f9-
a16d-1c795fe696d7",
            "CreationTimestamp": "2019-05-08T16:10:13.809Z",
            "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
            "Version": "3ec6d3af-eb85-48f9-a16d-1c795fe696d7"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLoggerDefinitionVersions。

list-logger-definitions

以下代码示例演示了如何使用 list-logger-definitions。

Amazon CLI

获取记录器定义的列表

以下 `list-logger-definitions` 示例列出您的 Amazon 账户的所有记录器定义。

```
aws greengrass list-logger-definitions
```

输出：

```
{
    "Definitions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23",
            "CreationTimestamp": "2019-05-08T16:10:13.809Z",
            "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
            "LastUpdatedTimestamp": "2019-05-08T16:10:13.809Z",
            "LatestVersion": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/
versions/5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLoggerDefinitions。

list-resource-definition-versions

以下代码示例演示了如何使用 `list-resource-definition-versions`。

Amazon CLI

列出资源定义的版本

以下 `list-resource-definition-versions` 示例列出指定 Greengrass 资源的版本。

```
aws greengrass list-resource-definition-versions \
    --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
            "CreationTimestamp": "2019-06-19T16:40:59.392Z",
            "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
            "Version": "26e8829a-491a-464d-9c87-664bf6f6f2be"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/432d92f6-12de-4ec9-a704-619a942a62aa",
            "CreationTimestamp": "2019-06-19T16:40:59.261Z",
            "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
            "Version": "432d92f6-12de-4ec9-a704-619a942a62aa"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceDefinitionVersions。

`list-resource-definitions`

以下代码示例演示了如何使用 `list-resource-definitions`。

Amazon CLI

列出定义的资源

以下 `list-resource-definitions` 示例列出为要使用的 Amazon IoT Greengrass 定义的资源。

```
aws greengrass list-resource-definitions
```

输出：

```
{
    "Definitions": [
        {
```

```
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658",
            "CreationTimestamp": "2019-06-19T16:40:59.261Z",
            "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
            "LastUpdatedTimestamp": "2019-06-19T16:40:59.261Z",
            "LatestVersion": "26e8829a-491a-464d-9c87-664bf6f6f2be",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
            "CreationTimestamp": "2019-06-19T21:51:28.212Z",
            "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
            "LastUpdatedTimestamp": "2019-06-19T21:51:28.212Z",
            "LatestVersion": "a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/
a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",
            "Name": "MyGreengrassResources"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceDefinitions。

list-subscription-definition-versions

以下代码示例演示了如何使用 list-subscription-definition-versions。

Amazon CLI

列出订阅定义的版本

以下 list-subscription-definition-versions 示例列出指定订阅的所有版本。您可以使用 list-subscription-definitions 命令来查找订阅 ID。

```
aws greengrass list-subscription-definition-versions \
    --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152"
```

输出：

```
{
    "Versions": [
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b",
            "CreationTimestamp": "2019-06-18T17:03:52.499Z",
            "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
            "Version": "88ae8699-12ac-4663-ba3f-4d7f0519140b"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/versions/7e320ba3-
c369-4069-a2f0-90acb7f219d6",
            "CreationTimestamp": "2019-06-18T17:03:52.392Z",
            "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
            "Version": "7e320ba3-c369-4069-a2f0-90acb7f219d6"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSubscriptionDefinitionVersions。

`list-subscription-definitions`

以下代码示例演示了如何使用 `list-subscription-definitions`。

Amazon CLI

获取列表订阅定义

以下 `list-subscription-definitions` 示例列出您的 Amazon 账户中定义的所有 Amazon IoT Greengrass 订阅。

```
aws greengrass list-subscription-definitions
```

输出：

```
{
    "Definitions": [
        {
```

```
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152",
            "CreationTimestamp": "2019-06-18T17:03:52.392Z",
            "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
            "LastUpdatedTimestamp": "2019-06-18T17:03:52.392Z",
            "LatestVersion": "88ae8699-12ac-4663-ba3f-4d7f0519140b",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/cd6f1c37-d9a4-4e90-be94-01a7404f5967",
            "CreationTimestamp": "2018-10-18T15:45:34.024Z",
            "Id": "cd6f1c37-d9a4-4e90-be94-01a7404f5967",
            "LastUpdatedTimestamp": "2018-10-18T15:45:34.024Z",
            "LatestVersion": "d1cf8fac-284f-4f6a-98fe-a2d36d089373",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/cd6f1c37-d9a4-4e90-be94-01a7404f5967/versions/
d1cf8fac-284f-4f6a-98fe-a2d36d089373"
        },
        {
            "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/fa81bc84-3f59-4377-a84b-5d0134da359b",
            "CreationTimestamp": "2018-10-22T17:09:31.429Z",
            "Id": "fa81bc84-3f59-4377-a84b-5d0134da359b",
            "LastUpdatedTimestamp": "2018-10-22T17:09:31.429Z",
            "LatestVersion": "086d1b08-b25a-477c-a16f-6f9b3a9c295a",
            "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/fa81bc84-3f59-4377-a84b-5d0134da359b/
versions/086d1b08-b25a-477c-a16f-6f9b3a9c295a"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSubscriptionDefinitions。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出附加到资源的标签

以下 `list-tags-for-resource` 示例列出附加到指定资源的标签及其值。

```
aws greengrass list-tags-for-resource \
    --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

输出：

```
{
    "tags": {
        "ResourceSubType": "USB",
        "ResourceType": "Device"
    }
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的标记您的 Greengrass 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

reset-deployments

以下代码示例演示了如何使用 `reset-deployments`。

Amazon CLI

清理 Greengrass 组的部署信息

以下 `reset-deployments` 示例清理指定 Greengrass 组的部署信息。添加 `--force option` 时，部署信息将重置，而无需等待核心设备响应。

```
aws greengrass reset-deployments \
    --group-id "1402daf9-71cf-4cfe-8be0-d5e80526d0d8" \
    --force
```

输出：

```
{
```

```
    "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8/
deployments/7dd4e356-9882-46a3-9e28-6d21900c011a",
    "DeploymentId": "7dd4e356-9882-46a3-9e28-6d21900c011a"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的重置部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetDeployments。

start-bulk-deployment

以下代码示例演示了如何使用 start-bulk-deployment。

Amazon CLI

启动批量部署操作

以下 start-bulk-deployment 示例启动批量部署操作，使用存储在 S3 存储桶中的文件来指定要部署的组。

```
aws greengrass start-bulk-deployment \
    --cli-input-json "{\"InputFileUri\":\"https://gg-group-deployment1.s3-us-
west-2.amazonaws.com/MyBulkDeploymentInputFile.txt\", \"ExecutionRoleArn\":
\"arn:aws:iam::123456789012:role/ggCreateDeploymentRole\",\"AmznClientToken\":
\"yourAmazonClientToken\"}"
```

输出：

```
{
    "BulkDeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
bulk/deployments/870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
    "BulkDeploymentId": "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的为组创建批量部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartBulkDeployment。

stop-bulk-deployment

以下代码示例演示了如何使用 stop-bulk-deployment。

Amazon CLI

停止批量部署

以下 stop-bulk-deployment 示例停止指定的批量部署。如果您尝试停止已完成的批量部署，您会收到错误：InvalidInputException: Cannot change state of finished execution.

```
aws greengrass stop-bulk-deployment \
    --bulk-deployment-id "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的为组创建批量部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopBulkDeployment。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签应用于资源

以下 tag-resource 示例将两个标签 ResourceType 和 ResourceSubType 应用于指定的 Greengrass 资源。此操作可以添加新的标签和值，或可以更新现有标签的值。使用 untag-resource 命令删除标签。

```
aws greengrass tag-resource \
    --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658" \
    --tags "ResourceType=Device,ResourceSubType=USB"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的标记您的 Greengrass 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签及其值

以下 untag-resource 示例从指定的 Greengrass 组中删除其键为 Category 的标签。如果指定资源不存在键 Category，则不会返回错误。

```
aws greengrass untag-resource \
    --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731" \
    --tag-keys "Category"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的标记您的 Greengrass 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-connectivity-info

以下代码示例演示了如何使用 update-connectivity-info。

Amazon CLI

更新 Greengrass 核心的连接信息

以下 update-connectivity-info 示例更改设备可用于连接到指定 Greengrass 核心的端点。连接信息是 IP 地址或域名的列表，以及相应的端口号和可选的客户定义元数据。当本地网络发生变更时，您可能需要更新连接信息。

```
aws greengrass update-connectivity-info \
    --thing-name "MyGroup_Core" \
    --connectivity-info "[{\"Metadata\":\"\",\"PortNumber\":8883,\"HostAddress\":
\"127.0.0.1\",\"Id\":\"localhost_127.0.0.1_0\"},{\"Metadata\":\"\",\"PortNumber
\":8883,\"HostAddress\":\"192.168.1.3\",\"Id\":\"localIP_192.168.1.3\"}]"
```

输出：

```
{
    "Version": "312de337-59af-4cf9-a278-2a23bd39c300"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConnectivityInfo。

update-connector-definition

以下代码示例演示了如何使用 update-connector-definition。

Amazon CLI

更新连接器定义的名称

以下 update-connector-definition 示例更新指定连接器定义的名称。如果要更新连接器的详细信息，请使用 create-connector-definition-version 命令创建新版本。

```
aws greengrass update-connector-definition \
    --connector-definition-id "55d0052b-0d7d-44d6-b56f-21867215e118" \
    --name "GreengrassConnectors2019"
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用连接器与服务和协议集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConnectorDefinition。

update-core-definition

以下代码示例演示了如何使用 update-core-definition。

Amazon CLI

更新核心定义

以下 update-core-definition 示例更改指定核心定义的名称。您只能更新核心定义的 name 属性。

```
aws greengrass update-core-definition \
```

```
    --core-definition-id "582efe12-b05a-409e-9a24-a2ba1bcc4a12" \
    --name "MyCoreDevices"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置 Amazon IoT Greengrass 核心。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCoreDefinition。

update-device-definition

以下代码示例演示了如何使用 update-device-definition。

Amazon CLI

更新设备定义

以下 update-device-definition 示例更改指定设备定义的名称。您只能更新设备定义的 name 属性。

```
aws greengrass update-device-definition \
    --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \
    --name "TemperatureSensors"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDeviceDefinition。

update-function-definition

以下代码示例演示了如何使用 update-function-definition。

Amazon CLI

更新函数定义的名称

以下 update-function-definition 示例更新指定函数定义的名称。如果要更新函数的详细信息，请使用 create-function-definition-version 命令创建新版本。

```
aws greengrass update-function-definition \
```

```
    --function-definition-id "e47952bd-dea9-4e2c-a7e1-37bbe8807f46" \
    --name ObsoleteFunction
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的运行本地 Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFunctionDefinition。

update-group-certificate-configuration

以下代码示例演示了如何使用 update-group-certificate-configuration。

Amazon CLI

更新组的证书的到期时间

以下 update-group-certificate-configuration 示例为面向指定组生成的证书设置为期 10 天的到期时间。

```
aws greengrass update-group-certificate-configuration \
    --group-id "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1" \
    --certificate-expiry-in-milliseconds 864000000
```

输出：

```
{
    "CertificateExpiryInMilliseconds": 864000000,
    "CertificateAuthorityExpiryInMilliseconds": 2524607999000,
    "GroupId": "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的 Amazon IoT Greengrass 安全性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroupCertificateConfiguration。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

更新组名称

以下 update-group 示例更新指定 Greengrass 组的名称。如果要更新组的详细信息，请使用 create-group-version 命令创建新版本。

```
aws greengrass update-group \
    --group-id "1402daf9-71cf-4cfe-8be0-d5e80526d0d8" \
    --name TestGroup4of6
```

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的在 Amazon IoT 上配置 Amazon IoT Greengrass。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

update-logger-definition

以下代码示例演示了如何使用 update-logger-definition。

Amazon CLI

更新记录器定义

以下 update-logger-definition 示例更改指定记录器定义的名称。您只能更新记录器定义的 name 属性。

```
aws greengrass update-logger-definition \
    --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0" \
    --name "LoggingConfigsForSensors"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Amazon IoT Greengrass 日志进行监控。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLoggerDefinition。

update-resource-definition

以下代码示例演示了如何使用 update-resource-definition。

Amazon CLI

更新资源定义的名称

以下 update-resource-definition 示例更新指定资源定义的名称。如果要更改资源的详细信息，请使用 create-resource-definition-version 命令创建新版本。

```
aws greengrass update-resource-definition \
    --resource-definition-id "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38" \
    --name GreengrassConnectorResources
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的使用 Lambda 函数和连接器访问本地资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResourceDefinition。

update-subscription-definition

以下代码示例演示了如何使用 update-subscription-definition。

Amazon CLI

更新订阅定义的名称

以下 update-subscription-definition 示例更新指定订阅定义的名称。如果要更改订阅的详细信息，请使用 create-subscription-definition-version 命令创建新版本。

```
aws greengrass update-subscription-definition \
    --subscription-definition-id "fa81bc84-3f59-4377-a84b-5d0134da359b" \
    --name "ObsoleteSubscription"
```

此命令不生成任何输出。

有关更多信息，请参阅《指南》中相应标题下的内容。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscriptionDefinition。

update-thing-runtime-configuration

以下代码示例演示了如何使用 update-thing-runtime-configuration。

Amazon CLI

在 Greengrass 核心的运行时配置中开启遥测

以下 `update-thing-runtime-configuration` 示例更新 Greengrass 核心的运行时配置以开启遥测。

```
aws greengrass update-thing-runtime-configuration \
    --thing-name SampleGreengrassCore \
    --telemetry-configuration {\"Telemetry\":\"On\"}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass 开发人员指南》中的配置遥测设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateThingRuntimeConfiguration。

Amazon IoT Greengrass V2使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Greengrass V2 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-service-role-to-account`

以下代码示例演示了如何使用 `associate-service-role-to-account`。

Amazon CLI

将 Greengrass 服务角色关联到 Amazon 账户

以下 associate-service-role-to-account 示例将服务角色与您的 Amazon 账户的 Amazon IoT Greengrass 相关联。

```
aws greengrassv2 associate-service-role-to-account \
    --role-arn arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole
```

输出：

```
{
    "associatedAt": "2022-01-19T19:21:53Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateServiceRoleToAccount。

batch-associate-client-device-with-core-device

以下代码示例演示了如何使用 batch-associate-client-device-with-core-device。

Amazon CLI

将客户端设备与核心设备相关联

以下 batch-associate-client-device-with-core-device 示例将两个客户端设备与核心设备相关联。

```
aws greengrassv2 batch-associate-client-device-with-core-device \
  --core-device-thing-name MyGreengrassCore \
  --entries thingName=MyClientDevice1 thingName=MyClientDevice2
```

输出：

```
{
    "errorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的与本地 IoT 设备交互。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchAssociateClientDeviceWithCoreDevice。

batch-disassociate-client-device-from-core-device

以下代码示例演示了如何使用 batch-disassociate-client-device-from-core-device。

Amazon CLI

取消客户端设备与核心设备的关联

以下 batch-disassociate-client-device-from-core-device 示例取消两个客户端设备与一个核心设备的关联。

```
aws greengrassv2 batch-disassociate-client-device-from-core-device \
  --core-device-thing-name MyGreengrassCore \
  --entries thingName=MyClientDevice1 thingName=MyClientDevice2
```

输出：

```
{
    "errorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的与本地 IoT 设备交互。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDisassociateClientDeviceFromCoreDevice。

cancel-deployment

以下代码示例演示了如何使用 cancel-deployment。

Amazon CLI

取消部署

以下 cancel-deployment 示例停止对事物组的持续部署。

```
aws greengrassv2 cancel-deployment \
    --deployment-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "message": "SUCCESS"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的取消部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelDeployment。

create-component-version

以下代码示例演示了如何使用 create-component-version。

Amazon CLI

示例 1：根据配方创建组件版本

以下 create-component-version 示例根据配方文件创建 Hello World 组件的一个版本。

```
aws greengrassv2 create-component-version \
    --inline-recipe fileb://com.example.HelloWorld-1.0.0.json
```

的内容com.example.HelloWorld-1.0.0.json：

```
{
    "RecipeFormatVersion": "2020-01-25",
    "ComponentName": "com.example.HelloWorld",
    "ComponentVersion": "1.0.0",
    "ComponentDescription": "My first AWS IoT Greengrass component.",
    "ComponentPublisher": "Amazon",
    "ComponentConfiguration": {
        "DefaultConfiguration": {
            "Message": "world"
        }
    },
    "Manifests": [
        {
            "Platform": {
                "os": "linux"
            },
            "Lifecycle": {
                "Run": "echo 'Hello {configuration:/Message}'"
            }
```

```
        }
    ]
}
```

输出：

```
{
    "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0",
    "componentName": "com.example.HelloWorld",
    "componentVersion": "1.0.0",
    "creationTimestamp": "2021-01-07T16:24:33.650000-08:00",
    "status": {
        "componentState": "REQUESTED",
        "message": "NONE",
        "errors": {}
    }
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的创建自定义组件和上传要部署的组件。

示例 2：通过 Amazon Lambda 函数创建组件版本

以下 `create-component-version` 示例通过 Amazon Lambda 函数创建 Hello World 组件的一个版本。

```
aws greengrassv2 create-component-version \
    --cli-input-json file://lambda-function-component.json
```

的内容`lambda-function-component.json`：

```
{
    "lambdaFunction": {
        "lambdaArn": "arn:aws:lambda:us-
west-2:123456789012:function:HelloWorldPythonLambda:1",
        "componentName": "com.example.HelloWorld",
        "componentVersion": "1.0.0",
        "componentLambdaParameters": {
            "eventSources": [
                {
```

```
                    "topic": "hello/world/+",
                    "type": "IOT_CORE"
                }
            ]
        }
    }
}
```

输出：

```
{
    "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0",
    "componentName": "com.example.HelloWorld",
    "componentVersion": "1.0.0",
    "creationTimestamp": "2021-01-07T17:05:27.347000-08:00",
    "status": {
        "componentState": "REQUESTED",
        "message": "NONE",
        "errors": {}
    }
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的运行 Amazon Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateComponentVersion。

create-deployment

以下代码示例演示了如何使用 create-deployment。

Amazon CLI

示例 1：创建部署

以下 create-deployment 示例将 Amazon IoT Greengrass 命令行界面部署到核心设备。

```
aws greengrassv2 create-deployment \
    --cli-input-json file://cli-deployment.json
```

的内容cli-deployment.json：

```
{
    "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/MyGreengrassCore",
    "deploymentName": "Deployment for MyGreengrassCore",
    "components": {
        "aws.greengrass.Cli": {
            "componentVersion": "2.0.3"
        }
    },
    "deploymentPolicies": {
        "failureHandlingPolicy": "DO_NOTHING",
        "componentUpdatePolicy": {
            "timeoutInSeconds": 60,
            "action": "NOTIFY_COMPONENTS"
        },
        "configurationValidationPolicy": {
            "timeoutInSeconds": 60
        }
    },
    "iotJobConfiguration": {}
}
```

输出：

```
{
    "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的创建部署。

示例 2：创建更新组件配置的部署

以下 `create-deployment` 示例将 Amazon IoT Greengrass 核心组件部署到一组核心设备。此部署对核心组件应用以下配置更新：

将目标设备的代理设置重置为其默认的无代理设置。将目标设备的 MQTT 设置重置为其默认值。设置核心的 JVM 的 JVM 选项。设置核心的日志记录级别。

```
aws greengrassv2 create-deployment \
    --cli-input-json file://nucleus-deployment.json
```

的内容`nucleus-deployment.json`：

```
{
    "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
MyGreengrassCoreGroup",
    "deploymentName": "Deployment for MyGreengrassCoreGroup",
    "components": {
        "aws.greengrass.Nucleus": {
            "componentVersion": "2.0.3",
            "configurationUpdate": {
                "reset": [
                    "/networkProxy",
                    "/mqtt"
                ],
                "merge": "{\"jvmOptions\":\"-Xmx64m\",\"logging\":{\"level\":\"WARN
\"}}"
            }
        }
    },
    "deploymentPolicies": {
        "failureHandlingPolicy": "ROLLBACK",
        "componentUpdatePolicy": {
            "timeoutInSeconds": 60,
            "action": "NOTIFY_COMPONENTS"
        },
        "configurationValidationPolicy": {
            "timeoutInSeconds": 60
        }
    },
    "iotJobConfiguration": {}
}
```

输出：

```
{
    "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的创建部署和更新组件配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeployment。

delete-component

以下代码示例演示了如何使用 delete-component。

Amazon CLI

删除组件版本

以下 delete-component 示例删除 Hello World 组件。

```
aws greengrassv2 delete-component \
    --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteComponent。

delete-core-device

以下代码示例演示了如何使用 delete-core-device。

Amazon CLI

删除核心设备

以下 delete-core-device 示例删除 Amazon IoT Greengrass 核心设备。

```
aws greengrassv2 delete-core-device \
    --core-device-thing-name MyGreengrassCore
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的卸载 Amazon IoT Greengrass 核心软件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCoreDevice。

describe-component

以下代码示例演示了如何使用 describe-component。

Amazon CLI

描述组件版本

以下 `describe-component` 示例描述 Hello World 组件。

```
aws greengrassv2 describe-component \
    --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0
```

输出：

```
{
    "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0",
    "componentName": "com.example.HelloWorld",
    "componentVersion": "1.0.0",
    "creationTimestamp": "2021-01-07T17:12:11.133000-08:00",
    "publisher": "Amazon",
    "description": "My first AWS IoT Greengrass component.",
    "status": {
        "componentState": "DEPLOYABLE",
        "message": "NONE",
        "errors": {}
    },
    "platforms": [
        {
            "attributes": {
                "os": "linux"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeComponent。

`disassociate-service-role-from-account`

以下代码示例演示了如何使用 `disassociate-service-role-from-account`。

Amazon CLI

取消 Greengrass 服务角色与 Amazon 账户的关联

以下 `disassociate-service-role-from-account` 示例取消 Greengrass 服务角色与您的 Amazon 账户的 Amazon IoT Greengrass 的关联。

```
aws greengrassv2 disassociate-service-role-from-account
```

输出：

```
{
    "disassociatedAt": "2022-01-19T19:26:09Z"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateServiceRoleFromAccount。

get-component-version-artifact

以下代码示例演示了如何使用 `get-component-version-artifact`。

Amazon CLI

获取用于下载组件构件的 URL

以下 `get-component-version-artifact` 示例获取用于下载本地调试控制台组件的 JAR 文件的 URL。

```
aws greengrassv2 get-component-version-artifact \
    --arn arn:aws:greengrass:us-
west-2:aws:components:aws.greengrass.LocalDebugConsole:versions:2.0.3 \
    --artifact-name "Uvt6ZEzQ9TKiAuLbfXBX_APdY0TWks3uc46tHFHTzBM=/
aws.greengrass.LocalDebugConsole.jar"
```

输出：

```
{
    "preSignedUrl": "https://evergreencomponentmanageme-
artifactbucket7410c9ef-g18n1iya8kwr.s3.us-west-2.amazonaws.com/public/
```

```
aws.greengrass.LocalDebugConsole/2.0.3/s3/ggv2-component-releases-prod-pdx/
EvergreenHttpDebugView/2ffc496ba41b39568968b22c582b4714a937193ee7687a45527238e696672521/
aws.greengrass.LocalDebugConsole/aws.greengrass.LocalDebugConsole.jar?X-Amz-
Security-Token=KwfLKSdEXAMPLE..."
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComponentVersionArtifact。

`get-component`

以下代码示例演示了如何使用 `get-component`。

Amazon CLI

示例 1：以 YAML 格式（Linux、macOS 或 Unix）下载组件的配方

以下 `get-component` 示例将 Hello World 组件的配方下载到 YAML 格式的文件中。此命令执行以下操作：

使用 `--output` 和 `--query` 参数控制该命令的输出。这些参数从命令的输出中提取配方 blob。有关控制输出的更多信息，请参阅《Amazon 命令行界面用户指南》中的控制命令输出。使用 `base64` 实用工具。此实用工具将提取的 blob 解码为原始文本。由成功执行的 `get-component` 命令返回的 blob 是 base64 编码的文本。必须解码此 blob 才能获取原始文本。将解码后的文本保存到文件中。命令的最后一部分 (`> com.example.HelloWorld-1.0.0.json`) 将解码后的文本保存到文件中。

```
aws greengrassv2 get-component \
    --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0 \
    --recipe-output-format YAML \
    --query recipe \
    --output text | base64 --decode > com.example.HelloWorld-1.0.0.json
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

示例 2：以 YAML 格式（Windows CMD）下载组件的配方

以下 `get-component` 示例将 Hello World 组件的配方下载到 YAML 格式的文件中。此命令使用 `certutil` 实用工具。

```
aws greengrassv2 get-component ^
    --arn arn:aws:greengrass:us-
west-2:675946970638:components:com.example.HelloWorld:versions:1.0.0 ^
    --recipe-output-format YAML ^
    --query recipe ^
    --output text > com.example.HelloWorld-1.0.0.yaml.b64

certutil -
decode com.example.HelloWorld-1.0.0.yaml.b64 com.example.HelloWorld-1.0.0.yaml
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

示例 3：以 YAML 格式（Windows PowerShell）下载组件的配方

以下 `get-component` 示例将 Hello World 组件的配方下载到 YAML 格式的文件中。此命令使用 `certutil` 实用工具。

```
aws greengrassv2 get-component `
    --arn arn:aws:greengrass:us-
west-2:675946970638:components:com.example.HelloWorld:versions:1.0.0 `
    --recipe-output-format YAML `
    --query recipe `
    --output text > com.example.HelloWorld-1.0.0.yaml.b64

certutil -
decode com.example.HelloWorld-1.0.0.yaml.b64 com.example.HelloWorld-1.0.0.yaml
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComponent。

get-connectivity-info

以下代码示例演示了如何使用 `get-connectivity-info`。

Amazon CLI

获取 Greengrass 核心设备的连接信息

以下 `get-connectivity-info` 示例获取 Greengrass 核心设备的连接信息。客户端设备使用此信息连接到在此核心设备上运行的 MQTT 代理。

```
aws greengrassv2 get-connectivity-info \
    --thing-name MyGreengrassCore
```

输出：

```
{
    "connectivityInfo": [
        {
            "id": "localIP_192.0.2.0",
            "hostAddress": "192.0.2.0",
            "portNumber": 8883
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理核心设备端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConnectivityInfo。

get-core-device

以下代码示例演示了如何使用 get-core-device。

Amazon CLI

获取核心设备

以下 get-core-device 示例获取有关 Amazon IoT Greengrass 核心设备的信息。

```
aws greengrassv2 get-core-device \
    --core-device-thing-name MyGreengrassCore
```

输出：

```
{
    "coreDeviceThingName": "MyGreengrassCore",
    "coreVersion": "2.0.3",
    "platform": "linux",
    "architecture": "amd64",
    "status": "HEALTHY",
    "lastStatusUpdateTimestamp": "2021-01-08T04:57:58.838000-08:00",
    "tags": {}
```

```
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的检查核心设备状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCoreDevice。

get-deployment

以下代码示例演示了如何使用 get-deployment。

Amazon CLI

获取部署

以下 get-deployment 示例获取有关将 Amazon IoT Greengrass 核心组件部署到一组核心设备的信息。

```
aws greengrassv2 get-deployment \
    --deployment-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
MyGreengrassCoreGroup",
    "revisionId": "14",
    "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "deploymentName": "Deployment for MyGreengrassCoreGroup",
    "deploymentStatus": "ACTIVE",
    "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222",
    "components": {
        "aws.greengrass.Nucleus": {
            "componentVersion": "2.0.3",
            "configurationUpdate": {
                "merge": "{\"jvmOptions\":\"-Xmx64m\",\"logging\":{\"level\":\"WARN
\"}}",
                "reset": [
                    "/networkProxy",
                    "/mqtt"
                ]
```

```
            }
        }
    },
    "deploymentPolicies": {
        "failureHandlingPolicy": "ROLLBACK",
        "componentUpdatePolicy": {
            "timeoutInSeconds": 60,
            "action": "NOTIFY_COMPONENTS"
        },
        "configurationValidationPolicy": {
            "timeoutInSeconds": 60
        }
    },
    "iotJobConfiguration": {},
    "creationTimestamp": "2021-01-07T17:21:20.691000-08:00",
    "isLatestForTarget": false,
    "tags": {}
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的将组件部署至设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeployment。

get-service-role-for-account

以下代码示例演示了如何使用 get-service-role-for-account。

Amazon CLI

获取 Amazon 账户的 Greengrass 服务角色

以下 get-service-role-for-account 示例获取与您的 Amazon 账户的 Amazon IoT Greengrass 关联的服务角色。

```
aws greengrassv2 get-service-role-for-account
```

输出：

```
{
    "associatedAt": "2022-01-19T19:21:53Z",
    "roleArn": "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的 Greengrass 服务角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceRoleForAccount。

list-client-devices-associated-with-core-device

以下代码示例演示了如何使用 list-client-devices-associated-with-core-device。

Amazon CLI

列出与核心设备关联的客户端设备

以下 list-client-devices-associated-with-core-device 示例列出与核心设备关联的所有客户端设备。

```
aws greengrassv2 list-client-devices-associated-with-core-device \
  --core-device-thing-name MyTestGreengrassCore
```

输出：

```
{
    "associatedClientDevices": [
        {
            "thingName": "MyClientDevice2",
            "associationTimestamp": "2021-07-12T16:33:55.843000-07:00"
        },
        {
            "thingName": "MyClientDevice1",
            "associationTimestamp": "2021-07-12T16:33:55.843000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的与本地 IoT 设备交互。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListClientDevicesAssociatedWithCoreDevice。

list-component-versions

以下代码示例演示了如何使用 list-component-versions。

Amazon CLI

列出组件的版本

以下 `list-component-versions` 示例列出 Hello World 组件的所有版本。

```
aws greengrassv2 list-component-versions \
    --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld
```

输出：

```
{
    "componentVersions": [
        {
            "componentName": "com.example.HelloWorld",
            "componentVersion": "1.0.1",
            "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.1"
        },
        {
            "componentName": "com.example.HelloWorld",
            "componentVersion": "1.0.0",
            "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListComponentVersions。

`list-components`

以下代码示例演示了如何使用 `list-components`。

Amazon CLI

列出组件

以下 `list-components` 示例列出当前区域的 Amazon 账户中定义的每个组件及其最新版本。

```
aws greengrassv2 list-components
```

输出：

```
{
    "components": [
        {
            "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld",
            "componentName": "com.example.HelloWorld",
            "latestVersion": {
                "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.1",
                "componentVersion": "1.0.1",
                "creationTimestamp": "2021-01-08T16:51:07.352000-08:00",
                "description": "My first AWS IoT Greengrass component.",
                "publisher": "Amazon",
                "platforms": [
                    {
                        "attributes": {
                            "os": "linux"
                        }
                    }
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理组件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListComponents。

`list-core-devices`

以下代码示例演示了如何使用 `list-core-devices`。

Amazon CLI

列出核心设备

以下 `list-core-devices` 示例列出了当前区域的 Amazon 账户中的 Amazon IoT Greengrass 核心设备。

```
aws greengrassv2 list-core-devices
```

输出：

```
{
    "coreDevices": [
        {
            "coreDeviceThingName": "MyGreengrassCore",
            "status": "HEALTHY",
            "lastStatusUpdateTimestamp": "2021-01-08T04:57:58.838000-08:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的检查核心设备状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCoreDevices。

`list-deployments`

以下代码示例演示了如何使用 `list-deployments`。

Amazon CLI

列出部署

以下 `list-deployments` 示例列出了当前区域的 Amazon 账户中定义的每个部署的最新修订版。

```
aws greengrassv2 list-deployments
```

输出：

```
{
    "deployments": [
        {
            "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
MyGreengrassCoreGroup",
            "revisionId": "14",
```

```
            "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "deploymentName": "Deployment for MyGreengrassCoreGroup",
            "creationTimestamp": "2021-01-07T17:21:20.691000-08:00",
            "deploymentStatus": "ACTIVE",
            "isLatestForTarget": false
        },
        {
            "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/
MyGreengrassCore",
            "revisionId": "1",
            "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "deploymentName": "Deployment for MyGreengrassCore",
            "creationTimestamp": "2021-01-06T16:10:42.407000-08:00",
            "deploymentStatus": "COMPLETED",
            "isLatestForTarget": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的将组件部署至设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeployments。

`list-effective-deployments`

以下代码示例演示了如何使用 `list-effective-deployments`。

Amazon CLI

列出部署作业

以下 `list-effective-deployments` 示例列出了适用于 Amazon IoT Greengrass 核心设备的部署。

```
aws greengrassv2 list-effective-deployments \
    --core-device-thing-name MyGreengrassCore
```

输出：

```
{
    "effectiveDeployments": [
        {
```

```
            "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "deploymentName": "Deployment for MyGreengrassCore",
            "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/
MyGreengrassCore",
            "coreDeviceExecutionStatus": "COMPLETED",
            "reason": "SUCCESSFUL",
            "creationTimestamp": "2021-01-06T16:10:42.442000-08:00",
            "modifiedTimestamp": "2021-01-08T17:21:27.830000-08:00"
        },
        {
            "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "deploymentName": "Deployment for MyGreengrassCoreGroup",
            "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
            "iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-
cdef-EXAMPLE44444",
            "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
MyGreengrassCoreGroup",
            "coreDeviceExecutionStatus": "SUCCEEDED",
            "reason": "SUCCESSFUL",
            "creationTimestamp": "2021-01-07T17:19:20.394000-08:00",
            "modifiedTimestamp": "2021-01-07T17:21:20.721000-08:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的检查核心设备状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEffectiveDeployments。

`list-installed-components`

以下代码示例演示了如何使用 `list-installed-components`。

Amazon CLI

列出安装在核心设备上的组件

以下 `list-installed-components` 示例列出了安装在 Amazon IoT Greengrass 核心设备上的组件。

```
aws greengrassv2 list-installed-components \
    --core-device-thing-name MyGreengrassCore
```

输出：

```
{
    "installedComponents": [
        {
            "componentName": "aws.greengrass.Cli",
            "componentVersion": "2.0.3",
            "lifecycleState": "RUNNING",
            "isRoot": true
        },
        {
            "componentName": "aws.greengrass.Nucleus",
            "componentVersion": "2.0.3",
            "lifecycleState": "FINISHED",
            "isRoot": true
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的检查核心设备状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInstalledComponents。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出了 Amazon IoT Greengrass 核心设备的所有标签。

```
aws greengrassv2 list-tags-for-resource \
    --resource-arn arn:aws:greengrass:us-
west-2:123456789012:coreDevices:MyGreengrassCore
```

输出：

```
{
    "tags": {
        "Owner": "richard-roe"
    }
```

```
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

将标签添加到资源中

以下 `tag-resource` 示例将所有者标签添加到 Amazon IoT Greengrass 核心设备。您可以使用此标签根据核心设备的所有者来控制对核心设备的访问权限。

```
aws greengrassv2 tag-resource \
    --resource-arn arn:aws:greengrass:us-
west-2:123456789012:coreDevices:MyGreengrassCore \
    --tags Owner=richard-roe
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源中删除标签

以下 `untag-resource` 示例从 Amazon IoT Greengrass 核心设备中删除所有者标签。

```
aws iotsitewise untag-resource \
    --resource-arn arn:aws:greengrass:us-
west-2:123456789012:coreDevices:MyGreengrassCore \
    --tag-keys Owner
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-connectivity-info

以下代码示例演示了如何使用 update-connectivity-info。

Amazon CLI

更新 Greengrass 核心设备的连接信息

以下 update-connectivity-info 示例获取 Greengrass 核心设备的连接信息。客户端设备使用此信息连接到在此核心设备上运行的 MQTT 代理。

```
aws greengrassv2 update-connectivity-info \
    --thing-name MyGreengrassCore \
    --cli-input-json file://core-device-connectivity-info.json
```

的内容core-device-connectivity-info.json：

```
{
    "connectivityInfo": [
        {
            "hostAddress": "192.0.2.0",
            "portNumber": 8883,
            "id": "localIP_192.0.2.0"
        }
    ]
}
```

输出：

```
{
    "version": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon IoT Greengrass V2 开发人员指南》中的管理核心设备端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConnectivityInfo。

使用 Amazon CLI 的 Amazon IoT Jobs SDK 发布 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Jobs SDK 发布 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-job-execution`

以下代码示例演示了如何使用 `describe-job-execution`。

Amazon CLI

获取作业执行的详细信息

以下 `describe-job-execution` 示例检索指定作业和事物最后一次执行的详细信息。

```
aws iot-jobs-data describe-job-execution \
    --job-id SampleJob \
    --thing-name MotionSensor1 \
    --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

输出：

```
{
    "execution": {
        "approximateSecondsBeforeTimedOut": 88,
        "executionNumber": 2939653338,
        "jobId": "SampleJob",
        "lastUpdatedAt": 1567701875.743,
        "queuedAt": 1567701902.444,
        "status": "QUEUED",
```

```
        "thingName": "MotionSensor1 ",
        "versionNumber": 3
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备和作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJobExecution。

get-pending-job-executions

以下代码示例演示了如何使用 get-pending-job-executions。

Amazon CLI

获取一个事物未处于终止状态的所有作业列表

以下 get-pending-job-executions 示例显示指定事物未处于终止状态的所有作业列表。

```
aws iot-jobs-data get-pending-job-executions \
    --thing-name MotionSensor1
    --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

输出：

```
{
    "inProgressJobs": [
    ],
    "queuedJobs": [
        {
            "executionNumber": 2939653338,
            "jobId": "SampleJob",
            "lastUpdatedAt": 1567701875.743,
            "queuedAt": 1567701902.444,
            "versionNumber": 3
      }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备和作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPendingJobExecutions。

start-next-pending-job-execution

以下代码示例演示了如何使用 start-next-pending-job-execution。

Amazon CLI

获取并启动事物的下一个待处理作业执行

以下 start-next-pending-job-execution 示例检索并启动指定事物的下一个状态为 IN_PROGRESS 或 QUEUED 的作业执行。

```
aws iot-jobs-data start-next-pending-job-execution \
    --thing-name MotionSensor1
    --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

输出：

```
{
    "execution": {
        "approximateSecondsBeforeTimedOut": 88,
        "executionNumber": 2939653338,
        "jobId": "SampleJob",
        "lastUpdatedAt": 1567714853.743,
        "queuedAt": 1567701902.444,
        "startedAt": 1567714871.690,
        "status": "IN_PROGRESS",
        "thingName": "MotionSensor1 ",
        "versionNumber": 3
   }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备和作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartNextPendingJobExecution。

update-job-execution

以下代码示例演示了如何使用 update-job-execution。

Amazon CLI

更新作业执行的状态

以下 update-job-execution 示例更新指定作业和事物的状态。

```
aws iot-jobs-data update-job-execution \
    --job-id SampleJob \
    --thing-name MotionSensor1 \
    --status REMOVED \
    --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

输出：

```
{
    "executionState": {
        "status": "REMOVED",
        "versionNumber": 3
    },
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的设备和作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJobExecution。

使用 Amazon CLI 的 Amazon IoT SiteWise 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT SiteWise 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

associate-assets

以下代码示例演示了如何使用 associate-assets。

Amazon CLI

将子资产与父资产关联

以下 associate-assets 示例将一个风电涡轮机资产与一个风电场资产相关联，其中风电涡轮机资产模型作为一个层次结构存在于风电场资产模型中。

```
aws iotsitewise associate-assets \
    --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \
    --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE \
    --child-asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的关联资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAssets。

batch-associate-project-assets

以下代码示例演示了如何使用 batch-associate-project-assets。

Amazon CLI

将资产与项目关联

以下 batch-associate-project-assets 示例将风电场资产与项目相关联。

```
aws iotsitewise batch-associate-project-assets \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE \
    --asset-ids a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的向项目添加资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchAssociateProjectAssets。

batch-disassociate-project-assets

以下代码示例演示了如何使用 batch-disassociate-project-assets。

Amazon CLI

取消资产与项目的关联

以下 batch-disassociate-project-assets 示例取消风电场资产与项目的关联。

```
aws iotsitewise batch-disassociate-project-assets \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE \
    --asset-ids a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的向项目添加资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDisassociateProjectAssets。

batch-put-asset-property-value

以下代码示例演示了如何使用 batch-put-asset-property-value。

Amazon CLI

将数据发送到资产属性

以下 batch-put-asset-property-value 示例将功率和温度数据发送到由属性别名标识的资产属性。

```
aws iotsitewise batch-put-asset-property-value \
    --cli-input-json file://batch-put-asset-property-value.json
```

的内容batch-put-asset-property-value.json：

```
{
    "entries": [
        {
            "entryId": "1575691200-company-windfarm-3-turbine-7-power",
            "propertyAlias": "company-windfarm-3-turbine-7-power",
            "propertyValues": [
                {
                    "value": {
                        "doubleValue": 4.92
```

```
                    },
                    "timestamp": {
                        "timeInSeconds": 1575691200
                    },
                    "quality": "GOOD"
                }
            ]
        },
        {
            "entryId": "1575691200-company-windfarm-3-turbine-7-temperature",
            "propertyAlias": "company-windfarm-3-turbine-7-temperature",
            "propertyValues": [
                {
                    "value": {
                        "integerValue": 38
                    },
                    "timestamp": {
                        "timeInSeconds": 1575691200
                    }
                }
            ]
        }
    ]
}
```

输出：

```
{
    "errorEntries": []
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用 Amazon IoT SiteWise API 接收数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchPutAssetPropertyValue。

`create-access-policy`

以下代码示例演示了如何使用 `create-access-policy`。

Amazon CLI

示例 1：向用户授予对门户的管理权限

以下 create-access-policy 示例创建了一个访问策略，该策略向用户授予对风电场公司门户网站的管理访问权限。

```
aws iotsitewise create-access-policy \
    --cli-input-json file://create-portal-administrator-access-policy.json
```

的内容create-portal-administrator-access-policy.json：

```
{
    "accessPolicyIdentity": {
        "user": {
            "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
        }
    },
    "accessPolicyPermission": "ADMINISTRATOR",
    "accessPolicyResource": {
        "portal": {
            "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE"
        }
    }
}
```

输出：

```
{
    "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE",
    "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/
a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的添加或删除门户管理员。

示例 2：向用户授予对项目的只读访问权限

以下 create-access-policy 示例创建了一个访问策略，该策略向用户授予对风电场项目的只读访问权限。

```
aws iotsitewise create-access-policy \
    --cli-input-json file://create-project-viewer-access-policy.json
```

的内容create-project-viewer-access-policy.json：

```
{
    "accessPolicyIdentity": {
        "user": {
            "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
        }
    },
    "accessPolicyPermission": "VIEWER",
    "accessPolicyResource": {
        "project": {
            "id": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE"
        }
    }
}
```

输出：

```
{
    "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE",
    "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/
a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的分配项目查看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccessPolicy。

`create-asset-model`

以下代码示例演示了如何使用 `create-asset-model`。

Amazon CLI

创建资产模型

以下 `create-asset-model` 示例创建一个资产模型，该模型使用下列属性定义了一个风电涡轮机：

序列号 - 风电涡轮机的序列号；发电量 - 风电涡轮机的发电量数据流；温度 C - 以摄氏度计的风电涡轮机的温度数据流；温度 F - 从摄氏度映射到华氏度的温度数据点

```
aws iotsitewise create-asset-model \
    --cli-input-json file://create-wind-turbine-model.json
```

的内容create-wind-turbine-model.json：

```
{
    "assetModelName": "Wind Turbine Model",
    "assetModelDescription": "Represents a wind turbine",
    "assetModelProperties": [
        {
            "name": "Serial Number",
            "dataType": "STRING",
            "type": {
                "attribute": {}
            }
        },
        {
            "name": "Generated Power",
            "dataType": "DOUBLE",
            "unit": "kW",
            "type": {
                "measurement": {}
            }
        },
        {
            "name": "Temperature C",
            "dataType": "DOUBLE",
            "unit": "Celsius",
            "type": {
                "measurement": {}
            }
        },
        {
            "name": "Temperature F",
            "dataType": "DOUBLE",
            "unit": "Fahrenheit",
            "type": {
                "transform": {
                    "expression": "temp_c * 9 / 5 + 32",
                    "variables": [
                        {
                            "name": "temp_c",
                            "value": {
                                "propertyId": "Temperature C"
                            }
                        }
                    ]
```

```
                }
            }
        },
        {
            "name": "Total Generated Power",
            "dataType": "DOUBLE",
            "unit": "kW",
            "type": {
                "metric": {
                    "expression": "sum(power)",
                    "variables": [
                        {
                            "name": "power",
                            "value": {
                                "propertyId": "Generated Power"
                            }
                        }
                    ],
                    "window": {
                        "tumbling": {
                            "interval": "1h"
                        }
                    }
                }
            }
        }
    ]
}
```

输出：

```
{
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "assetModelStatus": {
        "state": "CREATING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的定义资产模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAssetModel。

create-asset

以下代码示例演示了如何使用 create-asset。

Amazon CLI

创建资产

以下 create-asset 示例根据风电涡轮机资产模型创建风电涡轮机资产。

```
aws iotsitewise create-asset \
    --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --asset-name "Wind Turbine 1"
```

输出：

```
{
    "assetId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
    "assetArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
    "assetStatus": {
        "state": "CREATING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的创建资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAsset。

create-dashboard

以下代码示例演示了如何使用 create-dashboard。

Amazon CLI

创建仪表板

以下 create-dashboard 示例创建了一个带有折线图的仪表板，该仪表板显示了风电场的总发电量。

```
aws iotsitewise create-dashboard \
```

```
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE \
    --dashboard-name "Wind Farm" \
    --dashboard-definition file://create-wind-farm-dashboard.json
```

的内容create-wind-farm-dashboard.json：

```
{
    "widgets": [
        {
            "type": "monitor-line-chart",
            "title": "Generated Power",
            "x": 0,
            "y": 0,
            "height": 3,
            "width": 3,
            "metrics": [
                {
                    "label": "Power",
                    "type": "iotsitewise",
                    "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
                    "propertyId": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE"
                }
            ]
        }
    ]
}
```

输出：

```
{
    "dashboardId": "a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
    "dashboardArn": "arn:aws:iotsitewise:us-west-2:123456789012:dashboard/
a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的创建仪表板（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDashboard。

create-gateway

以下代码示例演示了如何使用 create-gateway。

Amazon CLI

创建网关

以下 `create-gateway` 示例创建在 Amazon IoT Greengrass 上运行的网关。

```
aws iotsitewise create-gateway \
    --gateway-name ExampleCorpGateway \
    --gateway-platform greengrass={groupArn=arn:aws:greengrass:us-
west-2:123456789012:/greengrass/groups/a1b2c3d4-5678-90ab-cdef-1b1b1EXAMPLE}
```

输出：

```
{
    "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
    "gatewayArn": "arn:aws:iotsitewise:us-west-2:123456789012:gateway/
a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的配置网关。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGateway。

create-portal

以下代码示例演示了如何使用 `create-portal`。

Amazon CLI

创建门户

以下 `create-portal` 示例为一家风电场公司创建一个门户网站。您只能在启用了 Amazon 单点登录的同一区域创建门户。

```
aws iotsitewise create-portal \
    --portal-name WindFarmPortal \
    --portal-description "A portal that contains wind farm projects for Example
 Corp." \
    --portal-contact-email support@example.com \
    --role-arn arn:aws:iam::123456789012:role/service-role/
MySiteWiseMonitorServiceRole
```

输出：

```
{
    "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "portalArn": "arn:aws:iotsitewise:us-west-2:123456789012:portal/
a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "portalStartUrl": "https://a1b2c3d4-5678-90ab-cdef-
aaaaaEXAMPLE.app.iotsitewise.aws",
    "portalStatus": {
        "state": "CREATING"
    },
    "ssoApplicationId": "ins-a1b2c3d4-EXAMPLE"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的 Amazon IoT SiteWise Monitor 入门和《Amazon IoT SiteWise 用户指南》中的启用 Amazon SSO。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePortal。

create-project

以下代码示例演示了如何使用 create-project。

Amazon CLI

创建项目

以下 create-project 示例创建了一个风电场项目。

```
aws iotsitewise create-project \
    --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE \
    --project-name "Wind Farm 1" \
    --project-description "Contains asset visualizations for Wind Farm #1 for
 Example Corp."
```

输出：

```
{
    "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE",
    "projectArn": "arn:aws:iotsitewise:us-west-2:123456789012:project/
a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE"
```

```
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的创建项目。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProject。

`delete-access-policy`

以下代码示例演示了如何使用 `delete-access-policy`。

Amazon CLI

撤消用户对项目或门户的访问权限

以下 `delete-access-policy` 示例删除了一个访问策略，该策略向用户授予对某个门户的管理访问权限。

```
aws iotsitewise delete-access-policy \
    --access-policy-id a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的添加或删除门户管理员。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessPolicy。

`delete-asset-model`

以下代码示例演示了如何使用 `delete-asset-model`。

Amazon CLI

删除资产模型

以下 `delete-asset-model` 示例删除了一个风电涡轮机资产模型。

```
aws iotsitewise delete-asset-model \
    --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
```

```
    "assetModelStatus": {
        "state": "DELETING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的删除资产模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAssetModel。

`delete-asset`

以下代码示例演示了如何使用 `delete-asset`。

Amazon CLI

删除资产

以下 `delete-asset` 示例删除一个风电涡轮机资产。

```
aws iotsitewise delete-asset \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

输出：

```
{
    "assetStatus": {
        "state": "DELETING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的删除资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAsset。

`delete-dashboard`

以下代码示例演示了如何使用 `delete-dashboard`。

Amazon CLI

删除仪表板

以下 delete-dashboard 示例删除一个风电涡轮机仪表板。

```
aws iotsitewise delete-dashboard \
    --dashboard-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的删除仪表板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDashboard。

delete-gateway

以下代码示例演示了如何使用 delete-gateway。

Amazon CLI

删除网关

以下 delete-gateway 示例删除网关。

```
aws iotsitewise delete-gateway \
    --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用网关提取数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGateway。

delete-portal

以下代码示例演示了如何使用 delete-portal。

Amazon CLI

删除门户

以下 delete-portal 示例删除一家风电场公司的门户网站。

```
aws iotsitewise delete-portal \
```

```
    --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

输出：

```
{
    "portalStatus": {
        "state": "DELETING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的删除门户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePortal。

delete-project

以下代码示例演示了如何使用 delete-project。

Amazon CLI

删除项目

以下 delete-project 示例删除一个风电场项目。

```
aws iotsitewise delete-project \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的删除项目。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProject。

describe-access-policy

以下代码示例演示了如何使用 describe-access-policy。

Amazon CLI

描述访问策略

以下 describe-access-policy 示例描述一个访问策略，该策略向用户授予对风电场公司门户网站的管理访问权限。

```
aws iotsitewise describe-access-policy \
    --access-policy-id a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE
```

输出：

```
{
    "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE",
    "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/
a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE",
    "accessPolicyIdentity": {
        "user": {
            "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
        }
    },
    "accessPolicyResource": {
        "portal": {
            "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE"
        }
    },
    "accessPolicyPermission": "ADMINISTRATOR",
    "accessPolicyCreationDate": "2020-02-20T22:35:15.552880124Z",
    "accessPolicyLastUpdateDate": "2020-02-20T22:35:15.552880124Z"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的添加或删除门户管理员。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccessPolicy。

describe-asset-model

以下代码示例演示了如何使用 describe-asset-model。

Amazon CLI

描述资产模型

以下 describe-asset-model 示例描述一个风电场资产模型。

```
aws iotsitewise describe-asset-model \
```

```
    --asset-model-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

输出：

```
{
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetModelName": "Wind Farm Model",
    "assetModelDescription": "Represents a wind farm that comprises many wind
 turbines",
    "assetModelProperties": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
            "name": "Total Generated Power",
            "dataType": "DOUBLE",
            "unit": "kW",
            "type": {
                "metric": {
                    "expression": "sum(power)",
                    "variables": [
                        {
                            "name": "power",
                            "value": {
                                "propertyId": "a1b2c3d4-5678-90ab-
cdef-66666EXAMPLE",
                                "hierarchyId": "a1b2c3d4-5678-90ab-
cdef-77777EXAMPLE"
                            }
                        }
                    ],
                    "window": {
                        "tumbling": {
                            "interval": "1h"
                        }
                    }
                }
            }
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",
            "name": "Region",
            "dataType": "STRING",
```

```
            "type": {
                "attribute": {
                    "defaultValue": " "
                }
            }
        }
    ],
    "assetModelHierarchies": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
            "name": "Wind Turbines",
            "childAssetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
        }
    ],
    "assetModelCreationDate": 1575671284.0,
    "assetModelLastUpdateDate": 1575671988.0,
    "assetModelStatus": {
        "state": "ACTIVE"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的描述特定资产模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAssetModel。

describe-asset-property

以下代码示例演示了如何使用 describe-asset-property。

Amazon CLI

描述资产属性

以下 describe-asset-property 示例描述风电场资产的总发电量属性。

```
aws iotsitewise describe-asset-property \
    --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-99999EXAMPLE
```

输出：

```
{
```

```
    "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
    "assetName": "Wind Farm 1",
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetProperty": {
        "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
        "name": "Total Generated Power",
        "notification": {
            "topic": "$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE/assets/a1b2c3d4-5678-90ab-cdef-44444EXAMPLE/properties/
a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
            "state": "DISABLED"
        },
        "dataType": "DOUBLE",
        "unit": "kW",
        "type": {
            "metric": {
                "expression": "sum(power)",
                "variables": [
                    {
                        "name": "power",
                        "value": {
                            "propertyId": "a1b2c3d4-5678-90ab-cdef-66666EXAMPLE",
                            "hierarchyId": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE"
                        }
                    }
                ],
                "window": {
                    "tumbling": {
                        "interval": "1h"
                    }
                }
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的描述特定资产属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAssetProperty。

`describe-asset`

以下代码示例演示了如何使用 `describe-asset`。

Amazon CLI

描述资产

以下 describe-asset 示例描述风电场资产。

```
aws iotsitewise describe-asset \
    --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

输出：

```
{
    "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
    "assetArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
    "assetName": "Wind Farm 1",
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetProperties": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",
            "name": "Region",
            "dataType": "STRING"
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
            "name": "Total Generated Power",
            "dataType": "DOUBLE",
            "unit": "kW"
        }
    ],
    "assetHierarchies": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
            "name": "Wind Turbines"
        }
    ],
    "assetCreationDate": 1575672453.0,
    "assetLastUpdateDate": 1575672453.0,
    "assetStatus": {
        "state": "ACTIVE"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的描述特定资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAsset。

describe-dashboard

以下代码示例演示了如何使用 describe-dashboard。

Amazon CLI

描述仪表板

以下 describe-dashboard 示例描述指定风电场仪表板。

```
aws iotsitewise describe-dashboard \
    --dashboard-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE
```

输出：

```
{
    "dashboardId": "a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
    "dashboardArn": "arn:aws:iotsitewise:us-west-2:123456789012:dashboard/
a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
    "dashboardName": "Wind Farm",
    "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE",
    "dashboardDefinition": "{\"widgets\":[{\"type\":\"monitor-line-chart\",\"title
\":\"Generated Power\",\"x\":0,\"y\":0,\"height\":3,\"width\":3,\"metrics\":
[{\"label\":\"Power\",\"type\":\"iotsitewise\",\"assetId\":\"a1b2c3d4-5678-90ab-
cdef-44444EXAMPLE\",\"propertyId\":\"a1b2c3d4-5678-90ab-cdef-99999EXAMPLE\"}]}]}",
    "dashboardCreationDate": "2020-05-01T20:32:12.228476348Z",
    "dashboardLastUpdateDate": "2020-05-01T20:32:12.228476348Z"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的查看仪表板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDashboard。

describe-gateway-capability-configuration

以下代码示例演示了如何使用 describe-gateway-capability-configuration。

Amazon CLI

描述网关功能

以下 describe-gateway-capability-configuration 示例描述 OPC-UA 源功能。

```
aws iotsitewise describe-gateway-capability-configuration \
    --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
    --capability-namespace "iotsitewise:opcuacollector:1"
```

输出：

```
{
    "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
    "capabilityNamespace": "iotsitewise:opcuacollector:1",
    "capabilityConfiguration": "{\"sources\":[{\"name\":\"Wind Farm #1\",
\"endpoint\":{\"certificateTrust\":{\"type\":\"TrustAny\"},\"endpointUri
\":\"opc.tcp://203.0.113.0:49320\",\"securityPolicy\":\"BASIC256\",
\"messageSecurityMode\":\"SIGN_AND_ENCRYPT\",\"identityProvider\":
{\"type\":\"Username\",\"usernameSecretArn\":\"arn:aws:secretsmanager:us-
east-1:123456789012:secret:greengrass-factory1-auth-3QNDmM\"},\"nodeFilterRules\":
[]},\"measurementDataStreamPrefix\":\"\"}]}",
    "capabilitySyncStatus": "IN_SYNC"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的配置数据源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGatewayCapabilityConfiguration。

describe-gateway

以下代码示例演示了如何使用 describe-gateway。

Amazon CLI

描述网关

以下 describe-gateway 示例描述网关。

```
aws iotsitewise describe-gateway \
```

```
    --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE
```

输出：

```
{
    "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
    "gatewayName": "ExampleCorpGateway",
    "gatewayArn": "arn:aws:iotsitewise:us-west-2:123456789012:gateway/
a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
    "gatewayPlatform": {
        "greengrass": {
            "groupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/a1b2c3d4-5678-90ab-cdef-1b1b1EXAMPLE"
        }
    },
    "gatewayCapabilitySummaries": [
        {
            "capabilityNamespace": "iotsitewise:opcuacollector:1",
            "capabilitySyncStatus": "IN_SYNC"
        }
    ],
    "creationDate": 1588369971.457,
    "lastUpdateDate": 1588369971.457
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用网关提取数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGateway。

describe-logging-options

以下代码示例演示了如何使用 describe-logging-options。

Amazon CLI

检索当前 Amazon IoT SiteWise 日志选项

以下 describe-logging-options 示例检索当前区域中您的 Amazon 账户的当前 Amazon IoT SiteWise 日志选项。

```
aws iotsitewise describe-logging-options
```

输出：

```
{
    "loggingOptions": {
        "level": "INFO"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用 Amazon CloudWatch Logs 监控 Amazon IoT SiteWise。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoggingOptions。

describe-portal

以下代码示例演示了如何使用 describe-portal。

Amazon CLI

描述门户

以下 describe-portal 示例描述一家风电场公司的门户网站。

```
aws iotsitewise describe-portal \
    --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

输出：

```
{
    "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "portalArn": "arn:aws:iotsitewise:us-west-2:123456789012:portal/
a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "portalName": "WindFarmPortal",
    "portalDescription": "A portal that contains wind farm projects for Example
 Corp.",
    "portalClientId": "E-a1b2c3d4e5f6_a1b2c3d4e5f6EXAMPLE",
    "portalStartUrl": "https://a1b2c3d4-5678-90ab-cdef-
aaaaaEXAMPLE.app.iotsitewise.aws",
    "portalContactEmail": "support@example.com",
    "portalStatus": {
        "state": "ACTIVE"
    },
    "portalCreationDate": "2020-02-04T23:01:52.90248068Z",
```

```
    "portalLastUpdateDate": "2020-02-04T23:01:52.90248078Z",
    "roleArn": "arn:aws:iam::123456789012:role/MySiteWiseMonitorServiceRole"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的管理您的门户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePortal。

`describe-project`

以下代码示例演示了如何使用 `describe-project`。

Amazon CLI

描述项目

以下 `describe-project` 示例描述了一个风电场项目。

```
aws iotsitewise describe-project \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE
```

输出：

```
{
    "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE",
    "projectArn": "arn:aws:iotsitewise:us-west-2:123456789012:project/
a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE",
    "projectName": "Wind Farm 1",
    "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "projectDescription": "Contains asset visualizations for Wind Farm #1 for
 Example Corp.",
    "projectCreationDate": "2020-02-20T21:58:43.362246001Z",
    "projectLastUpdateDate": "2020-02-20T21:58:43.362246095Z"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的查看项目详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProject。

`disassociate-assets`

以下代码示例演示了如何使用 `disassociate-assets`。

Amazon CLI

取消子资产与父资产的关联

以下 disassociate-assets 示例取消风电涡轮机资产与风电场资产的关联。

```
aws iotsitewise disassociate-assets \
    --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \
    --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE \
    --child-asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的关联资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateAssets。

get-asset-property-aggregates

以下代码示例演示了如何使用 get-asset-property-aggregates。

Amazon CLI

检索资产属性的聚合平均值和计数值

以下 get-asset-property-aggregates 示例检索风电涡轮机资产在 1 小时内的平均总功率和总功率数据点计数。

```
aws iotsitewise get-asset-property-aggregates \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \
    --start-date 1580849400 \
    --end-date 1580853000 \
    --aggregate-types AVERAGE COUNT \
    --resolution 1h
```

输出：

```
{
    "aggregatedValues": [
        {
            "timestamp": 1580850000.0,
            "quality": "GOOD",
```

```
            "value": {
                "average": 8723.46538886233,
                "count": 12.0
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的查询资产属性聚合值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAssetPropertyAggregates。

get-asset-property-value-history

以下代码示例演示了如何使用 get-asset-property-value-history。

Amazon CLI

检索资产属性的历史值

以下 get-asset-property-value-history 示例检索风电涡轮机资产在 20 分钟内的总功率值。

```
aws iotsitewise get-asset-property-value-history \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \
    --start-date 1580851800 \
    --end-date 1580853000
```

输出：

```
{
    "assetPropertyValueHistory": [
        {
            "value": {
                "doubleValue": 7217.787046814844
            },
            "timestamp": {
                "timeInSeconds": 1580852100,
                "offsetInNanos": 0
            },
            "quality": "GOOD"
        },
```

```
        {
            "value": {
                "doubleValue": 6941.242811875451
            },
            "timestamp": {
                "timeInSeconds": 1580852400,
                "offsetInNanos": 0
            },
            "quality": "GOOD"
        },
        {
            "value": {
                "doubleValue": 6976.797662266717
            },
            "timestamp": {
                "timeInSeconds": 1580852700,
                "offsetInNanos": 0
            },
            "quality": "GOOD"
        },
        {
            "value": {
                "doubleValue": 6890.8677520453875
            },
            "timestamp": {
                "timeInSeconds": 1580853000,
                "offsetInNanos": 0
            },
            "quality": "GOOD"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的查询历史资产属性值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAssetPropertyValueHistory。

`get-asset-property-value`

以下代码示例演示了如何使用 `get-asset-property-value`。

Amazon CLI

检索资产属性的当前值

以下 get-asset-property-value 示例检索风电涡轮机资产的当前总功率。

```
aws iotsitewise get-asset-property-value \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE
```

输出：

```
{
    "propertyValue": {
        "value": {
            "doubleValue": 6890.8677520453875
        },
        "timestamp": {
            "timeInSeconds": 1580853000,
            "offsetInNanos": 0
        },
        "quality": "GOOD"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的查询当前资产属性值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAssetPropertyValue。

list-access-policies

以下代码示例演示了如何使用 list-access-policies。

Amazon CLI

列出所有访问策略

以下 list-access-policies 示例列出作为门户管理员的用户的所有访问策略。

```
aws iotsitewise list-access-policies \
    --identity-type USER \
    --identity-id a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE
```

输出：

```
{
```

```
    "accessPolicySummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE",
            "identity": {
                "user": {
                    "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
                }
            },
            "resource": {
                "portal": {
                    "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE"
                }
            },
            "permission": "ADMINISTRATOR"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的管理您的门户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessPolicies。

`list-asset-models`

以下代码示例演示了如何使用 `list-asset-models`。

Amazon CLI

列出所有资产模型

以下 `list-asset-models` 示例列出当前区域中您的 Amazon 账户内定义的所有资产模型。

```
aws iotsitewise list-asset-models
```

输出：

```
{
    "assetModelSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
```

```
            "name": "Wind Farm Model",
            "description": "Represents a wind farm that comprises many wind
 turbines",
            "creationDate": 1575671284.0,
            "lastUpdateDate": 1575671988.0,
            "status": {
                "state": "ACTIVE"
            }
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "name": "Wind Turbine Model",
            "description": "Represents a wind turbine manufactured by Example Corp",
            "creationDate": 1575671207.0,
            "lastUpdateDate": 1575686273.0,
            "status": {
                "state": "ACTIVE"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的列出所有资产模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssetModels。

`list-assets`

以下代码示例演示了如何使用 `list-assets`。

Amazon CLI

示例 1：列出所有顶级资产

以下 `list-assets` 示例列出当前区域中您的 Amazon 账户内定义的处于资产层次结构树顶级的所有资产。

```
aws iotsitewise list-assets \
    --filter TOP_LEVEL
```

输出：

```
{
    "assetSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
            "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
            "name": "Wind Farm 1",
            "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "creationDate": 1575672453.0,
            "lastUpdateDate": 1575672453.0,
            "status": {
                "state": "ACTIVE"
            },
            "hierarchies": [
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
                    "name": "Wind Turbines"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的列出资产。

示例 2：列出基于某个资产模型的所有资产

以下 `list-assets` 示例列出当前区域中您的 Amazon 账户内定义的基于某个资产模型的所有资产。

```
aws iotsitewise list-assets \
    --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "assetSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "name": "Wind Turbine 1",
```

```
            "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "creationDate": 1575671550.0,
            "lastUpdateDate": 1575686308.0,
            "status": {
                "state": "ACTIVE"
            },
            "hierarchies": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的列出资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssets。

list-associated-assets

以下代码示例演示了如何使用 list-associated-assets。

Amazon CLI

列出与特定层次结构中某个资产关联的所有资产

以下 list-associated-assets 示例列出与指定风电场资产关联的所有风电涡轮机资产。

```
aws iotsitewise list-associated-assets \
    --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \
    --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE
```

输出：

```
{
    "assetSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "name": "Wind Turbine 1",
            "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "creationDate": 1575671550.0,
            "lastUpdateDate": 1575686308.0,
            "status": {
                "state": "ACTIVE"
```

```
            },
            "hierarchies": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的列出与特定资产关联的资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssociatedAssets。

`list-dashboards`

以下代码示例演示了如何使用 `list-dashboards`。

Amazon CLI

列出项目中的所有仪表板

以下 `list-dashboards` 示例列出在某个项目中定义的所有仪表板。

```
aws iotsitewise list-dashboards \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE
```

输出：

```
{
    "dashboardSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
            "name": "Wind Farm",
            "creationDate": "2020-05-01T20:32:12.228476348Z",
            "lastUpdateDate": "2020-05-01T20:32:12.228476348Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的查看仪表板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDashboards。

`list-gateways`

以下代码示例演示了如何使用 `list-gateways`。

Amazon CLI

列出所有网关

以下 `list-gateways` 示例列出当前区域中您的 Amazon 账户内定义的所有网关。

```
aws iotsitewise list-gateways
```

输出：

```
{
    "gatewaySummaries": [
        {
            "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
            "gatewayName": "ExampleCorpGateway",
            "gatewayCapabilitySummaries": [
                {
                    "capabilityNamespace": "iotsitewise:opcuacollector:1",
                    "capabilitySyncStatus": "IN_SYNC"
                }
            ],
            "creationDate": 1588369971.457,
            "lastUpdateDate": 1588369971.457
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用网关提取数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGateways。

list-portals

以下代码示例演示了如何使用 `list-portals`。

Amazon CLI

列出所有门户

以下 `list-portals` 示例列出当前区域中您的 Amazon 账户内定义的所有门户。

```
aws iotsitewise list-portals
```

输出：

```
{
    "portalSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
            "name": "WindFarmPortal",
            "description": "A portal that contains wind farm projects for Example
 Corp.",
            "startUrl": "https://a1b2c3d4-5678-90ab-cdef-
aaaaaEXAMPLE.app.iotsitewise.aws",
            "creationDate": "2020-02-04T23:01:52.90248068Z",
            "lastUpdateDate": "2020-02-04T23:01:52.90248078Z",
            "roleArn": "arn:aws:iam::123456789012:role/service-role/
MySiteWiseMonitorServiceRole"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的管理您的门户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPortals。

list-project-assets

以下代码示例演示了如何使用 list-project-assets。

Amazon CLI

列出与项目关联的所有资产

以下 list-project-assets 示例列出与某个风电场项目关联的所有资产。

```
aws iotsitewise list-projects \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE
```

输出：

```
{
    "assetIds": [
        "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"
    ]
```

```
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的向项目添加资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProjectAssets。

list-projects

以下代码示例演示了如何使用 list-projects。

Amazon CLI

列出门户中的所有项目

以下 list-projects 示例列出在某个门户中定义的所有项目。

```
aws iotsitewise list-projects \
    --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

输出：

```
{
    "projectSummaries": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE",
            "name": "Wind Farm 1",
            "description": "Contains asset visualizations for Wind Farm #1 for
 Example Corp.",
            "creationDate": "2020-02-20T21:58:43.362246001Z",
            "lastUpdateDate": "2020-02-20T21:58:43.362246095Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的查看项目详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProjects。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源的所有标签

以下 `list-tags-for-resource` 示例列出某个风电涡轮机资产的所有标签。

```
aws iotsitewise list-tags-for-resource \
    --resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

输出：

```
{
    "tags": {
        "Owner": "richard-roe"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-logging-options

以下代码示例演示了如何使用 `put-logging-options`。

Amazon CLI

指定日志级别

以下 `put-logging-options` 示例在 Amazon IoT SiteWise 中启用 `INFO` 级别日志。其他级别包括 `DEBUG` 和 `OFF`。

```
aws iotsitewise put-logging-options \
    --logging-options level=INFO
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用 Amazon CloudWatch Logs 监控 Amazon IoT SiteWise。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingOptions。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 示例为风电涡轮机资产添加所有者标签。这使您可以基于资产的拥有者来控制对资产的访问。

```
aws iotsitewise tag-resource \
    --resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --tags Owner=richard-roe
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从风电涡轮机资产中移除所有者标签。

```
aws iotsitewise untag-resource \
    --resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --tag-keys Owner
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-access-policy

以下代码示例演示了如何使用 update-access-policy。

Amazon CLI

授予项目查看者对项目的所有权

以下 update-access-policy 示例更新了向项目查看者授予项目所有权的访问策略。

```
aws iotsitewise update-access-policy \
    --access-policy-id a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE \
    --cli-input-json file://update-project-viewer-access-policy.json
```

的内容update-project-viewer-access-policy.json：

```
{
    "accessPolicyIdentity": {
        "user": {
            "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
        }
    },
    "accessPolicyPermission": "ADMINISTRATOR",
    "accessPolicyResource": {
        "project": {
            "id": "a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE"
        }
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的分配项目所有者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccessPolicy。

update-asset-model

以下代码示例演示了如何使用 update-asset-model。

Amazon CLI

更新资产模型

以下 update-asset-model 示例更新一个风电场资产模型的描述。此示例包括模型的现有 ID 和定义，因为 update-asset-model 用新模型覆盖了现有模型。

```
aws iotsitewise update-asset-model \
    --cli-input-json file://update-wind-farm-model.json
```

的内容update-wind-farm-model.json：

```
{
    "assetModelName": "Wind Farm Model",
    "assetModelDescription": "Represents a wind farm that comprises many wind
 turbines",
    "assetModelProperties": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",
            "name": "Region",
            "dataType": "STRING",
            "type": {
                "attribute": {}
            }
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
            "name": "Total Generated Power",
            "dataType": "DOUBLE",
            "unit": "kW",
            "type": {
                "metric": {
                    "expression": "sum(power)",
                    "variables": [
                        {
                            "name": "power",
                            "value": {
                                "hierarchyId": "a1b2c3d4-5678-90ab-
cdef-77777EXAMPLE",
                                "propertyId": "a1b2c3d4-5678-90ab-cdef-66666EXAMPLE"
                            }
                        }
                    ],
                    "window": {
                        "tumbling": {
                            "interval": "1h"
                        }
```

```
                    }
                }
            }
        }
    ],
    "assetModelHierarchies": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
            "name": "Wind Turbines",
            "childAssetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
        }
    ]
}
```

输出：

```
{
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetModelStatus": {
        "state": "CREATING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的更新资产模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAssetModel。

update-asset-property

以下代码示例演示了如何使用 update-asset-property。

Amazon CLI

示例 1：更新资产属性的别名

以下 update-asset-property 示例更新风电涡轮机资产的功率属性别名。

```
aws iotsitewise update-asset-property \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-55555EXAMPLE \
    --property-alias "/examplecorp/windfarm/1/turbine/1/power" \
```

```
    --property-notification-state DISABLED
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的将工业数据流映射到资产属性。

示例 2：启用资产属性通知

以下 update-asset-property 示例为某个风电涡轮机资产的功率属性启用资产属性更新通知。属性值更新将发布到 MQTT 主题 $aws/sitewise/asset-models/<assetModelId>/assets/<assetId>/properties/<propertyId>，其中每个 ID 都替换为该资产属性的属性、资产和模型 ID。

```
aws iotsitewise update-asset-property \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \
    --property-notification-state ENABLED \
    --property-alias "/examplecorp/windfarm/1/turbine/1/power"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的与其他服务交互。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAssetProperty。

update-asset

以下代码示例演示了如何使用 update-asset。

Amazon CLI

更新资产的名称

以下 update-asset 示例更新了风电涡轮机资产的名称。

```
aws iotsitewise update-asset \
    --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
    --asset-name "Wind Turbine 2"
```

输出：

```
{
```

```
    "assetStatus": {
        "state": "UPDATING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的更新资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAsset。

update-dashboard

以下代码示例演示了如何使用 update-dashboard。

Amazon CLI

更新仪表板

以下 update-dashboard 示例更改了显示风电场总发电量的仪表板折线图的标题。

```
aws iotsitewise update-dashboard \
    --project-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE \
    --dashboard-name "Wind Farm" \
    --dashboard-definition file://update-wind-farm-dashboard.json
```

的内容update-wind-farm-dashboard.json：

```
{
    "widgets": [
        {
            "type": "monitor-line-chart",
            "title": "Total Generated Power",
            "x": 0,
            "y": 0,
            "height": 3,
            "width": 3,
            "metrics": [
                {
                    "label": "Power",
                    "type": "iotsitewise",
                    "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
                    "propertyId": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE"
                }
            ]
```

```
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的创建仪表板（CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDashboard。

update-gateway-capability-configuration

以下代码示例演示了如何使用 update-gateway-capability-configuration。

Amazon CLI

更新网关功能

以下 update-gateway-capability-configuration 示例使用下列属性配置 OPC-UA 源：

信任任何证书。使用 Basic256 算法保护消息。使用 SignAndEncrypt 模式保护连接。使用存储在 Amazon Secrets Manager 密钥中的身份验证凭证。

```
aws iotsitewise update-gateway-capability-configuration \
    --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
    --capability-namespace "iotsitewise:opcuacollector:1" \
    --capability-configuration file://opc-ua-capability-configuration.json
```

的内容opc-ua-capability-configuration.json：

```
{
    "sources": [
        {
            "name": "Wind Farm #1",
            "endpoint": {
                "certificateTrust": {
                    "type": "TrustAny"
                },
                "endpointUri": "opc.tcp://203.0.113.0:49320",
                "securityPolicy": "BASIC256",
                "messageSecurityMode": "SIGN_AND_ENCRYPT",
                "identityProvider": {
```

```
                    "type": "Username",
                    "usernameSecretArn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:greengrass-windfarm1-auth-1ABCDE"
                },
                "nodeFilterRules": []
            },
            "measurementDataStreamPrefix": ""
        }
    ]
}
```

输出：

```
{
    "capabilityNamespace": "iotsitewise:opcuacollector:1",
    "capabilitySyncStatus": "OUT_OF_SYNC"
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的配置数据源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGatewayCapabilityConfiguration。

update-gateway

以下代码示例演示了如何使用 update-gateway。

Amazon CLI

更新网关的名称

以下 update-gateway 示例更新网关的名称。

```
aws iotsitewise update-gateway \
    --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
    --gateway-name ExampleCorpGateway1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的使用网关提取数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGateway。

update-portal

以下代码示例演示了如何使用 update-portal。

Amazon CLI

更新门户的详细信息

以下 update-portal 示例更新一家风电场公司的门户网站。

```
aws iotsitewise update-portal \
    --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE \
    --portal-name WindFarmPortal \
    --portal-description "A portal that contains wind farm projects for Example
 Corp." \
    --portal-contact-email support@example.com \
    --role-arn arn:aws:iam::123456789012:role/MySiteWiseMonitorServiceRole
```

输出：

```
{
    "portalStatus": {
        "state": "UPDATING"
    }
}
```

有关更多信息，请参阅《Amazon IoT SiteWise 用户指南》中的管理您的门户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePortal。

update-project

以下代码示例演示了如何使用 update-project。

Amazon CLI

更新项目的详细信息

以下 update-project 示例更新一个风电场项目。

```
aws iotsitewise update-project \
    --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE \
    --project-name "Wind Farm 1" \
```

```
    --project-description "Contains asset visualizations for Wind Farm #1 for
Example Corp."
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT SiteWise Monitor 应用指南》中的更改项目详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProject。

Amazon IoT Things Graph使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Things Graph 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-entity-to-thing`

以下代码示例演示了如何使用 `associate-entity-to-thing`。

Amazon CLI

将事物与设备关联

以下 `associate-entity-to-thing` 示例将事物与设备关联。该示例使用公共命名空间中的运动传感器设备。

```
aws iotthingsgraph associate-entity-to-thing \
    --thing-name "MotionSensorName" \
    --entity-id "urn:tdm:aws/examples:Device:HCSR501MotionSensor"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建和上传模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateEntityToThing。

create-flow-template

以下代码示例演示了如何使用 create-flow-template。

Amazon CLI

创建流程

以下 create-flow-template 示例创建一个流程（工作流程）。MyFlowDefinition 的值是用于流程建模的 GraphQL。

```
aws iotthingsgraph create-flow-template \
    --definition language=GRAPHQL,text="MyFlowDefinition"
```

输出：

```
{
    "summary": {
        "createdAt": 1559248067.545,
        "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
        "revisionNumber": 1
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFlowTemplate。

create-system-instance

以下代码示例演示了如何使用 create-system-instance。

Amazon CLI

创建系统实例

以下 create-system-instance 示例创建一个系统实例。MySystemInstanceDefinition 的值是用于系统实例建模的 GraphQL。

```
aws iotthingsgraph create-system-instance -\
    -definition language=GRAPHQL,text="MySystemInstanceDefinition" \
    --target CLOUD \
    --flow-actions-role-arn myRoleARN
```

输出：

```
{
    "summary": {
        "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/default/
Room218",
        "status": "NOT_DEPLOYED",
        "target": "CLOUD",
        "createdAt": 1559249315.208,
        "updatedAt": 1559249315.208
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSystemInstance。

create-system-template

以下代码示例演示了如何使用 create-system-template。

Amazon CLI

创建系统

以下 create-system-template 示例创建一个系统。MySystemDefinition 的值是用于系统建模的 GraphQL。

```
aws iotthingsgraph create-system-template \
    --definition language=GRAPHQL,text="MySystemDefinition"
```

输出：

```
{
    "summary": {
        "createdAt": 1559249776.254,
```

```
        "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/
MySystem",
        "revisionNumber": 1
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建系统。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSystemTemplate。

delete-flow-template

以下代码示例演示了如何使用 delete-flow-template。

Amazon CLI

删除流程

以下 delete-flow-template 示例删除一个流程（工作流程）。

```
aws iotthingsgraph delete-flow-template \
    --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFlowTemplate。

delete-namespace

以下代码示例演示了如何使用 delete-namespace。

Amazon CLI

删除命名空间

以下 delete-namespace 示例删除一个命名空间。

```
aws iotthingsgraph delete-namespace
```

输出：

```
{
   "namespaceArn": "arn:aws:iotthingsgraph:us-west-2:123456789012",
   "namespaceName": "us-west-2/123456789012/default"
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNamespace。

delete-system-instance

以下代码示例演示了如何使用 delete-system-instance。

Amazon CLI

删除系统实例

以下 delete-system-instance 示例删除一个系统实例。

```
aws iotthingsgraph delete-system-instance \
    --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSystemInstance。

delete-system-template

以下代码示例演示了如何使用 delete-system-template。

Amazon CLI

删除系统

以下 delete-system-template 示例删除一个系统。

```
aws iotthingsgraph delete-system-template \
```

```
    --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSystemTemplate。

deploy-system-instance

以下代码示例演示了如何使用 deploy-system-instance。

Amazon CLI

部署系统实例

以下 delete-system-template 示例部署一个系统实例。

```
aws iotthingsgraph deploy-system-instance \
    --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

输出：

```
{
   "summary": {
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment:Room218",
      "createdAt": 1559249776.254,
      "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
      "status": "DEPLOYED_IN_TARGET",
      "target": "CLOUD",
      "updatedAt": 1559249776.254
   }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeploySystemInstance。

deprecate-flow-template

以下代码示例演示了如何使用 deprecate-flow-template。

Amazon CLI

弃用流程

以下 deprecate-flow-template 示例弃用一个流程（工作流程）。

```
aws iotthingsgraph deprecate-flow-template \
    --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprecateFlowTemplate。

deprecate-system-template

以下代码示例演示了如何使用 deprecate-system-template。

Amazon CLI

弃用系统

以下 deprecate-system-template 示例弃用一个系统。

```
aws iotthingsgraph deprecate-system-template \
    --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprecateSystemTemplate。

describe-namespace

以下代码示例演示了如何使用 describe-namespace。

Amazon CLI

获取命名空间的描述

以下 describe-namespace 示例获取您的命名空间的描述。

```
aws iotthingsgraph describe-namespace
```

输出：

```
{
    "namespaceName": "us-west-2/123456789012/default",
    "trackingNamespaceName": "aws",
    "trackingNamespaceVersion": 1,
    "namespaceVersion": 5
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNamespace。

dissociate-entity-from-thing

以下代码示例演示了如何使用 dissociate-entity-from-thing。

Amazon CLI

取消事物与设备的关联

以下 dissociate-entity-from-thing 示例取消事物与设备的关联。

```
aws iotthingsgraph dissociate-entity-from-thing \
    --thing-name "MotionSensorName" \
    --entity-type "DEVICE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建和上传模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DissociateEntityFromThing。

get-entities

以下代码示例演示了如何使用 get-entities。

Amazon CLI

获取实体的定义

以下 get-entities 示例获取设备模型的定义。

```
aws iotthingsgraph get-entities \
    --ids "urn:tdm:aws/examples:DeviceModel:MotionSensor"
```

输出：

```
{
    "descriptions": [
        {
            "id": "urn:tdm:aws/examples:DeviceModel:MotionSensor",
            "type": "DEVICE_MODEL",
            "createdAt": 1559256190.599,
            "definition": {
                "language": "GRAPHQL",
                "text": "##\n# Specification of motion sensor devices interface.\n##
\ntype MotionSensor @deviceModel(id: \"urn:tdm:aws/examples:deviceModel:MotionSensor
\",\n        capability: \"urn:tdm:aws/examples:capability:MotionSensorCapability\")
 {ignore:void}"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建和上传模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEntities。

get-flow-template-revisions

以下代码示例演示了如何使用 get-flow-template-revisions。

Amazon CLI

获取有关流程的修订信息

以下 get-flow-template-revisions 示例获取有关流程（工作流程）的修订信息。

```
aws iotthingsgraph get-flow-template-revisions \
```

```
    --id urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow
```

输出：

```
{
    "summaries": [
        {
            "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
            "revisionNumber": 1,
            "createdAt": 1559247540.292
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFlowTemplateRevisions。

get-flow-template

以下代码示例演示了如何使用 get-flow-template。

Amazon CLI

获取流程定义

以下 get-flow-template 示例获取一个流程（工作流程）的定义。

```
aws iotthingsgraph get-flow-template \
    --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

输出：

```
{
    "description": {
        "summary": {
            "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
            "revisionNumber": 1,
            "createdAt": 1559247540.292
        },
        "definition": {
            "language": "GRAPHQL",
```

```
            "text": "{\nquery MyFlow($camera: string!, $screen: string!)
 @workflowType(id: \"urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow\")
 @annotation(type: \"tgc:FlowEvent\", id: \"sledged790c1b2bcd949e09da0c9bfc077f79d
\", x: 1586, y: 653) @triggers(definition: \"{MotionSensor(description:
 \\\"\\\") @position(x: 1045, y: 635.6666564941406) {\\n  condition(expr:
 \\\"devices[name == \\\\\\\"motionSensor\\\\\\\"].events[name == \\\
\\\\\"StateChanged\\\\\\\"].lastEvent\\\")\\n  action(expr: \\\"\\\")\
\n}}\") {\n  variables {\n    cameraResult @property(id: \"urn:tdm:aws/
examples:property:CameraStateProperty\")\n  }\n  steps {\n    step(name: \"Camera
\", outEvent: [\"sledged790c1b2bcd949e09da0c9bfc077f79d\"]) @position(x: 1377,
 y: 638.6666564941406) {\n      DeviceActivity(deviceModel: \"urn:tdm:aws/
examples:deviceModel:Camera\", out: \"cameraResult\", deviceId: \"${camera}\")
 {\n        capture\n      }\n    }\n    step(name: \"Screen\", inEvent:
 [\"sledged790c1b2bcd949e09da0c9bfc077f79d\"]) @position(x: 1675.6666870117188,
 y: 637.9999847412109) {\n      DeviceActivity(deviceModel: \"urn:tdm:aws/
examples:deviceModel:Screen\", deviceId: \"${screen}\") {\n        display(imageUrl:
 \"${cameraResult.lastClickedImage}\")\n      }\n    }\n  }\n}\n}"
        },
        "validatedNamespaceVersion": 5
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFlowTemplate。

get-namespace-deletion-status

以下代码示例演示了如何使用 get-namespace-deletion-status。

Amazon CLI

获取命名空间删除任务的状态

以下 get-namespace-deletion-status 示例获取命名空间删除任务的状态。

```
aws iotthingsgraph get-namespace-deletion-status
```

输出：

```
{
   "namespaceArn": "arn:aws:iotthingsgraph:us-west-2:123456789012",
```

```
    "namespaceName": "us-west-2/123456789012/default"
    "status": "SUCCEEDED "
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的命名空间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetNamespaceDeletionStatus。

get-system-instance

以下代码示例演示了如何使用 get-system-instance。

Amazon CLI

获取系统实例

以下 get-system-instance 示例获取一个系统实例的定义。

```
aws iotthingsgraph get-system-instance \
    --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

输出：

```
{
    "description": {
        "summary": {
            "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218",
            "status": "NOT_DEPLOYED",
            "target": "CLOUD",
            "createdAt": 1559249315.208,
            "updatedAt": 1559249315.208
        },
        "definition": {
            "language": "GRAPHQL",
            "text": "{\r\nquery Room218 @deployment(id: \"urn:tdm:us-
west-2/123456789012/default:Deployment:Room218\", systemId: \"urn:tdm:us-
west-2/123456789012/default:System:SecurityFlow\") {\r\n    motionSensor(deviceId:
 \"MotionSensorName\")\r\n    screen(deviceId: \"ScreenName\")\r\n
 camera(deviceId: \"CameraName\") \r\n    triggers {MotionEventTrigger(description:
 \"a trigger\") {  \r\n    condition(expr: \"devices[name ==
```

```
 'motionSensor'].events[name == 'StateChanged'].lastEvent\") \r\n    action(expr:
 \"ThingsGraph.startFlow('SecurityFlow', bindings[name == 'camera'].deviceId,
 bindings[name == 'screen'].deviceId)\")\r\n    }\r\n   }\r\n  }\r\n  }"
        },
        "metricsConfiguration": {
            "cloudMetricEnabled": false
        },
        "validatedNamespaceVersion": 5,
        "flowActionsRoleArn": "arn:aws:iam::123456789012:role/ThingsGraphRole"
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSystemInstance。

get-system-template-revisions

以下代码示例演示了如何使用 get-system-template-revisions。

Amazon CLI

获取有关系统的修订信息

以下 get-system-template-revisions 示例获取有关一个系统的修订信息。

```
aws iotthingsgraph get-system-template-revisions \
    --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

输出：

```
{
    "summaries": [
        {
            "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/
MySystem",
            "revisionNumber": 1,
            "createdAt": 1559247540.656
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSystemTemplateRevisions。

get-system-template

以下代码示例演示了如何使用 get-system-template。

Amazon CLI

获取系统

以下 get-system-template 示例获取一个系统的定义。

```
aws iotthingsgraph get-system-template \
    --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

输出：

```
{
    "description": {
        "summary": {
            "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/
MyFlow",
            "revisionNumber": 1,
            "createdAt": 1559247540.656
        },
        "definition": {
            "language": "GRAPHQL",
            "text": "{\ntype MySystem @systemType(id: \"urn:tdm:us-
west-2/123456789012/default:System:MySystem\", description: \"\") {\n  camera:
 Camera @thing(id: \"urn:tdm:aws/examples:deviceModel:Camera\")\n  screen:
 Screen @thing(id: \"urn:tdm:aws/examples:deviceModel:Screen\")\n  motionSensor:
 MotionSensor @thing(id: \"urn:tdm:aws/examples:deviceModel:MotionSensor
\")\n  MyFlow: MyFlow @workflow(id: \"urn:tdm:us-west-2/123456789012/
default:Workflow:MyFlow\")\n}\n}"
        },
        "validatedNamespaceVersion": 5
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSystemTemplate。

get-upload-status

以下代码示例演示了如何使用 get-upload-status。

Amazon CLI

获取实体上传的状态

以下 get-upload-status 示例获取您的实体上传操作的状态。MyUploadId 的值是 upload-entity-definitions 操作返回的 ID 值。

```
aws iotthingsgraph get-upload-status \
    --upload-id "MyUploadId"
```

输出：

```
{
    "namespaceName": "us-west-2/123456789012/default",
    "namespaceVersion": 5,
    "uploadId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",
    "uploadStatus": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的实体建模。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUploadStatus。

list-flow-execution-messages

以下代码示例演示了如何使用 list-flow-execution-messages。

Amazon CLI

获取有关流程执行中事件的信息

以下 list-flow-execution-messages 示例获取有关流程执行中事件的信息。

```
aws iotthingsgraph list-flow-execution-messages \
```

```
    --flow-execution-id "urn:tdm:us-west-2/123456789012/
default:Workflow:SecurityFlow_2019-05-11T19:39:55.317Z_MotionSensor_69b151ad-
a611-42f5-ac21-fe537f9868ad"
```

输出：

```
{
    "messages": [
        {
         "eventType": "EXECUTION_STARTED",
         "messageId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",
         "payload": "Flow execution started",
         "timestamp": 1559247540.656
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFlowExecutionMessages。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源的所有标签

以下 list-tags-for-resource 示例列出一个 Amazon IoT Things Graph 资源的所有标签。

```
aws iotthingsgraph list-tags-for-resource \
    --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218"
```

输出：

```
{
   "tags": [
      {
         "key": "Type",
         "value": "Residential"
```

```
      }
   ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的标记您的 Amazon IoT Things Graph 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

search-entities

以下代码示例演示了如何使用 search-entities。

Amazon CLI

搜索实体

以下 search-entities 示例搜索类型为 EVENT 的所有实体。

```
aws iotthingsgraph search-entities \
    --entity-types "EVENT"
```

输出：

```
{
    "descriptions": [
        {
            "id": "urn:tdm:aws/examples:Event:MotionSensorEvent",
            "type": "EVENT",
            "definition": {
                "language": "GRAPHQL",
                "text": "##\n# Description of events emitted by motion
 sensor.\n##\ntype MotionSensorEvent @eventType(id: \"urn:tdm:aws/
examples:event:MotionSensorEvent\",\n            payload: \"urn:tdm:aws/
examples:property:MotionSensorStateProperty\") {ignore:void}"
            }
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/
default:Event:CameraClickedEventV2",
            "type": "EVENT",
            "definition": {
                "language": "GRAPHQL",
```

```
                "text": "type CameraClickedEventV2 @eventType(id: \"urn:tdm:us-
west-2/123456789012/default:event:CameraClickedEventV2\",\r\npayload:
 \"urn:tdm:aws:Property:Boolean\"){ignore:void}"
            }
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/
default:Event:MotionSensorEventV2",
            "type": "EVENT",
            "definition": {
                "language": "GRAPHQL",
                "text": "# Event emitted by the motion sensor.\r\ntype
 MotionSensorEventV2 @eventType(id: \"urn:tdm:us-west-2/123456789012/
default:event:MotionSensorEventV2\",\r\npayload: \"urn:tdm:us-west-2/123456789012/
default:property:MotionSensorStateProperty2\") {ignore:void}"
            }
        }
    ],
    "nextToken": "urn:tdm:us-west-2/123456789012/default:Event:MotionSensorEventV2"
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 数据模型参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchEntities。

search-flow-executions

以下代码示例演示了如何使用 search-flow-executions。

Amazon CLI

搜索流程执行

以下 search-flow-executions 示例搜索指定系统实例中某个流程的所有执行。

```
aws iotthingsgraph search-flow-executions \
    --system-instance-id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

输出：

```
{
   "summaries": [
```

```
      {
         "createdAt": 1559247540.656,
         "flowExecutionId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",
         "flowTemplateId": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
         "status": "RUNNING ",
         "systemInstanceId": "urn:tdm:us-west-2/123456789012/
default:System:MySystem",
         "updatedAt": 1559247540.656
      }
   ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchFlowExecutions。

`search-flow-templates`

以下代码示例演示了如何使用 `search-flow-templates`。

Amazon CLI

搜索流程（或工作流程）

以下 `search-flow-templates` 示例搜索包含相机设备模型的所有流程（工作流程）。

```
aws iotthingsgraph search-flow-templates \
    --filters name="DEVICE_MODEL_ID",value="urn:tdm:aws/examples:DeviceModel:Camera"
```

输出：

```
{
    "summaries": [
        {
            "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
            "revisionNumber": 1,
            "createdAt": 1559247540.292
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/default:Workflow:SecurityFlow",
            "revisionNumber": 3,
            "createdAt": 1548283099.27
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchFlowTemplates。

`search-system-instances`

以下代码示例演示了如何使用 `search-system-instances`。

Amazon CLI

搜索系统实例

以下 `search-system-instances` 示例搜索包含指定系统的所有系统实例。

```
aws iotthingsgraph search-system-instances \
    --filters name="SYSTEM_TEMPLATE_ID",value="urn:tdm:us-west-2/123456789012/
default:System:SecurityFlow"
```

输出：

```
{
    "summaries": [
        {
            "id": "urn:tdm:us-west-2/123456789012/
default:Deployment:DeploymentForSample",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/DeploymentForSample",
            "status": "NOT_DEPLOYED",
            "target": "GREENGRASS",
            "greengrassGroupName": "ThingsGraphGrnGr",
            "createdAt": 1555716314.707,
            "updatedAt": 1555716314.707
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/
default:Deployment:MockDeployment",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/MockDeployment",
            "status": "DELETED_IN_TARGET",
            "target": "GREENGRASS",
```

```
            "greengrassGroupName": "ThingsGraphGrnGr",
            "createdAt": 1549416462.049,
            "updatedAt": 1549416722.361,
            "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",
            "greengrassGroupVersionId": "7365aed7-2d3e-4d13-aad8-75443d45eb05"
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/
default:Deployment:MockDeployment2",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/MockDeployment2",
            "status": "DEPLOYED_IN_TARGET",
            "target": "GREENGRASS",
            "greengrassGroupName": "ThingsGraphGrnGr",
            "createdAt": 1549572385.774,
            "updatedAt": 1549572418.408,
            "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",
            "greengrassGroupVersionId": "bfa70ab3-2bf7-409c-a4d4-bc8328ae5b86"
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room215",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room215",
            "status": "NOT_DEPLOYED",
            "target": "GREENGRASS",
            "greengrassGroupName": "ThingsGraphGG",
            "createdAt": 1547056918.413,
            "updatedAt": 1547056918.413
        },
        {
            "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218",
            "status": "NOT_DEPLOYED",
            "target": "CLOUD",
            "createdAt": 1559249315.208,
            "updatedAt": 1559249315.208
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用系统和流程配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchSystemInstances。

search-system-templates

以下代码示例演示了如何使用 search-system-templates。

Amazon CLI

搜索系统

以下 search-system-templates 示例搜索包含指定流程的所有系统。

```
aws iotthingsgraph search-system-templates \
    --filters name="FLOW_TEMPLATE_ID",value="urn:tdm:us-west-2/123456789012/
default:Workflow:SecurityFlow"
```

输出：

```
{
    "summaries": [
        {
            "id": "urn:tdm:us-west-2/123456789012/default:System:SecurityFlow",
            "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/
SecurityFlow",
            "revisionNumber": 1,
            "createdAt": 1548283099.433
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchSystemTemplates。

search-things

以下代码示例演示了如何使用 search-things。

Amazon CLI

搜索与设备和设备模型关联的事物

以下 search-things 示例搜索与 HCSR501MotionSensor 设备关联的所有事物。

```
aws iotthingsgraph search-things \
```

```
    --entity-id "urn:tdm:aws/examples:Device:HCSR501MotionSensor"
```

输出：

```
{
    "things": [
        {
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MotionSensor1",
            "thingName": "MotionSensor1"
        },
        {
            "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/TG_MS",
            "thingName": "TG_MS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建和上传模型。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchThings。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源创建标签

以下 tag-resource 示例为指定资源创建标签。

```
aws iotthingsgraph tag-resource \
    --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218" \
    --tags key="Type",value="Residential"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的标记您的 Amazon IoT Things Graph 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

undeploy-system-instance

以下代码示例演示了如何使用 undeploy-system-instance。

Amazon CLI

从目标中取消部署系统实例

以下 undeploy-system-instance 示例将一个系统实例从其目标中移除。

```
aws iotthingsgraph undeploy-system-instance \
    --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room215"
```

输出：

```
{
    "summary": {
        "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room215",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/default/
Room215",
        "status": "PENDING_DELETE",
        "target": "GREENGRASS",
        "greengrassGroupName": "ThingsGraphGrnGr",
        "createdAt": 1553189694.255,
        "updatedAt": 1559344549.601,
        "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",
        "greengrassGroupVersionId": "731b371d-d644-4b67-ac64-3934e99b75d7"
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的 Amazon IoT Things Graph 实体、流程、系统和部署的生命周期管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UndeploySystemInstance。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除资源的标签

以下 untag-resource 示例移除指定资源的标签。

```
aws iotthingsgraph untag-resource \
    --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/default/Room218" \
    --tag-keys "Type"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的标记您的 Amazon IoT Things Graph 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-flow-template

以下代码示例演示了如何使用 update-flow-template。

Amazon CLI

更新流程

以下 update-flow-template 示例更新一个流程（工作流程）。MyFlowDefinition 的值是用于流程建模的 GraphQL。

```
aws iotthingsgraph update-flow-template \
    --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow" \
    --definition language=GRAPHQL,text="MyFlowDefinition"
```

输出：

```
{
    "summary": {
        "createdAt": 1559248067.545,
        "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
        "revisionNumber": 2
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的使用流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFlowTemplate。

update-system-template

以下代码示例演示了如何使用 update-system-template。

Amazon CLI

更新系统

以下 update-system-template 示例更新一个系统。MySystemDefinition 的值是用于系统建模的 GraphQL。

```
aws iotthingsgraph update-system-template \
    --id "urn:tdm:us-west-2/123456789012/default:System:MySystem" \
    --definition language=GRAPHQL,text="MySystemDefinition"
```

输出：

```
{
    "summary": {
        "createdAt": 1559249776.254,
        "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/
MySystem",
        "revisionNumber": 2
    }
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的创建系统。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSystemTemplate。

upload-entity-definitions

以下代码示例演示了如何使用 upload-entity-definitions。

Amazon CLI

上传实体定义

以下 upload-entity-definitions 示例将实体定义上传到您的命名空间。MyEntityDefinitions 的值是用于实体建模的 GraphQL。

```
aws iotthingsgraph upload-entity-definitions \
    --document language=GRAPHQL,text="MyEntityDefinitions"
```

输出：

```
{
    "uploadId": "f6294f1e-b109-4bbe-9073-f451a2dda2da"
}
```

有关更多信息，请参阅《Amazon IoT Things Graph 用户指南》中的实体建模。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadEntityDefinitions。

使用 Amazon CLI 的 Amazon IoT Wireless 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IoT Wireless 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-aws-account-with-partner-account`

以下代码示例演示了如何使用 `associate-aws-account-with-partner-account`。

Amazon CLI

将合作伙伴账户与您的 Amazon 账户关联

以下 `associate-aws-account-with-partner-account` 示例将下列 Sidewalk 账户凭证与您的 Amazon 账户相关联。

```
aws iotwireless associate-aws-account-with-partner-account \
    --sidewalk
 AmazonId="12345678901234",AppServerPrivateKey="a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78
```

输出：

```
{
    "Sidewalk": {
        "AmazonId": "12345678901234",
        "AppServerPrivateKey":
 "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT Core 的 Amazon Sidewalk 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAwsAccountWithPartnerAccount。

associate-wireless-device-with-thing

以下代码示例演示了如何使用 associate-wireless-device-with-thing。

Amazon CLI

将事物与无线设备关联

以下 associate-wireless-device-with-thing 示例将事物与您的具有指定 ID 的无线设备相关联。

```
aws iotwireless associate-wireless-device-with-thing \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
    --thing-arn "arn:aws:iot:us-east-1:123456789012:thing/MyIoTWirelessThing"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateWirelessDeviceWithThing。

associate-wireless-gateway-with-certificate

以下代码示例演示了如何使用 associate-wireless-gateway-with-certificate。

Amazon CLI

将证书与无线网关关联

以下 associate-wireless-gateway-with-certificate 命令将无线网关与证书相关联。

```
aws iotwireless associate-wireless-gateway-with-certificate \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
    --iot-certificate-
id "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
```

输出：

```
{
    "IotCertificateId":
 "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateWirelessGatewayWithCertificate。

associate-wireless-gateway-with-thing

以下代码示例演示了如何使用 associate-wireless-gateway-with-thing。

Amazon CLI

将事物与无线网关关联

以下 associate-wireless-gateway-with-thing 示例将事物与无线网关关联。

```
aws iotwireless associate-wireless-gateway-with-thing \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
    --thing-arn "arn:aws:iot:us-east-1:123456789012:thing/MyIoTWirelessThing"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateWirelessGatewayWithThing。

create-destination

以下代码示例演示了如何使用 create-destination。

Amazon CLI

创建 IoT 无线目标

以下 create-destination 示例创建用于将设备消息映射到 Amazon IoT 规则的目标。在运行此命令之前，您必须先创建一个 IAM 角色，该角色赋予 Amazon IoT Core for LoRaWAN 将数据发送到 Amazon IoT 规则所必需的权限。

```
aws iotwireless create-destination \
    --name IoTWirelessDestination \
    --expression-type RuleName \
    --expression IoTWirelessRule \
    --role-arn arn:aws:iam::123456789012:role/IoTWirelessDestinationRole
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination",
    "Name": "IoTWirelessDestination"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDestination。

create-device-profile

以下代码示例演示了如何使用 create-device-profile。

Amazon CLI

创建新的设备配置文件

以下 create-device-profile 示例创建了一个新的 IoT 无线设备配置文件。

```
aws iotwireless create-device-profile
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDeviceProfile。

create-service-profile

以下代码示例演示了如何使用 create-service-profile。

Amazon CLI

创建新的服务配置文件

以下 create-service-profile 示例创建了一个新的 IoT 无线服务配置文件。

```
aws iotwireless create-service-profile
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:ServiceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceProfile。

create-wireless-device

以下代码示例演示了如何使用 create-wireless-device。

Amazon CLI

创建 IoT 无线设备

以下 create-wireless-device 示例创建类型 LoRaWAN 的无线设备资源。

```
aws iotwireless create-wireless-device \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "Description": "My LoRaWAN wireless device"
    "DestinationName": "IoTWirelessDestination"
    "LoRaWAN": {
        "DeviceProfileId": "ab0c23d3-b001-45ef-6a01-2bc3de4f5333",
        "ServiceProfileId": "fe98dc76-cd12-001e-2d34-5550432da100",
        "OtaaV1_1": {
            "AppKey": "3f4ca100e2fc675ea123f4eb12c4a012",
            "JoinEui": "b4c231a359bc2e3d",
            "NwkKey": "01c3f004a2d6efffe32c4eda14bcd2b4"
        },
        "DevEui": "ac12efc654d23fc2"
    },
    "Name": "SampleIoTWirelessThing"
    "Type": LoRaWAN
}
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessDevice/1ffd32c8-8130-4194-96df-622f072a315f",
```

```
    "Id": "1ffd32c8-8130-4194-96df-622f072a315f"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWirelessDevice。

create-wireless-gateway-task-definition

以下代码示例演示了如何使用 create-wireless-gateway-task-definition。

Amazon CLI

创建无线网关作业定义

以下 create-wireless-gateway-task-definition 命令自动为所有具有指定当前版本的网关创建使用此作业定义的任务。

```
aws iotwireless create-wireless-gateway-task-definition \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "AutoCreateTasks": true,
    "Name": "TestAutoUpdate",
    "Update":{
        "UpdateDataSource" : "s3://cupsalphagafirmwarebin/station",
        "UpdateDataRole" : "arn:aws:iam::001234567890:role/SDK_Test_Role",
        "LoRaWAN" :{
            "CurrentVersion" :{
                "PackageVersion" : "1.0.0",
                "Station" : "2.0.5",
                "Model" : "linux"
            },
            "UpdateVersion" :{
                "PackageVersion" : "1.0.1",
                "Station" : "2.0.5",
                "Model" : "minihub"
            }
        }
```

```
    }
}
```

输出：

```
{
    "Id": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWirelessGatewayTaskDefinition。

`create-wireless-gateway-task`

以下代码示例演示了如何使用 `create-wireless-gateway-task`。

Amazon CLI

为无线网关创建任务

以下 `create-wireless-gateway-task` 示例为无线网关创建了一个任务。

```
aws iotwireless create-wireless-gateway-task \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
    --wireless-gateway-task-definition-id "aa000102-0304-b0cd-ef56-a1b23cde456a"
```

输出：

```
{
    "WirelessGatewayTaskDefinitionId": "aa204003-0604-30fb-ac82-a4f95aaf450a",
    "Status": "Success"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWirelessGatewayTask。

create-wireless-gateway

以下代码示例演示了如何使用 create-wireless-gateway。

Amazon CLI

创建无线网关

以下 create-wireless-gateway 示例创建无线 LoRaWAN 设备网关。

```
aws iotwireless create-wireless-gateway \
    --lorawan GatewayEui="a1b2c3d4567890ab",RfRegion="US915" \
    --name "myFirstLoRaWANGateway" \
    --description "Using my first LoRaWAN gateway"
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:WirelessGateway/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWirelessGateway。

delete-destination

以下代码示例演示了如何使用 delete-destination。

Amazon CLI

删除 IoT 无线目标

以下 delete-destination 示例删除您创建的名称为 IoTWirelessDestination 的无线目标资源。

```
aws iotwireless delete-destination \
    --name "IoTWirelessDestination"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDestination。

delete-device-profile

以下代码示例演示了如何使用 delete-device-profile。

Amazon CLI

删除设备配置文件

以下 delete-device-profile 示例删除了您创建的具有指定 ID 的设备配置文件。

```
aws iotwireless delete-device-profile \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDeviceProfile。

delete-service-profile

以下代码示例演示了如何使用 delete-service-profile。

Amazon CLI

删除服务配置文件

以下 delete-service-profile 示例删除了您创建的具有指定 ID 的服务配置文件。

```
aws iotwireless delete-service-profile \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceProfile。

delete-wireless-device

以下代码示例演示了如何使用 delete-wireless-device。

Amazon CLI

删除无线设备

以下 delete-wireless-device 示例删除具有指定 ID 的无线设备。

```
aws iotwireless delete-wireless-device \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWirelessDevice。

delete-wireless-gateway-task-definition

以下代码示例演示了如何使用 delete-wireless-gateway-task-definition。

Amazon CLI

删除无线网关作业定义

以下 delete-wireless-gateway-task-definition 示例删除您创建的具有以下 ID 的无线网关作业定义。

```
aws iotwireless delete-wireless-gateway-task-definition \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWirelessGatewayTaskDefinition。

delete-wireless-gateway-task

以下代码示例演示了如何使用 delete-wireless-gateway-task。

Amazon CLI

删除无线网关任务

以下 delete-wireless-gateway-task 示例删除具有指定 ID 的无线网关任务。

```
aws iotwireless delete-wireless-gateway-task \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWirelessGatewayTask。

delete-wireless-gateway

以下代码示例演示了如何使用 delete-wireless-gateway。

Amazon CLI

删除无线网关

以下 delete-wireless-gateway 示例删除具有指定 ID 的无线网关。

```
aws iotwireless delete-wireless-gateway \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWirelessGateway。

disassociate-aws-account-from-partner-account

以下代码示例演示了如何使用 disassociate-aws-account-from-partner-account。

Amazon CLI

取消合作伙伴账户与 Amazon 账户关联

以下 disassociate-aws-account-from-partner-account 示例取消合作伙伴账户与您当前关联的 Amazon 账户的关联。

```
aws iotwireless disassociate-aws-account-from-partner-account \
    --partner-account-id "12345678901234" \
    --partner-type "Sidewalk"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateAwsAccountFromPartnerAccount。

disassociate-wireless-device-from-thing

以下代码示例演示了如何使用 disassociate-wireless-device-from-thing。

Amazon CLI

取消事物与无线设备的关联

以下 disassociate-wireless-device-from-thing 示例取消无线设备与其当前关联的事物的关联。

```
aws iotwireless disassociate-wireless-device-from-thing \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateWirelessDeviceFromThing。

disassociate-wireless-gateway-from-certificate

以下代码示例演示了如何使用 disassociate-wireless-gateway-from-certificate。

Amazon CLI

取消证书与无线网关的关联

以下 disassociate-wireless-gateway-from-certificate 示例取消无线网关与其当前关联的证书的关联。

```
aws iotwireless disassociate-wireless-gateway-from-certificate \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateWirelessGatewayFromCertificate。

disassociate-wireless-gateway-from-thing

以下代码示例演示了如何使用 disassociate-wireless-gateway-from-thing。

Amazon CLI

取消事物与无线网关的关联

以下 disassociate-wireless-gateway-from-thing 示例取消无线网关与其当前关联的事物的关联。

```
aws iotwireless disassociate-wireless-gateway-from-thing \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将网关和无线设备添加到 Amazon IoT Core for LoRaWAN 中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateWirelessGatewayFromThing。

get-destination

以下代码示例演示了如何使用 get-destination。

Amazon CLI

获取有关 IoT 无线目标的信息

以下 get-destination 示例获取有关您创建的名称为 IoTWirelessDestination 的目标资源的信息。

```
aws iotwireless get-destination \
    --name "IoTWirelessDestination"
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination",
    "Name": "IoTWirelessDestination",
    "Expression": "IoTWirelessRule",
    "ExpressionType": "RuleName",
    "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDestination。

get-device-profile

以下代码示例演示了如何使用 get-device-profile。

Amazon CLI

获取有关设备配置文件的信息

以下 get-device-profile 示例获取有关您创建的具有指定 ID 的设备配置文件的信息。

```
aws iotwireless get-device-profile \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
    "LoRaWAN": {
    "MacVersion": "1.0.3",
    "MaxDutyCycle": 10,
    "Supports32BitFCnt": false,
    "RegParamsRevision": "RP002-1.0.1",
    "SupportsJoin": true,
    "RfRegion": "US915",
    "MaxEirp": 13,
    "SupportsClassB": false,
    "SupportsClassC": false
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDeviceProfile。

get-partner-account

以下代码示例演示了如何使用 get-partner-account。

Amazon CLI

获取合作伙伴账户信息

以下 get-partner-account 示例获取有关具有以下 ID 的 Sidewalk 账户的信息。

```
aws iotwireless get-partner-account \
    --partner-account-id "12345678901234" \
    --partner-type "Sidewalk"
```

输出：

```
{
    "Sidewalk": {
        "AmazonId": "12345678901234",
        "Fingerprint":
 "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
    },
    "AccountLinked": false
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT Core 的 Amazon Sidewalk 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPartnerAccount。

get-service-endpoint

以下代码示例演示了如何使用 get-service-endpoint。

Amazon CLI

获取服务端点

以下 get-service-endpoint 示例获取 CUPS 协议的特定账户端点。

```
aws iotwireless get-service-endpoint
```

输出：

```
{
    "ServiceType": "CUPS",
    "ServiceEndpoint": "https://A1RMKZ37ACAGOT.cups.lorawan.us-
east-1.amazonaws.com:443",
    "ServerTrust": "-----BEGIN CERTIFICATE-----\n
    MIIESTCCAzGgAwIBAgITBn+UV4WH6Kx33rJTMlu8mYtWDTANBgkqhkiG9w0BAQsF\n
```

```
    ADA5MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1hem9uMRkwFwYDVQQDExBBbWF6\n
    b24gUm9vdCBDQSAxMB4XDTE1MTAyMjAwMDAwMFoXDTI1MTAxOTAwMDAwMFowRjEL\n
    MAkGA1UEBhMCVVMxDzANBgNVBAoTBkFtYXpvbjEVMBMGA1UECxMMU2VydmVyIENB\n
    IDFCMQ8wDQYDVQQDEwZBbWF6b24wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK\n
    AoIBAQDCThZn3c68asg3Wuw6MLAd5tES6BIoSMzoKcG5blPVo+sDORrMd4f2AbnZ\n
    cMzPa43j4wNxhplty6aUKk4T1qe9BOwKFjwK6zmxxLVYo7bHViXsPlJ6qOMpFge5\n
    blDP+18x+B26A0piiQOuPkfyDyeR4xQghfj66Yo19V+emU3nazfvpFA+ROz6WoVm\n
    B5x+F2pV8xeKNR7u6azDdU5YVX1TawprmxRC1+WsAYmz6qP+z8ArDITC2FMVy2fw\n
    0IjKOtEXc/VfmtTFch5+AfGYMGMqqvJ6LcXiAhqG5TI+Dr0RtM88k+8XUBCeQ8IG\n
    KuANaL7TiItKZYxK1MMuTJtV9IblAgMBAAGjggE7MIIBNzASBgNVHRMBAf8ECDAG\n
    AQH/AgEAMA4GA1UdDwEB/wQEAwIBhjAdBgNVHQ4EFgQUWaRmBlKge5WSPKOUByeW\n
    dFv5PdAwHwYDVR0jBBgwFoAUhBjMhTTsvAyUlC4IWZzHshBOCggwewYIKwYBBQUH\n
    AQEEbzBtMC8GCCsGAQUFBzABhiNodHRwOi8vb2NzcC5yb290Y2ExLmFtYXpvbnRy\n
    dXN0LmNvbTA6BggrBgEFBQcwAoYuaHR0cDovL2NydC5yb290Y2ExLmFtYXpvbnRy\n
    dXN0LmNvbS9yb290Y2ExLmNlcjA/BgNVHR8EODA2MDSgMqAwhi5odHRwOi8vY3Js\n
    LnJvb3RjYTEuYW1hem9udHJ1c3QuY29tL3Jvb3RjYTEuY3JsMBMGA1UdIAQMMAow\n
    CAYGZ4EMAQIBMA0GCSqGSIb3DQEBCwUAA4IBAQCFkr41u3nPo4FCHOTjY3NTOVI1\n
    59Gt/a6ZiqyJEi+752+a1U5y6iAwYfmXss2lJwJFqMp2PphKg5625kXg8kP2CN5t\n
    6G7bMQcT8C8xDZNtYTd7WPD8UZiRKAJPBXa30/AbwuZe0GaFEQ8ugcYQgSn+IGBI\n
    8/LwhBNTZTUVEWuCUUBVV18YtbAiPq3yXqMB48Oz+ctBWuZSkbvkNodPLamkB2g1\n
    upRyzQ7qDn1X8nn8N8V7YJ6y68AtkHcNSRAnpTitxBKjtKPISLMVCx7i4hncxHZS\n
    yLyKQXhw2W2Xs0qLeC1etA+jTGDK4UfLeC0SF7FSi8o5LL21L8IzApar2pR/\n
    -----END CERTIFICATE-----\n"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceEndpoint。

get-service-profile

以下代码示例演示了如何使用 get-service-profile。

Amazon CLI

获取有关服务配置文件的信息

以下 get-service-profile 示例获取有关您创建的具有指定 ID 的服务配置文件的信息。

```
aws iotwireless get-service-profile \
    --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

输出：

```
{
    "Arn": "arn:aws:iotwireless:us-east-1:651419225604:ServiceProfile/538185bb-
d7e7-4b95-96a0-c51aa4a5b9a0",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
    "LoRaWAN": {
        "HrAllowed": false,
        "NwkGeoLoc": false,
        "DrMax": 15,
        "UlBucketSize": 4096,
        "PrAllowed": false,
        "ReportDevStatusBattery": false,
        "DrMin": 0,
        "DlRate": 60,
        "AddGwMetadata": false,
        "ReportDevStatusMargin": false,
        "MinGwDiversity": 1,
        "RaAllowed": false,
        "DlBucketSize": 4096,
        "DevStatusReqFreq": 24,
        "TargetPer": 5,
        "UlRate": 60
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceProfile。

get-wireless-device-statistics

以下代码示例演示了如何使用 get-wireless-device-statistics。

Amazon CLI

获取有关无线设备的操作信息

以下 get-wireless-device-statistics 示例获取有关无线设备的操作信息。

```
aws iotwireless get-wireless-device-statistics \
    --wireless-device-id "1ffd32c8-8130-4194-96df-622f072a315f"
```

输出：

```
{
    "WirelessDeviceId": "1ffd32c8-8130-4194-96df-622f072a315f"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessDeviceStatistics。

get-wireless-device

以下代码示例演示了如何使用 get-wireless-device。

Amazon CLI

获取有关无线设备的信息

以下 get-wireless-device 示例列出您的 Amazon 账户中可用的微件。

```
aws iotwireless get-wireless-device \
    --identifier "1ffd32c8-8130-4194-96df-622f072a315f" \
    --identifier-type WirelessDeviceID
```

输出：

```
{
    "Name": "myLoRaWANDevice",
    "ThingArn": "arn:aws:iot:us-east-1:123456789012:thing/44b87eb4-9bce-423d-
b5fc-973f5ecc358b",
    "DestinationName": "IoTWirelessDestination",
    "Id": "1ffd32c8-8130-4194-96df-622f072a315f",
    "ThingName": "44b87eb4-9bce-423d-b5fc-973f5ecc358b",
    "Type": "LoRaWAN",
    "LoRaWAN": {
        "DeviceProfileId": "ab0c23d3-b001-45ef-6a01-2bc3de4f5333",
        "ServiceProfileId": "fe98dc76-cd12-001e-2d34-5550432da100",
        "OtaaV1_1": {
            "AppKey": "3f4ca100e2fc675ea123f4eb12c4a012",
            "JoinEui": "b4c231a359bc2e3d",
            "NwkKey": "01c3f004a2d6efffe32c4eda14bcd2b4"
```

```
        },
        "DevEui": "ac12efc654d23fc2"
    },
    "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessDevice/1ffd32c8-8130-4194-96df-622f072a315f",
    "Description": "My LoRaWAN wireless device"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessDevice。

get-wireless-gateway-certificate

以下代码示例演示了如何使用 get-wireless-gateway-certificate。

Amazon CLI

获取与无线网关关联的证书的 ID

以下 get-wireless-gateway-certificate 示例获取与具有指定 ID 的无线网关关联的证书 ID。

```
aws iotwireless get-wireless-gateway-certificate \
    --id "6c44ab31-8b4d-407a-bed3-19b6c7cda551"
```

输出：

```
{
    "IotCertificateId":
 "8ea4aeae3db34c78cce75d9abd830356869ead6972997e0603e5fd032c804b6f"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGatewayCertificate。

get-wireless-gateway-firmware-information

以下代码示例演示了如何使用 get-wireless-gateway-firmware-information。

Amazon CLI

获取有关无线网关的固件信息

以下 get-wireless-gateway-firmware-information 示例获取有关无线网关的固件版本和其他信息。

```
aws iotwireless get-wireless-gateway-firmware-information \
    --id "3039b406-5cc9-4307-925b-9948c63da25b"
```

输出：

```
{
    "LoRaWAN" :{
        "CurrentVersion" :{
            "PackageVersion" : "1.0.0",
            "Station" : "2.0.5",
            "Model" : "linux"
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGatewayFirmwareInformation。

get-wireless-gateway-statistics

以下代码示例演示了如何使用 get-wireless-gateway-statistics。

Amazon CLI

获取有关无线网关的操作信息

以下 get-wireless-gateway-statistics 示例获取有关无线网关的操作信息。

```
aws iotwireless get-wireless-gateway-statistics \
    --wireless-gateway-id "3039b406-5cc9-4307-925b-9948c63da25b"
```

输出：

```
{
    "WirelessGatewayId": "3039b406-5cc9-4307-925b-9948c63da25b"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGatewayStatistics。

get-wireless-gateway-task-definition

以下代码示例演示了如何使用 get-wireless-gateway-task-definition。

Amazon CLI

获取有关无线网关作业定义的信息

以下 get-wireless-gateway-task-definition 示例获取有关具有指定 ID 的无线作业定义的信息。

```
aws iotwireless get-wireless-gateway-task-definition \
    --id "b7d3baad-25c7-35e7-a4e1-1683a0d61da9"
```

输出：

```
{
    "AutoCreateTasks": true,
    "Name": "TestAutoUpdate",
    "Update":{
        "UpdateDataSource" : "s3://cupsalphagafirmwarebin/station",
        "UpdateDataRole" : "arn:aws:iam::001234567890:role/SDK_Test_Role",
        "LoRaWAN" :{
            "CurrentVersion" :{
                "PackageVersion" : "1.0.0",
                "Station" : "2.0.5",
                "Model" : "linux"
            },
            "UpdateVersion" :{
                "PackageVersion" : "1.0.1",
                "Station" : "2.0.5",
```

```
                "Model" : "minihub"
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGatewayTaskDefinition。

get-wireless-gateway-task

以下代码示例演示了如何使用 get-wireless-gateway-task。

Amazon CLI

获取有关无线网关任务的信息

以下 get-wireless-gateway-task 示例获取有关具有指定 ID 的无线网关定义的信息。

```
aws iotwireless get-wireless-gateway-task \
    --id "11693a46-6866-47c3-a031-c9a616e7644b"
```

输出：

```
{
    "WirelessGatewayId": "6c44ab31-8b4d-407a-bed3-19b6c7cda551",
    "WirelessGatewayTaskDefinitionId": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9",
    "Status": "Success"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGatewayTask。

get-wireless-gateway

以下代码示例演示了如何使用 get-wireless-gateway。

Amazon CLI

获取有关无线网关的信息

以下 get-wireless-gateway 示例获取有关无线网关 myFirstLoRaWANGateway 的信息。

```
aws iotwireless get-wireless-gateway \
    --identifier "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
    --identifier-type WirelessGatewayId
```

输出：

```
{
    "Description": "My first LoRaWAN gateway",
    "ThingArn": "arn:aws:iot:us-east-1:123456789012:thing/a1b2c3d4-5678-90ab-
cdef-12ab345c67de",
    "LoRaWAN": {
        "RfRegion": "US915",
        "GatewayEui": "a1b2c3d4567890ab"
    },
    "ThingName": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
    "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
    "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/6c44ab31-8b4d-407a-bed3-19b6c7cda551",
    "Name": "myFirstLoRaWANGateway"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWirelessGateway。

list-destinations

以下代码示例演示了如何使用 list-destinations。

Amazon CLI

列出无线目标

以下 list-destinations 示例列出注册到您的 Amazon 账户的可用目标。

```
aws iotwireless list-destinations
```

输出：

```
{
    "DestinationList": [
        {
            "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination",
            "Name": "IoTWirelessDestination",
            "Expression": "IoTWirelessRule",
            "Description": "Destination for messages processed using
 IoTWirelessRule",
            "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
        },
        {
            "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination2",
            "Name": "IoTWirelessDestination2",
            "Expression": "IoTWirelessRule2",
            "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDestinations。

list-device-profiles

以下代码示例演示了如何使用 list-device-profiles。

Amazon CLI

列出设备配置文件

以下 list-device-profiles 示例列出注册到您的 Amazon 账户的可用设备配置文件。

```
aws iotwireless list-device-profiles
```

输出：

```
{
    "DeviceProfileList": [
        {
            "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
            "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:DeviceProfile/12345678-a1b2-3c45-67d8-e90fa1b2c34d"
        },
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
            "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/
a1b2c3d4-5678-90ab-cdef-12ab345c67de"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeviceProfiles。

`list-partner-accounts`

以下代码示例演示了如何使用 `list-partner-accounts`。

Amazon CLI

列出合作伙伴账户

以下 `list-partner-accounts` 示例列出与您的 Amazon 账户关联的可用合作伙伴账户。

```
aws iotwireless list-partner-accounts
```

输出：

```
{
    "Sidewalk": [
        {
            "AmazonId": "78965678771228",
            "Fingerprint":
 "bd96d8ef66dbfd2160eb60e156849e82ad7018b8b73c1ba0b4fc65c32498ee35"
```

```
        },
        {
            "AmazonId": "89656787651228",
            "Fingerprint":
 "bc5e99e151c07be14be7e6603e4489c53f858b271213a36ebe3370777ba06e9b"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT Core 的 Amazon Sidewalk 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPartnerAccounts。

`list-service-profiles`

以下代码示例演示了如何使用 `list-service-profiles`。

Amazon CLI

列出服务配置文件

以下 `list-service-profiles` 示例列出注册到您的 Amazon 账户的可用服务配置文件。

```
aws iotwireless list-service-profiles
```

输出：

```
{
    "ServiceProfileList": [
        {
            "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
            "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:ServiceProfile/538185bb-d7e7-4b95-96a0-c51aa4a5b9a0"
        },
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
            "Arn": "arn:aws:iotwireless:us-east-1:123456789012:ServiceProfile/
ea8bc823-5d13-472e-8d26-9550737d8100"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceProfiles。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出分配给资源的标签

以下 `list-tags-for-resource` 示例列出分配给无线目标资源的标签。

```
aws iotwireless list-tags-for-resource \
    --resource-arn "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination"
```

输出：

```
{
    "Tags": [
        {
            "Value": "MyValue",
            "Key": "MyTag"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的描述您的 Amazon IoT Core for LoRaWAN 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-wireless-devices`

以下代码示例演示了如何使用 `list-wireless-devices`。

Amazon CLI

列出可用的无线设备

以下 list-wireless-devices 示例列出注册到您的 Amazon 账户的可用无线设备。

```
aws iotwireless list-wireless-devices
```

输出：

```
{
    "WirelessDeviceList": [
        {
            "Name": "myLoRaWANDevice",
            "DestinationName": "IoTWirelessDestination",
            "Id": "1ffd32c8-8130-4194-96df-622f072a315f",
            "Type": "LoRaWAN",
            "LoRaWAN": {
                "DevEui": "ac12efc654d23fc2"
            },
            "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessDevice/1ffd32c8-8130-4194-96df-622f072a315f"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWirelessDevices。

list-wireless-gateway-task-definitions

以下代码示例演示了如何使用 list-wireless-gateway-task-definitions。

Amazon CLI

列出无线网关作业定义

以下 list-wireless-gateway-task-definitions 示例列出注册到您的 Amazon 账户的可用无线网关作业定义。

```
aws iotwireless list-wireless-gateway-task-definitions
```

输出：

```
{
    "TaskDefinitions": [
        {
            "Id": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9",
            "LoRaWAN" :
                {
                "CurrentVersion" :{
                    "PackageVersion" : "1.0.0",
                    "Station" : "2.0.5",
                    "Model" : "linux"
                },
                "UpdateVersion" :{
                    "PackageVersion" : "1.0.1",
                    "Station" : "2.0.5",
                    "Model" : "minihub"
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWirelessGatewayTaskDefinitions。

`list-wireless-gateways`

以下代码示例演示了如何使用 `list-wireless-gateways`。

Amazon CLI

列出无线网关

以下 `list-wireless-gateways` 示例列出您的 Amazon 账户中可用的无线网关。

```
aws iotwireless list-wireless-gateways
```

输出：

```
{
```

```
    "WirelessGatewayList": [
        {
            "Description": "My first LoRaWAN gateway",
            "LoRaWAN": {
                "RfRegion": "US915",
                "GatewayEui": "dac632ebc01d23e4"
            },
            "Id": "3039b406-5cc9-4307-925b-9948c63da25b",
            "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/3039b406-5cc9-4307-925b-9948c63da25b",
            "Name": "myFirstLoRaWANGateway"
        },
        {
            "Description": "My second LoRaWAN gateway",
            "LoRaWAN": {
                "RfRegion": "US915",
                "GatewayEui": "cda123fffe92ecd2"
            },
            "Id": "3285bdc7-5a12-4991-84ed-dadca65e342e",
            "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/3285bdc7-5a12-4991-84ed-dadca65e342e",
            "Name": "mySecondLoRaWANGateway"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWirelessGateways。

send-data-to-wireless-device

以下代码示例演示了如何使用 send-data-to-wireless-device。

Amazon CLI

向无线设备发送数据

以下 send-data-to-wireless-device 示例将解密的应用程序数据帧发送到无线设备。

```
aws iotwireless send-data-to-wireless-device \
    --id "11aa5eae-2f56-4b8e-a023-b28d98494e49" \
```

```
    --transmit-mode "1" \
    --payload-data "SGVsbG8gVG8gRGV2c2lt" \
    --wireless-metadata LoRaWAN={FPort=1}
```

输出：

```
{
    MessageId: "6011dd36-0043d6eb-0072-0008"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendDataToWirelessDevice。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源指定标签键和值

以下 tag-resource 示例标记键为 MyTag 且值为 MyValue 的无线目标 IoTWirelessDestination。

```
aws iotwireless tag-resource \
    --resource-arn "arn:aws:iotwireless:us-east-1:651419225604:Destination/IoTWirelessDestination" \
    --tags Key="MyTag",Value="MyValue"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的描述您的 Amazon IoT Core for LoRaWAN 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

test-wireless-device

以下代码示例演示了如何使用 test-wireless-device。

Amazon CLI

测试无线设备

以下 `test-wireless-device` 示例将 `Hello` 的上行链路数据发送到具有指定 ID 的设备。

```
aws iotwireless test-wireless-device \
    --id "11aa5eae-2f56-4b8e-a023-b28d98494e49"
```

输出：

```
{
    Result: "Test succeeded. one message is sent with payload: hello"
}
```

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TestWirelessDevice。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从资源中移除一个或多个标签

以下 `untag-resource` 示例从无线目标 `IoTWirelessDestination` 中移除标签 `MyTag` 及其值。

```
aws iotwireless untag-resource \
    --resource-arn "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination" \
    --tag-keys "MyTag"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的描述您的 Amazon IoT Core for LoRaWAN 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-destination

以下代码示例演示了如何使用 update-destination。

Amazon CLI

更新目标的属性

以下 update-destination 示例更新了一个无线目标的描述属性。

```
aws iotwireless update-destination \
    --name "IoTWirelessDestination" \
    --description "Destination for messages processed using IoTWirelessRule"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的为 Amazon IoT Core for LoRaWAN 添加目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDestination。

update-partner-account

以下代码示例演示了如何使用 update-partner-account。

Amazon CLI

更新合作伙伴账户的属性

以下 update-partner-account 更新了具有指定 ID 的账户的 AppServerPrivateKey。

```
aws iotwireless update-partner-account \
    --partner-account-id "78965678771228" \
    --partner-type "Sidewalk" \
    --sidewalk
 AppServerPrivateKey="f798ab4899346a88599180fee9e14fa1ada7b6df989425b7c6d2146dd6c815bb"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的 Amazon IoT Core 的 Amazon Sidewalk 集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePartnerAccount。

update-wireless-device

以下代码示例演示了如何使用 update-wireless-device。

Amazon CLI

更新无线设备的属性

以下 update-wireless-device 示例更新注册到您的 Amazon 账户的无线设备的属性。

```
aws iotwireless update-wireless-device \
    --id "1ffd32c8-8130-4194-96df-622f072a315f" \
    --destination-name IoTWirelessDestination2 \
    --description "Using my first LoRaWAN device"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWirelessDevice。

update-wireless-gateway

以下代码示例演示了如何使用 update-wireless-gateway。

Amazon CLI

更新无线网关

以下 update-wireless-gateway 示例更新无线网关的描述。

```
aws iotwireless update-wireless-gateway \
    --id "3285bdc7-5a12-4991-84ed-dadca65e342e" \
    --description "Using my LoRaWAN gateway"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IoT 开发人员指南》中的将设备和网关连接到 Amazon IoT Core for LoRaWAN。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWirelessGateway。

使用 Amazon CLI 的 Amazon IVS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IVS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-get-channel`

以下代码示例演示了如何使用 `batch-get-channel`。

Amazon CLI

获取有关多个通道的通道配置信息

以下 `batch-get-channel` 示例列出有关指定通道的信息。

```
aws ivs batch-get-channel \
    --arns arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
        arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl
```

输出：

```
{
    "channels": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "authorized": false,
            "containerFormat": "TS",
            "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
            "insecureIngest": false,
            "latencyMode": "LOW",
            "multitrackInputConfiguration": {
```

```
                "enabled": false,
                "maximumResolution": "FULL_HD",
                "policy": "ALLOW"
            },
            "name": "channel-1",
            "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/
api/video/v1/us-west-2.123456789012.channel-1.abcdEFGH.m3u8",
            "preset": "",
            "playbackRestrictionPolicyArn": "",
            "recordingConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABCD12cdEFgh",
            "srt": {
                "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
                "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
            },
            "tags": {},
            "type": "STANDARD"
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
            "authorized": false,
            "containerFormat": "FRAGMENTED_MP4",
            "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
            "insecureIngest": false,
            "latencyMode": "LOW",
            "multitrackInputConfiguration": {
                "enabled": true,
                "maximumResolution": "FULL_HD",
                "policy": "ALLOW"
            },
            "name": "channel-2",
            "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/
api/video/v1/us-west-2.123456789012.channel-2.abcdEFGH.m3u8",
            "preset": "",
            "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ"",
            "recordingConfigurationArn": "",
            "srt": {
                "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
                "passphrase":
 "BA1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
            },
            "tags": {},
```

```
            "type": "STANDARD"
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetChannel。

batch-get-stream-key

以下代码示例演示了如何使用 batch-get-stream-key。

Amazon CLI

获取有关多个直播密钥的信息

以下 batch-get-stream-key 示例获取有关指定直播密钥的信息。

```
aws ivs batch-get-stream-key \
    --arns arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh \
       arn:aws:ivs:us-west-2:123456789012:stream-key/skSKIJKLmnop
```

输出：

```
{
    "streamKeys": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh",
            "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKIJKLmnop",
            "value": "sk_us-west-2_abcdABCDefgh_567890ghijkl",
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "tags": {}
        }
     ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetStreamKey。

batch-start-viewer-session-revocation

以下代码示例演示了如何使用 batch-start-viewer-session-revocation。

Amazon CLI

撤消多个 channel-ARN 和 viewer-ID 对的观看者会话

以下 batch-start-viewer-session-revocation 示例同时对多个 channel-ARN 和 viewer-ID 对执行会话撤销。如果调用者无权撤消指定会话，则请求可能正常完成，但会在错误字段中返回值。

```
aws ivs batch-start-viewer-session-revocation \
    --viewer-sessions '[{"channelArn":"arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh1","viewerId":"abcdefg1","viewerSessionVersionsLessThanOrEqualTo":1234567890},
 \
      {"channelArn":"arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh2","viewerId":"abcdefg2","viewerSessionVersionsLessThanOrEqualTo":1234567890}]'
```

输出：

```
{
    "errors": [
        {
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh1",
            "viewerId": "abcdefg1",
            "code": "403",
            "message": "not authorized",
        },
        {
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh2",
            "viewerId": "abcdefg2",
            "code": "403",
            "message": "not authorized",
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchStartViewerSessionRevocation。

create-channel

以下代码示例演示了如何使用 create-channel。

Amazon CLI

示例 1：创建没有录制的通道

以下 create-channel 示例创建了一个新通道和关联的直播流密钥以启动直播。

```
aws ivs create-channel \
    --name 'test-channel' \
    --no-insecure-ingest
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "authorized": false,
        "containerFormat": "TS",
        "name": "test-channel",
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
```

```
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "tags": {},
        "type": "STANDARD"
    },
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

示例 2：使用通过 ARN 指定的 RecordingConfiguration 资源来创建启动了录制的通道

以下 `create-channel` 示例创建一个新通道和关联的直播密钥来启动直播，并为通道设置了录制：

```
aws ivs create-channel \
    --name test-channel-with-recording \
    --insecure-ingest \
    --recording-configuration-arn 'arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "containerFormat": "TS",
        "name": "test-channel-with-recording",
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
```

```
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "BA1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": true,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {},
        "type": "STANDARD"
    },
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的录制到 Amazon S3。

示例 3：使用通过 ARN 指定的播放限制策略创建通道

以下 `create-channel` 示例创建一个新通道和关联的直播密钥来启动直播，并为通道设置了播放限制策略：

```
aws ivs create-channel \
    --name test-channel-with-playback-restriction-policy\
    --insecure-ingest \
    --playback-restriction-policy-arn 'arn:aws:ivs:us-west-2:123456789012:playback-
restriction-policy/ABcdef34ghIJ'
```

输出：

```
{
    "channel": {
```

```
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "containerFormat": "TS",
        "name": "test-channel-with-playback-restriction-policy",
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2edfGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": true,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {},
        "type": "STANDARD"
    },
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

示例 4：创建启用了多轨道的通道

以下 `create-channel` 示例创建一个新通道和关联的直播流密钥以启动直播，并启用多轨道。

```
aws ivs create-channel \
    --name 'test-channel' \
    --no-insecure-ingest \
```

```
    --container-format 'FRAGMENTED_MP4' \
    --multitrack-input-configuration '{"enabled": true,"maximumResolution":
 "FULL_HD","policy": "ALLOW"}'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "authorized": false,
        "containerFormat": "FRAGMENTED_MP4",
        "name": "test-channel",
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": true,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "tags": {},
        "type": "STANDARD"
    },
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChannel。

create-playback-restriction-policy

以下代码示例演示了如何使用 create-playback-restriction-policy。

Amazon CLI

创建播放限制策略

以下 create-playback-restriction-policy 示例创建新的播放限制策略。

```
aws ivs create-playback-restriction-policy \
    --name "test-playback-restriction-policy" \
    --enable-strict-origin-enforcement \
    --tags "key1=value1, key2=value2" \
    --allowed-countries US MX \
    --allowed-origins https://www.website1.com https://www.website2.com
```

输出：

```
{
    "playbackRestrictionPolicy": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
        "allowedCountries": [
            "US",
            "MX"
        ],
        "allowedOrigins": [
            "https://www.website1.com",
            "https://www.website2.com"
        ],
        "enableStrictOriginEnforcement": true,
        "name": "test-playback-restriction-policy",
        "tags": {
            "key1": "value1",
            "key2": "value2"
        }
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePlaybackRestrictionPolicy。

create-recording-configuration

以下代码示例演示了如何使用 create-recording-configuration。

Amazon CLI

创建 RecordingConfiguration 资源

以下 create-recording-configuration 示例创建 RecordingConfiguration 资源以启用录制到 Amazon S3。

```
aws ivs create-recording-configuration \
    --name "test-recording-config" \
    --recording-reconnect-window-seconds 60 \
    --tags "key1=value1, key2=value2" \
    --rendition-configuration renditionSelection="CUSTOM",renditions="HD" \
    --thumbnail-configuration
 recordingMode="INTERVAL",targetIntervalSeconds=1,storage="LATEST",resolution="LOWEST_RESOLU
 \
    --destination-configuration s3={bucketName=demo-recording-bucket}
```

输出：

```
{
    "recordingConfiguration": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
        "name": "test-recording-config",
        "destinationConfiguration": {
            "s3": {
                "bucketName": "demo-recording-bucket"
            }
        },
        "state": "CREATING",
        "tags": {
            "key1": "value1",
            "key2": "value2"
        },
        "thumbnailConfiguration": {
            "recordingMode": "INTERVAL",
            "targetIntervalSeconds": 1,
            "resolution": "LOWEST_RESOLUTION",
            "storage": [
```

```
                "LATEST"
            ]
        },
        "recordingReconnectWindowSeconds": 60,
        "renditionConfiguration": {
            "renditionSelection": "CUSTOM",
            "renditions": [
                "HD"
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的录制到 Amazon S3。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRecordingConfiguration。

`create-stream-key`

以下代码示例演示了如何使用 `create-stream-key`。

Amazon CLI

创建直播密钥

以下 `create-stream-key` 示例为指定 ARN（Amazon 资源名称）创建直播密钥。

```
aws ivs create-stream-key \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

输出：

```
{
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStreamKey。

delete-channel

以下代码示例演示了如何使用 delete-channel。

Amazon CLI

删除通道及其关联的直播密钥

以下 delete-channel 示例删除具有指定 ARN（Amazon 资源名称）的通道。

```
aws ivs delete-channel \
    --arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

此命令不生成任何输出。

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteChannel。

delete-playback-key-pair

以下代码示例演示了如何使用 delete-playback-key-pair。

Amazon CLI

删除指定的播放密钥对

以下 delete-playback-key-pair 示例返回指定密钥对的指纹。

```
aws ivs delete-playback-key-pair \
    --arn arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePlaybackKeyPair。

delete-playback-restriction-policy

以下代码示例演示了如何使用 delete-playback-restriction-policy。

Amazon CLI

删除播放限制策略

以下 delete-playback-restriction-policy 示例删除具有指定策略 ARN（Amazon 资源名称）的播放限制策略。

```
aws ivs delete-playback-restriction-policy \
    --arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ"
```

此命令不生成任何输出。

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePlaybackRestrictionPolicy。

delete-recording-configuration

以下代码示例演示了如何使用 delete-recording-configuration。

Amazon CLI

删除通过 ARN 指定的 RecordingConfiguration 资源

以下 delete-recording-configuration 示例删除具有指定 ARN 的 RecordingConfiguration 资源。

```
aws ivs delete-recording-configuration \
    --arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/ABcdef34ghIJ"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的录制到 Amazon S3。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRecordingConfiguration。

delete-stream-key

以下代码示例演示了如何使用 delete-stream-key。

Amazon CLI

删除直播密钥

以下 delete-stream-key 示例删除了指定 ARN（Amazon 资源名称）的直播密钥，因此该密钥不能再用于直播。

```
aws ivs delete-stream-key \
    --arn arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6
```

此命令不生成任何输出。

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStreamKey。

get-channel

以下代码示例演示了如何使用 get-channel。

Amazon CLI

获取通道的配置信息

以下 get-channel 示例获取指定通道 ARN（Amazon 资源名称）的通道配置。

```
aws ivs get-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "authorized": false,
        "containerFormat": "TS",
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
```

```
            "policy": "ALLOW"
        },
        "name": "channel-1",
        "playbackRestrictionPolicyArn": "",
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "tags": {}
        "type": "STANDARD",
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetChannel。

get-playback-key-pair

以下代码示例演示了如何使用 get-playback-key-pair。

Amazon CLI

获取指定的播放密钥对

以下 get-playback-key-pair 示例返回指定密钥对的指纹。

```
aws ivs get-playback-key-pair \
    --arn arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh
```

输出：

```
{
    "keyPair": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
        "name": "my-playback-key",
        "fingerprint": "0a:1b:2c:ab:cd:ef:34:56:70:b1:b2:71:01:2a:a3:72",
        "tags": {}
```

```
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPlaybackKeyPair。

get-playback-restriction-policy

以下代码示例演示了如何使用 get-playback-restriction-policy。

Amazon CLI

获取播放限制策略的配置信息

以下 get-playback-restriction-policy 示例获取具有指定策略 ARN（Amazon 资源名称）的播放限制策略配置。

```
aws ivs get-playback-restriction-policy \
    --arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ"
```

输出：

```
{
    "playbackRestrictionPolicy": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
        "allowedCountries": [
            "US",
            "MX"
        ],
        "allowedOrigins": [
            "https://www.website1.com",
            "https://www.website2.com"
        ],
        "enableStrictOriginEnforcement": true,
        "name": "test-playback-restriction-policy",
        "tags": {
            "key1": "value1",
            "key2": "value2"
        }
    }
```

```
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPlaybackRestrictionPolicy。

`get-recording-configuration`

以下代码示例演示了如何使用 `get-recording-configuration`。

Amazon CLI

获取有关 RecordingConfiguration 资源的信息

以下 `get-recording-configuration` 示例获取指定 ARN 的 RecordingConfiguration 资源信息。

```
aws ivs get-recording-configuration \
    --arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/ABcdef34ghIJ"
```

输出：

```
{
    "recordingConfiguration": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
        "destinationConfiguration": {
            "s3": {
                "bucketName": "demo-recording-bucket"
            }
        },
        "name": "test-recording-config",
        "recordingReconnectWindowSeconds": 60,
        "state": "ACTIVE",
        "tags": {
            "key1" : "value1",
            "key2" : "value2"
        },
        "thumbnailConfiguration": {
            "recordingMode": "INTERVAL",
            "targetIntervalSeconds": 1,
            "resolution": "LOWEST_RESOLUTION",
            "storage": [
```

```
                "LATEST"
            ]
        },
        "renditionConfiguration": {
            "renditionSelection": "CUSTOM",
            "renditions": [
                "HD"
            ]
        }
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的录制到 Amazon S3。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRecordingConfiguration。

get-stream-key

以下代码示例演示了如何使用 get-stream-key。

Amazon CLI

获取有关直播的信息

以下 get-stream-key 示例获取有关指定直播密钥的信息。

```
aws ivs get-stream-key \
    --arn arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh --region=us-
west-2
```

输出：

```
{
    "streamKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh",
        "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStreamKey。

get-stream-session

以下代码示例演示了如何使用 get-stream-session。

Amazon CLI

获取指定直播的元数据

以下 get-stream-session 示例获取指定通道 ARN（Amazon 资源名称）和指定直播的元数据配置；如果未提供 streamId，则会选择通道的最近直播。

```
aws ivs get-stream-session \
    --channel-arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --stream-id 'mystream'
```

输出：

```
{
    "streamSession": {
        "streamId": "mystream1",
        "startTime": "2023-06-26T19:09:28+00:00",
        "channel": {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "name": "mychannel",
            "latencyMode": "LOW",
            "type": "STANDARD",
            "recordingConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABcdef34ghIJ",
            "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
            "playbackUrl": "url-string",
            "authorized": false,
            "insecureIngest": false,
            "preset": ""
        },
        "ingestConfiguration": {
            "audio": {
                "channels": 2,
                "codec": "mp4a.40.2",
                "sampleRate": 8000,
                "targetBitrate": 46875,
                "track": "Track0"
```

```
            },
            "video": {
                "avcProfile": "Baseline",
                "avcLevel": "4.2",
                "codec": "avc1.42C02A",
                "encoder": "Lavf58.45.100",
                "level": "4.2",
                "profile": "Baseline",
                "targetBitrate": 8789062,
                "targetFramerate": 60,
                "track": "Track0",
                "videoHeight": 1080,
                "videoWidth": 1920
            }
        },
        "ingestConfigurations": {
            "audioConfigurations": [
                {
                    "channels": 2,
                    "codec": "mp4a.40.2",
                    "sampleRate": 8000,
                    "targetBitrate": 46875,
                    "track": "Track0"
                }
            ],
            "videoConfigurations": [
                {
                    "codec": "avc1.42C02A",
                    "encoder": "Lavf58.45.100",
                    "level": "4.2",
                    "profile": "Baseline",
                    "targetBitrate": 8789062,
                    "targetFramerate": 60,
                    "track": "Track0",
                    "videoHeight": 1080,
                    "videoWidth": 1920
                }
            ]
        },
        "recordingConfiguration": {
            "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
            "name": "test-recording-config",
            "destinationConfiguration": {
```

```
                "s3": {
                    "bucketName": "demo-recording-bucket"
                }
            },
            "state": "ACTIVE",
            "tags": {
                "key1": "value1",
                "key2": "value2"
            },
            "thumbnailConfiguration": {
                "recordingMode": "INTERVAL",
                "targetIntervalSeconds": 1,
                "resolution": "LOWEST_RESOLUTION",
                "storage": [
                    "LATEST"
                ]
            },
            "recordingReconnectWindowSeconds": 60,
            "renditionConfiguration": {
                "renditionSelection": "CUSTOM",
                "renditions": [
                    "HD"
                ]
            }
        },
        "truncatedEvents": [
            {
                "code": "StreamTakeoverInvalidPriority",
                "name": "Stream Takeover Failure",
                "type": "IVS Stream State Change",
                "eventTime": "2023-06-26T19:09:48+00:00"
            },
            {
                "name": "Stream Takeover",
                "type": "IVS Stream State Change",
                "eventTime": "2023-06-26T19:09:47+00:00"
            },
            {
                "name": "Recording Start",
                "type": "IVS Recording State Change",
                "eventTime": "2023-06-26T19:09:35+00:00"
            },
            {
                "name": "Stream Start",
```

```
                "type": "IVS Stream State Change",
                "eventTime": "2023-06-26T19:09:34+00:00"
            },
            {
                "name": "Session Created",
                "type": "IVS Stream State Change",
                "eventTime": "2023-06-26T19:09:28+00:00"
            }
        ]
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStreamSession。

get-stream

以下代码示例演示了如何使用 get-stream。

Amazon CLI

获取有关直播的信息

以下 get-stream 示例获取有关指定通道的直播的信息。

```
aws ivs get-stream \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

输出：

```
{
    "stream": {
        "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "startTime": "2020-05-05T21:55:38Z",
        "state": "LIVE",
        "health": "HEALTHY",
        "streamId": "st-ABCDEfghij01234KLMN5678",
        "viewerCount": 1
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStream。

`import-playback-key-pair`

以下代码示例演示了如何使用 `import-playback-key-pair`。

Amazon CLI

导入新密钥对的公共部分

以下 `import-playback-key-pair` 示例导入指定公钥（以 PEM 格式指定的字符串），并返回新密钥对的 arn 和指纹。

```
aws ivs import-playback-key-pair \
    --name "my-playback-key" \
    --public-key-material "G1lbnQxOTA3BgNVBAMMMFdoeSBhcmUgeW91IGRl..."
```

输出：

```
{
    "keyPair": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
        "name": "my-playback-key",
        "fingerprint": "0a:1b:2c:ab:cd:ef:34:56:70:b1:b2:71:01:2a:a3:72",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportPlaybackKeyPair。

`list-channels`

以下代码示例演示了如何使用 `list-channels`。

Amazon CLI

示例 1：获取所有通道的摘要信息

以下 `list-channels` 示例列出您的 Amazon 账户的所有通道。

```
aws ivs list-channels
```

输出：

```
{
    "channels": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "name": "channel-1",
            "latencyMode": "LOW",
            "authorized": false,
            "insecureIngest": false,
            "preset": "",
            "playbackRestrictionPolicyArn": "",
            "recordingConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABCD12cdEFgh",
            "tags": {},
            "type": "STANDARD"
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
            "name": "channel-2",
            "latencyMode": "LOW",
            "authorized": false,
            "preset": "",
            "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
            "recordingConfigurationArn": "",
            "tags": {},
            "type": "STANDARD"
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

示例 2：获取按指定 RecordingConfiguration ARN 筛选的所有通道的摘要信息

以下 `list-channels` 示例列出您的 Amazon 账户中与指定 RecordingConfiguration ARN 关联的所有通道。

```
aws ivs list-channels \
    --filter-by-recording-configuration-arn "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABCD12cdEFgh"
```

输出：

```
{
    "channels": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "name": "channel-1",
            "latencyMode": "LOW",
            "authorized": false,
            "insecureIngest": false,
            "preset": "",
            "playbackRestrictionPolicyArn": "",
            "recordingConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABCD12cdEFgh",
            "tags": {},
            "type": "STANDARD"
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的录制到 Amazon S3。

示例 3：获取按指定 PlaybackRestrictionPolicy ARN 筛选的所有通道的摘要信息

以下 `list-channels` 示例列出您的 Amazon 账户中与指定 PlaybackRestrictionPolicy ARN 关联的所有通道。

```
aws ivs list-channels \
    --filter-by-playback-restriction-policy-arn "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ"
```

输出：

```
{
    "channels": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
            "name": "channel-2",
```

```
            "latencyMode": "LOW",
            "authorized": false,
            "preset": "",
            "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
            "recordingConfigurationArn": "",
            "tags": {},
            "type": "STANDARD"
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChannels。

`list-playback-key-pairs`

以下代码示例演示了如何使用 `list-playback-key-pairs`。

Amazon CLI

获取有关所有播放密钥对的摘要信息

以下 `list-playback-key-pairs` 示例返回所有密钥对的信息。

```
aws ivs list-playback-key-pairs
```

输出：

```
{
    "keyPairs": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
            "name": "test-key-0",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/ijkl5678mnop",
            "name": "test-key-1",
            "tags": {}
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPlaybackKeyPairs。

`list-playback-restriction-policies`

以下代码示例演示了如何使用 `list-playback-restriction-policies`。

Amazon CLI

获取有关所有播放限制策略的摘要信息

以下 `list-playback-restriction-policies` 示例列出您的 Amazon 账户的所有播放限制策略。

```
aws ivs list-playback-restriction-policies
```

输出：

```
{
    "playbackRestrictionPolicies": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
            "allowedCountries": [
                "US",
                "MX"
            ],
            "allowedOrigins": [
                "https://www.website1.com",
                "https://www.website2.com"
            ],
            "enableStrictOriginEnforcement": true,
            "name": "test-playback-restriction-policy",
            "tags": {
                "key1": "value1",
                "key2": "value2"
            }
        }
    ]
```

```
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPlaybackRestrictionPolicies。

`list-recording-configurations`

以下代码示例演示了如何使用 `list-recording-configurations`。

Amazon CLI

列出在此账户中创建的所 RecordingConfiguration 资源

以下 `list-recording-configurations` 示例获取您的账户中所有 RecordingConfiguration 资源的信息。

```
aws ivs list-recording-configurations
```

输出：

```
{
    "recordingConfigurations": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
            "name": "test-recording-config-1",
            "destinationConfiguration": {
                "s3": {
                    "bucketName": "demo-recording-bucket-1"
                }
            },
            "state": "ACTIVE",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
CD12abcdGHIJ",
            "name": "test-recording-config-2",
            "destinationConfiguration": {
                "s3": {
                    "bucketName": "demo-recording-bucket-2"
                }
```

```
            },
            "state": "ACTIVE",
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的录制到 Amazon S3。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRecordingConfigurations。

list-stream-keys

以下代码示例演示了如何使用 list-stream-keys。

Amazon CLI

获取直播密钥列表

以下 list-stream-keys 示例列出指定 ARN（Amazon 资源名称）的所有直播密钥。

```
aws ivs list-stream-keys \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

输出：

```
{
    "streamKeys": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreamKeys。

list-stream-sessions

以下代码示例演示了如何使用 list-stream-sessions。

Amazon CLI

获取当前 Amazon 区域中指定通道的当前和之前直播的摘要

以下 `list-stream-sessions` 示例报告指定通道 ARN（Amazon 资源名称）的直播的摘要信息。

```
aws ivs list-stream-sessions \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
    --max-results 25 \
    --next-token ""
```

输出：

```
{
    "nextToken": "set-2",
    "streamSessions": [
        {
            "startTime": 1641578182,
            "endTime": 1641579982,
            "hasErrorEvent": false,
            "streamId": "mystream"
        }
        ...
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreamSessions。

`list-streams`

以下代码示例演示了如何使用 `list-streams`。

Amazon CLI

获取直播列表及其状态

以下 `list-streams` 示例列出您的 Amazon 账户的所有直播。

```
aws ivs list-streams
```

输出：

```
{
   "streams": [
        {
            "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
            "state": "LIVE",
            "health": "HEALTHY",
            "streamId": "st-ABCDEfghij01234KLMN5678",
            "viewerCount": 1
        }
    ]
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreams。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 Amazon 资源（如通道、直播密钥）的所有标签

以下 `list-tags-for-resource` 示例列出指定资源 ARN（Amazon 资源名称）的所有标签。

```
aws ivs list-tags-for-resource \
    --resource-arn arn:aws:ivs:us-west-2:12345689012:channel/abcdABCDefgh
```

输出：

```
{
    "tags":
    {
        "key1": "value1",
        "key2": "value2"
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-metadata

以下代码示例演示了如何使用 put-metadata。

Amazon CLI

将元数据插入到指定通道的活跃直播中

以下 put-metadata 示例将给定元数据插入到指定通道的直播中。

```
aws ivs put-metadata \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
    --metadata '{"my": "metadata"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMetadata。

start-viewer-session-revocation

以下代码示例演示了如何使用 start-viewer-session-revocation。

Amazon CLI

撤消给定多 channel-ARN 和 viewer-ID 对的观看者会话

以下 start-viewer-session-revocation 示例启动撤销与指定通道 ARN 和观看者 ID 相关联的观看者会话的过程，撤销的会话直至且包括指定版本号。如果未提供版本，则默认值为 0。

```
aws ivs batch-start-viewer-session-revocation \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
    --viewer-id abcdefg \
    --viewer-session-versions-less-than-or-equal-to 1234567890
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的设置私有通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartViewerSessionRevocation。

stop-stream

以下代码示例演示了如何使用 stop-stream。

Amazon CLI

停止指定直播

以下 stop-stream 示例停止指定通道上的直播。

```
aws ivs stop-stream \
    --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

此命令不生成任何输出。

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopStream。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为 Amazon 资源（如通道、直播密钥）添加或更新标签

以下 tag-resource 示例为指定资源 ARN（Amazon 资源名称）添加或更新标签。

```
aws ivs tag-resource \
    --resource-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
    --tags "tagkey1=tagvalue1, tagkey2=tagvalue2"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除 Amazon 资源（如通道、直播密钥）的标签

以下 untag-resource 示例移除指定资源 ARN（Amazon 资源名称）的指定标签。

```
aws ivs untag-resource \
    --resource-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
    --tag-keys "tagkey1, tagkey2"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-channel

以下代码示例演示了如何使用 update-channel。

Amazon CLI

示例 1：更新通道的配置信息

以下 update-channel 示例更新了指定通道 ARN 的通道配置以更改通道名称。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --name 'channel-1' \
    --insecure-ingest
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "name": "channel-1",
        "latencyMode": "LOW",
        "containerFormat": "TS",
        "multitrackInputConfiguration": {
```

```
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": true,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的创建通道。

示例 2：更新通道配置以启用录制

以下 update-channel 示例更新了指定通道 ARN 的通道配置以启用录制。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --no-insecure-ingest \
    --recording-configuration-arn 'arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "name": "test-channel-with-recording",
        "latencyMode": "LOW",
        "containerFormat": "TS",
        "multitrackInputConfiguration": {
```

```
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "BA1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的录制到 Amazon S3。

示例 3：更新通道配置以禁用录制

以下 update-channel 示例更新了指定通道 ARN 的通道配置以禁用录制。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --recording-configuration-arn ''
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "name": "test-channel-with-recording",
        "latencyMode": "LOW",
        "containerFormat": "TS",
        "multitrackInputConfiguration": {
```

```
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2edfGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的录制到 Amazon S3。

示例 4：更新通道配置以启用播放限制

以下 update-channel 示例更新了指定通道 ARN 的通道配置以应用播放限制策略。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --no-insecure-ingest \
    --playback-restriction-policy-arn 'arn:aws:ivs:us-west-2:123456789012:playback-
restriction-policy/ABcdef34ghIJ'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "name": "test-channel-with-playback-restriction-policy",
        "latencyMode": "LOW",
        "containerFormat": "TS",
```

```
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaCBDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

示例 5：更新通道配置以禁用播放限制

以下 update-channel 示例更新了指定通道 ARN 的通道配置以禁用播放限制。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --playback-restriction-policy-arn ''
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "name": "test-channel-with-playback-restriction-policy",
        "latencyMode": "LOW",
        "containerFormat": "TS",
```

```
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDeFghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

示例 6：更新通道配置以启用多轨道

以下 update-channel 示例更新指定通道 ARN 的通道配置以启用多轨道。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --container-format 'FRAGMENTED_MP4' \
    --multitrack-input-configuration '{"enabled": true,"maximumResolution":
 "FULL_HD","policy": "ALLOW"}'
```

输出：

```
{
    "channel": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "containerFormat": "FRAGMENTED_MP4",
        "name": "test-channel-with-multitrack",
```

```
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": true,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaCBDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

示例 7：更新通道配置以禁用播放限制

以下 update-channel 示例更新指定通道 ARN 的通道配置以禁用多轨道。这不会影响此通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-channel \
    --arn 'arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh' \
    --container-format 'TS' \
    --multitrack-input-configuration '{"enabled": false}'
```

输出：

```
{
    "channel": {
        ""arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
        "containerFormat": "TS",
        "name": "test-channel-with-multitrack",
```

```
        "latencyMode": "LOW",
        "multitrackInputConfiguration": {
            "enabled": false,
            "maximumResolution": "FULL_HD",
            "policy": "ALLOW"
        },
        "type": "STANDARD",
        "playbackRestrictionPolicyArn": "",
        "recordingConfigurationArn": "",
        "srt": {
            "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
            "passphrase":
 "AB1C2defGHijkLMNo3PqQRstUvwxyzaCBDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
        },
        "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
        "insecureIngest": false,
        "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
        "preset": "",
        "authorized": false,
        "tags": {}
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateChannel。

update-playback-restriction-policy

以下代码示例演示了如何使用 update-playback-restriction-policy。

Amazon CLI

更新播放限制策略

以下 update-playback-restriction-policy 示例使用指定的策略 ARN 更新播放限制策略，以禁用严格的来源强制执行。这不会影响关联通道正在进行的直播；您必须停止并重新启动直播才能使更改生效。

```
aws ivs update-playback-restriction-policy \
    --arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ" \
```

```
    --no-enable-strict-origin-enforcement
```

输出：

```
{
    "playbackRestrictionPolicy": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
        "allowedCountries": [
            "US",
            "MX"
        ],
        "allowedOrigins": [
            "https://www.website1.com",
            "https://www.website2.com"
        ],
        "enableStrictOriginEnforcement": false,
        "name": "test-playback-restriction-policy",
        "tags": {
            "key1": "value1",
            "key2": "value2"
        }
    }
}
```

有关更多信息，请参阅《IVS Low-Latency 用户指南》中的不想要的内容和观看者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePlaybackRestrictionPolicy。

使用 Amazon CLI 的 Amazon IVS 聊天功能

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IVS 聊天功能结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-chat-token`

以下代码示例演示了如何使用 `create-chat-token`。

Amazon CLI

创建聊天令牌

以下 `create-chat-token` 示例创建一个加密聊天令牌，用于建立与一个聊天室的单个 WebSocket 连接。该令牌的有效期为一分钟，通过该令牌建立的连接（会话）在指定期间有效。

```
aws ivschat create-chat-token \
    --roomIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6", \
    --userId" "11231234" \
    --capabilities "SEND_MESSAGE", \
    --sessionDurationInMinutes" 30
```

输出：

```
{
    "token": "ACEGmnoq#1rstu2...BDFH3vxwy!4hlm!#5",
    "sessionExpirationTime": "2022-03-16T04:44:09+00:00"
    "state": "CREATING",
    "tokenExpirationTime": "2022-03-16T03:45:09+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的步骤 3：对聊天客户端进行身份验证和授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChatToken。

`create-logging-configuration`

以下代码示例演示了如何使用 `create-logging-configuration`。

Amazon CLI

创建聊天 LoggingConfiguration 资源

以下 `create-logging-configuration` 示例创建一个 LoggingConfiguration 资源，该资源允许客户端存储和记录发送的消息。

```
aws ivschat create-logging-configuration \
    --destination-configuration s3={bucketName=demo-logging-bucket} \
    --name "test-logging-config" \
    --tags "key1=value1, key2=value2"
```

输出：

```
{
    "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ",
    "createTime": "2022-09-14T17:48:00.653000+00:00",
    "destinationConfiguration": {
        "s3": {
            "bucketName": "demo-logging-bucket"
        }
    },
    "id": "ABcdef34ghIJ",
    "name": "test-logging-config",
    "state": "ACTIVE",
    "tags": { "key1" : "value1", "key2" : "value2" },
    "updateTime": "2022-09-14T17:48:01.104000+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoggingConfiguration。

create-room

以下代码示例演示了如何使用 create-room。

Amazon CLI

创建聊天室

以下 create-room 示例创建一个新的聊天室。

```
aws ivschat create-room \
    --name "test-room-1" \
    --logging-configuration-identifiers "arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ" \
```

```
    --maximum-message-length 256 \
    --maximum-message-rate-per-second 5
```

输出：

```
{
    "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",
    "id": "g1H2I3j4k5L6",
    "createTime": "2022-03-16T04:44:09+00:00",
    "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ"],
    "maximumMessageLength": 256,
    "maximumMessageRatePerSecond": 5,
    "name": "test-room-1",
    "tags": {}
    "updateTime": "2022-03-16T07:22:09+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的步骤 2：创建聊天室。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRoom。

`delete-logging-configuration`

以下代码示例演示了如何使用 `delete-logging-configuration`。

Amazon CLI

删除聊天 LoggingConfiguration 资源

以下 `delete-logging-configuration` 示例删除指定 ARN 的 LoggingConfiguration 资源。

```
aws ivschat delete-logging-configuration \
    --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoggingConfiguration。

delete-message

以下代码示例演示了如何使用 delete-message。

Amazon CLI

从指定聊天室删除消息

以下 delete-message 示例发送一个事件给指定聊天室，指示客户端删除指定的消息，也就是说，取消呈现该消息，并将其从客户端的聊天记录中删除。

```
aws ivschat delete-message \
    --roomIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \
    --id "ABC123def456" \
    --reason "Message contains profanity"
```

输出：

```
{
    "id": "12345689012"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMessage。

delete-room

以下代码示例演示了如何使用 delete-room。

Amazon CLI

删除聊天室

以下 delete-room 示例删除指定聊天室。连接的客户端断开连接。成功后，它将返回带有空响应正文的 HTTP 204。

```
aws ivschat delete-room \
    --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRoom。

`disconnect-user`

以下代码示例演示了如何使用 `disconnect-user`。

Amazon CLI

断开用户与聊天室的连接

以下 `disconnect-user` 示例断开指定用户与指定聊天室的所有连接。成功后，它将返回带有空响应正文的 HTTP 200。

```
aws ivschat disconnect-user \
    --roomIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \
    --userId "ABC123def456" \
    --reason "Violated terms of service"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisconnectUser。

`get-logging-configuration`

以下代码示例演示了如何使用 `get-logging-configuration`。

Amazon CLI

获取有关 LoggingConfiguration 资源的信息

以下 `get-logging-configuration` 示例删除指定 ARN 的 LoggingConfiguration 资源。

```
aws ivschat get-logging-configuration \
```

```
    --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ"
```

输出：

```
{
    "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ",
    "createTime": "2022-09-14T17:48:00.653000+00:00",
    "destinationConfiguration": {
        "s3": {
            "bucketName": "demo-logging-bucket"
        }
    },
    "id": "ABcdef34ghIJ",
    "name": "test-logging-config",
    "state": "ACTIVE",
    "tags": { "key1" : "value1", "key2" : "value2" },
    "updateTime": "2022-09-14T17:48:01.104000+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoggingConfiguration。

get-room

以下代码示例演示了如何使用 get-room。

Amazon CLI

获取指定聊天室

以下 get-room 示例获取有关指定聊天室的信息。

```
aws ivschat get-room \
    --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6"
```

输出：

```
{
```

```
    "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",
    "createTime": "2022-03-16T04:44:09+00:00",
    "id": "g1H2I3j4k5L6",
    "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ"],
    "maximumMessageLength": 256,
    "maximumMessageRatePerSecond": 5,
    "name": "test-room-1",
    "tags": {},
    "updateTime": "2022-03-16T07:22:09+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRoom。

`list-logging-configurations`

以下代码示例演示了如何使用 `list-logging-configurations`。

Amazon CLI

获取处理 API 请求的 Amazon 区域中用户的所有日志配置的摘要信息

以下 `list-logging-configurations` 示例列出处理 API 请求的 Amazon 区域中用户的所有 LoggingConfiguration 资源的信息。

```
aws ivschat list-logging-configurations \
    --max-results 2 \
    --next-token ""
```

输出：

```
{
    "nextToken": "set-2",
    "loggingConfigurations": [
        {
            "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ",
            "createTime": "2022-09-14T17:48:00.653000+00:00",
            "destinationConfiguration": {
```

```
                "s3": {
                    "bucketName": "demo-logging-bucket"
                }
            },
            "id": "ABcdef34ghIJ",
            "name": "test-logging-config",
            "state": "ACTIVE",
            "tags": { "key1" : "value1", "key2" : "value2" },
            "updateTime": "2022-09-14T17:48:01.104000+00:00"
        }
        ...
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLoggingConfigurations。

`list-rooms`

以下代码示例演示了如何使用 `list-rooms`。

Amazon CLI

获取当前区域内有关您的所有聊天室的摘要信息

以下 `list-rooms` 示例获取处理请求的 Amazon 区域中所有聊天室的摘要信息。结果按 updateTime 降序排序。

```
aws ivschat list-rooms \
    --logging-configuration-identifier "arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ" \
    --max-results 10 \
    --next-token ""
```

输出：

```
{
    "nextToken": "page3",
    "rooms": [
        {
```

```
            "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",
            "createTime": "2022-03-16T04:44:09+00:00",
            "id": "g1H2I3j4k5L6",
            "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ"],
            "name": "test-room-1",
            "tags": {},
            "updateTime": "2022-03-16T07:22:09+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRooms。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 Amazon 资源（如聊天室）的所有标签

以下 `list-tags-for-resource` 示例列出指定资源 ARN（Amazon 资源名称）的所有标签。

```
aws ivschat list-tags-for-resource \
    --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6
```

输出：

```
{
    "tags":
    {
        "key1": "value1",
        "key2": "value2"
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

send-event

以下代码示例演示了如何使用 send-event。

Amazon CLI

将事件发送到聊天室

以下 send-event 示例将给定事件发送到指定的聊天室。

```
aws ivschat send-event \
    --roomIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \
    --eventName "SystemMessage" \
    --attributes \
        "msgType"="user-notification", \
        "msgText"="This chat room will close in 15 minutes."
```

输出：

```
{
    "id": "12345689012"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendEvent。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为 Amazon 资源（如聊天室）添加或更新标签

以下 tag-resource 示例为指定资源 ARN（Amazon 资源名称）添加或更新标签。成功后，它将返回带有空响应正文的 HTTP 200。

```
aws ivschat tag-resource \
    --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6 \
    --tags "tagkey1=tagkeyvalue1, tagkey2=tagkeyvalue2"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除 Amazon 资源（如聊天室）的标签

以下 untag-resource 示例移除指定资源 ARN（Amazon 资源名称）的指定标签。成功后，它将返回带有空响应正文的 HTTP 200。

```
aws ivschat untag-resource \
    --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6 \
    --tag-keys "tagkey1, tagkey2"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service API 参考》中的标记。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-logging-configuration

以下代码示例演示了如何使用 update-logging-configuration。

Amazon CLI

更新聊天室的日志配置

以下 update-logging-configuration 示例使用给定数据更新 LoggingConfiguration 资源。

```
aws ivschat update-logging-configuration \
    --destination-configuration s3={bucketName=demo-logging-bucket} \
    --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ" \
    --name "test-logging-config"
```

输出：

```
{
    "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ",
    "createTime": "2022-09-14T17:48:00.653000+00:00",
    "destinationConfiguration": {
        "s3": {
            "bucketName": "demo-logging-bucket"
        }
    },
    "id": "ABcdef34ghIJ",
    "name": "test-logging-config",
    "state": "ACTIVE",
    "tags": { "key1" : "value1", "key2" : "value2" },
    "updateTime": "2022-09-14T17:48:01.104000+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLoggingConfiguration。

update-room

以下代码示例演示了如何使用 update-room。

Amazon CLI

更新聊天室配置

以下 update-room 示例使用给定数据更新指定聊天室的配置。

```
aws ivschat update-room \
    --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \
    --logging-configuration-identifiers "arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ" \
    --name "chat-room-a" \
    --maximum-message-length 256 \
    --maximum-message-rate-per-second 5
```

输出：

```
{
    "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",
    "createTime": "2022-03-16T04:44:09+00:00",
    "id": "g1H2I3j4k5L6",
    "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ"],
    "maximumMessageLength": 256,
    "maximumMessageRatePerSecond": 5,
    "name": "chat-room-a",
    "tags": {},
    "updateTime": "2022-03-16T07:22:09+00:00"
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 Amazon IVS 聊天功能入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRoom。

使用 Amazon CLI 的 Amazon IVS 实时直播功能示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon IVS 实时直播功能结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-encoder-configuration`

以下代码示例演示了如何使用 `create-encoder-configuration`。

Amazon CLI

创建合成编码器配置

以下 create-encoder-configuration 示例使用指定属性创建一个合成编码器配置。

```
aws ivs-realtime create-encoder-configuration \
    --name test-ec --video bitrate=3500000,framerate=30.0,height=1080,width=1920
```

输出：

```
{
    "encoderConfiguration": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
ABabCDcdEFef",
        "name": "test-ec",
        "tags": {},
        "video": {
            "bitrate": 3500000,
            "framerate": 30,
            "height": 1080,
            "width": 1920
        }
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEncoderConfiguration。

create-ingest-configuration

以下代码示例演示了如何使用 create-ingest-configuration。

Amazon CLI

创建摄取配置

以下 create-ingest-configuration 示例使用 RTMPS 协议创建摄取配置。

```
aws ivs-realtime create-ingest-configuration \
    --name ingest1 \
    --ingest-protocol rtmps
```

输出：

```
{
    "ingestConfiguration": {
        "name": "ingest1",
        "arn": "arn:aws:ivs:us-west-2:123456789012:ingest-configuration/
AbCdEfGh1234",
        "ingestProtocol": "RTMPS",
        "streamKey": "rt_123456789012_us-
west-2_AbCdEfGh1234_abcd1234efgh5678ijkl9012MNOP34",
        "stageArn": "",
        "participantId": "xyZ654abC321",
        "state": "INACTIVE",
        "userId": "",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateIngestConfiguration。

create-participant-token

以下代码示例演示了如何使用 create-participant-token。

Amazon CLI

创建阶段参与者令牌

以下 create-participant-token 示例为指定阶段创建参与者令牌。

```
aws ivs-realtime create-participant-token \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --user-id bob
```

输出：

```
{
    "participantToken": {
        "expirationTime": "2023-03-07T09:47:43+00:00",
        "participantId": "ABCDEfghij01234KLMN6789",
```

```
        "token": "abcd1234defg5678"
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateParticipantToken。

create-stage

以下代码示例演示了如何使用 create-stage。

Amazon CLI

示例 1：创建阶段

以下 create-stage 示例为指定用户创建阶段和阶段参与者令牌。

```
aws ivs-realtime create-stage \
    --name stage1 \
    --participant-token-configurations userId=alice
```

输出：

```
{
    "participantTokens": [
        {
            "participantId": "ABCDEfghij01234KLMN5678",
            "token": "a1b2c3d4567890ab",
            "userId": "alice"
        }
    ],
    "stage": {
        "activeSessionId": "st-a1b2c3d4e5f6g",
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "storageConfigurationArn": "",
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
```

```
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "DISABLED"
            },
            "recordingReconnectWindowSeconds": 0,
            "hlsConfiguration": {
                "targetSegmentDurationSeconds": 6
            },
            "recordParticipantReplicas": true
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://9x0y8z7s6t5u.global-bm.whip.live-video.net"
        },
        "name": "stage1",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 2：创建阶段并配置单个参与者录制

以下 `create-stage` 示例创建一个阶段并配置单个参与者录制。

```
aws ivs-realtime create-stage \
    --name stage1 \
    --auto-participant-recording-configuration '{"mediaTypes":
 ["AUDIO_VIDEO"],"storageConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:storage-configuration/abcdABCDefgh",
 "recordingReconnectWindowSeconds": 100, \
        "hlsConfiguration": {"targetSegmentDurationSeconds": 5}}'
```

输出：

```
{
    "stage": {
```

```
        "activeSessionId": "st-a1b2c3d4e5f6g",
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "storageConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:storage-
configuration/abcdABCDefgh"
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "DISABLED"
            },
            "recordingReconnectWindowSeconds": 100,
            "hlsConfiguration": {
                "targetSegmentDurationSeconds": 5
            },
            "recordParticipantReplicas": true
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://9x0y8z7s6t5u.global-bm.whip.live-video.net"
        },
        "name": "stage1",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 3：创建阶段并配置启用缩略图录制的单个参与者录制

以下 `create-stage` 示例创建一个阶段，并配置启用缩略图录制的单个参与者录制。

```
aws ivs-realtime create-stage \
    --name stage1 \
```

```
    --auto-participant-recording-configuration '{"mediaTypes":
 ["AUDIO_VIDEO"],"storageConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:storage-configuration/abcdABCDefgh", \
        "thumbnailConfiguration": {"recordingMode": "INTERVAL","storage":
 ["SEQUENTIAL"],"targetIntervalSeconds": 60}}'
```

输出：

```
{
    "stage": {
        "activeSessionId": "st-a1b2c3d4e5f6g",
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "storageConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:storage-
configuration/abcdABCDefgh",
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "INTERVAL"
            },
            "recordingReconnectWindowSeconds": 0,
            "hlsConfiguration": {
                "targetSegmentDurationSeconds": 6
            },
            "recordParticipantReplicas": true
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://9x0y8z7s6t5u.global-bm.whip.live-video.net"
        },
        "name": "stage1",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStage。

create-storage-configuration

以下代码示例演示了如何使用 create-storage-configuration。

Amazon CLI

创建合成存储配置

以下 create-storage-configuration 示例使用指定属性创建一个合成存储配置。

```
aws ivs-realtime create-storage-configuration \
    --name "test-sc" --s3 "bucketName=amzn-s3-demo-bucket"
```

输出：

```
{
    "storageConfiguration": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
ABabCDcdEFef",
        "name": "test-sc",
        "s3": {
            "bucketName": "amzn-s3-demo-bucket"
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStorageConfiguration。

delete-encoder-configuration

以下代码示例演示了如何使用 delete-encoder-configuration。

Amazon CLI

删除合成编码器配置

以下 delete-encoder-configuration 命令删除由给定 ARN（Amazon 资源名称）指定的合成编码器配置。

```
aws ivs-realtime delete-encoder-configuration \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
ABabCDcdEFef"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEncoderConfiguration。

delete-ingest-configuration

以下代码示例演示了如何使用 delete-ingest-configuration。

Amazon CLI

示例 1：删除非活动摄取配置

以下 delete-ingest-configuration 示例删除指定摄取配置 ARN（Amazon 资源名称）的非活动摄取配置。

```
aws ivs-realtime delete-ingest-configuration \
    --arn arn:aws:ivs:us-west-2:123456789012:ingest-configuration/AbCdEfGh1234
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

示例 2：强制删除活动摄取配置

以下 delete-ingest-configuration 示例强制删除指定摄取配置 ARN（Amazon 资源名称）的活动摄取配置。

```
aws ivs-realtime delete-ingest-configuration \
```

```
    --arn arn:aws:ivs:us-west-2:123456789012:ingest-configuration/AbCdEfGh1234 \
    --force
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteIngestConfiguration。

delete-public-key

以下代码示例演示了如何使用 delete-public-key。

Amazon CLI

删除公钥

以下 delete-public-key 命令删除指定公钥。

```
aws ivs-realtime delete-public-key \
    --arn arn:aws:ivs:us-west-2:123456789012:public-key/abcdABC1efg2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon IVS 实时直播功能用户指南》中的分发参与者令牌。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePublicKey。

delete-stage

以下代码示例演示了如何使用 delete-stage。

Amazon CLI

删除阶段

以下 delete-stage 示例删除指定阶段。

```
aws ivs-realtime delete-stage \
    --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStage。

`delete-storage-configuration`

以下代码示例演示了如何使用 `delete-storage-configuration`。

Amazon CLI

删除合成存储配置

以下 `delete-storage-configuration` 命令删除由给定 ARN（Amazon 资源名称）指定的合成存储配置。

```
aws ivs-realtime delete-storage-configuration \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
ABabCDcdEFef"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStorageConfiguration。

`disconnect-participant`

以下代码示例演示了如何使用 `disconnect-participant`。

Amazon CLI

断开阶段参与者的连接

以下 `disconnect-participant` 示例断开指定参与者与指定阶段的连接。

```
aws ivs-realtime disconnect-participant \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --participant-id ABCDEfghij01234KLMN5678
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisconnectParticipant。

get-composition

以下代码示例演示了如何使用 get-composition。

Amazon CLI

示例 1：获取采用默认布局设置的合成

以下 get-composition 示例获取指定 ARN（Amazon 资源名称）的合成。

```
aws ivs-realtime get-composition \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh"
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "startTime": "2023-10-16T23:26:00+00:00",
                "state": "ACTIVE"
            },
            {
                "configuration": {
                    "name": "",
```

```
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE",
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "startTime": "2023-10-16T23:26:00+00:00",
                "state": "STARTING"
            }
        ],
        "layout": {
            "grid": {
                "featuredParticipantAttribute": "",
                "gridGap": 2,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "",
                "videoAspectRatio": "VIDEO",
                "videoFillMode": ""
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
        "state": "ACTIVE",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

示例 2：获取采用 PiP 布局的合成

以下 `get-composition` 示例获取采用 PiP 布局的指定 ARN（Amazon 资源名称）的合成。

```
aws ivs-realtime get-composition \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs"
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "startTime": "2023-10-16T23:26:00+00:00",
                "state": "ACTIVE"
            },
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
```

```
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "startTime": "2023-10-16T23:26:00+00:00",
                "state": "STARTING"
            }
        ],
        "layout": {
            "pip": {
                "featuredParticipantAttribute": "abcdefg",
                "gridGap": 0,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "",
                "pipBehavior": "STATIC",
                "pipOffset": 0,
                "pipParticipantAttribute": "",
                "pipPosition": "BOTTOM_RIGHT",
                "videoFillMode": "COVER"
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
        "state": "ACTIVE",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

示例 3：在启用缩略图录制的情况下获取合成

以下 `get-composition` 示例获取指定的 ARN（Amazon 资源名称）的合成，该 ARN 已使用默认设置启用缩略图录制。

```
aws ivs-realtime get-composition \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh"
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "startTime": "2023-10-16T23:26:00+00:00",
                "state": "ACTIVE"
            },
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE",
                        "thumbnailConfigurations": [
                           {
                              "targetIntervalSeconds": 60,
                              "storage": [
```

```
                                "SEQUENTIAL"
                            ],
                        }
                    ]
                }
            },
            "detail": {
                "s3": {
                    "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                }
            },
            "id": "GHFabcgefABC",
            "startTime": "2023-10-16T23:26:00+00:00",
            "state": "STARTING"
        }
    ],
    "layout": {
        "grid": {
            "featuredParticipantAttribute": ""
            "gridGap": 2,
            "omitStoppedVideo": false,
            "participantOrderAttribute": "",
            "videoAspectRatio": "VIDEO",
            "videoFillMode": ""                }
    },
    "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
    "startTime": "2023-10-16T23:24:00+00:00",
    "state": "ACTIVE",
    "tags": {}
  }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetComposition。

get-encoder-configuration

以下代码示例演示了如何使用 get-encoder-configuration。

Amazon CLI

获取合成编码器配置

以下 get-encoder-configuration 示例获取由给定 ARN（Amazon 资源名称）指定的合成编码器配置。

```
aws ivs-realtime get-encoder-configuration \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
abcdABCDefgh"
```

输出：

```
{
    "encoderConfiguration": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
abcdABCDefgh",
        "name": "test-ec",
        "tags": {},
        "video": {
            "bitrate": 3500000,
            "framerate": 30,
            "height": 1080,
            "width": 1920
        }
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEncoderConfiguration。

get-ingest-configuration

以下代码示例演示了如何使用 get-ingest-configuration。

Amazon CLI

获取摄取配置信息

以下 get-ingest-configuration 示例获取指定摄取配置 ARN（Amazon 资源名称）的摄取配置。

```
aws ivs-realtime get-ingest-configuration \
    --arn arn:aws:ivs:us-west-2:123456789012:ingest-configuration/AbCdEfGh1234
```

输出：

```
{
    "ingestConfiguration": {
        "name": "ingest1",
        "arn": "arn:aws:ivs:us-west-2:123456789012:ingest-configuration/
AbCdEfGh1234",
        "ingestProtocol": "RTMPS",
        "streamKey": "rt_123456789012_us-
west-2_AbCdEfGh1234_abcd1234efgh5678ijkl9012MNOP34",
        "stageArn": "",
        "participantId": "xyZ654abC321",
        "state": "INACTIVE",
        "userId": "",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetIngestConfiguration。

get-participant

以下代码示例演示了如何使用 get-participant。

Amazon CLI

示例 1：获取阶段参与者

以下 get-participant 示例获取指定阶段 ARN（Amazon 资源名称）中指定参与者 ID 和会话 ID 的阶段参与者。

```
aws ivs-realtime get-participant \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g \
```

```
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "participant": {
        "browserName": "Google Chrome",
        "browserVersion": "116",
        "firstJoinTime": "2023-04-26T20:30:34+00:00",
        "ispName": "Comcast",
        "osName": "Microsoft Windows 10 Pro",
        "osVersion": "10.0.19044",
        "participantId": "abCDEf12GHIj",
        "published": true,
        "recordingS3BucketName": "bucket-name",
        "recordingS3Prefix": "abcdABCDefgh/st-a1b2c3d4e5f6g/
abCDEf12GHIj/1234567890",
        "recordingState": "ACTIVE",
        "sdkVersion": "",
        "state": "CONNECTED",
        "userId": ""
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 2：获取已复制到另一个阶段的阶段参与者

以下 `get-participant` 示例当指定阶段 ARN（Amazon 资源名称）中指定参与者 ID 和会话 ID 的阶段参与者也被复制到另一个阶段时，获取该阶段参与者。

```
aws ivs-realtime get-participant \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "participant": {
        "browserName": "Google Chrome",
```

```
        "browserVersion": "116",
        "firstJoinTime": "2023-04-26T20:30:34+00:00",
        "ispName": "Comcast",
        "osName": "Microsoft Windows 10 Pro",
        "osVersion": "10.0.19044",
        "participantId": "abCDEf12GHIj",
        "published": true,
        "recordingS3BucketName": "bucket-name",
        "recordingS3Prefix": "abcdABCDefgh/st-a1b2c3d4e5f6g/
abCDEf12GHIj/1234567890",
        "recordingState": "ACTIVE",
        "replicationState": "ACTIVE",
        "replicationType": "SOURCE",
        "sdkVersion": "",
        "state": "CONNECTED",
        "userId": ""
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 3：获取已从另一个阶段复制的阶段参与者

以下 `get-participant` 示例当从另一个阶段复制指定阶段 ARN（Amazon 资源名称）中指定参与者 ID 和会话 ID 的阶段参与者时，获取该阶段参与者。

```
aws ivs-realtime get-participant \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "participant": {
        "browserName": "Google Chrome",
        "browserVersion": "116",
        "firstJoinTime": "2023-04-26T20:30:34+00:00",
        "ispName": "Comcast",
        "osName": "Microsoft Windows 10 Pro",
        "osVersion": "10.0.19044",
```

```
        "participantId": "abCDEf12GHIj",
        "published": true,
        "recordingS3BucketName": "bucket-name",
        "recordingS3Prefix": "abcdABCDefgh/st-a1b2c3d4e5f6g/
abCDEf12GHIj/1234567890",
        "recordingState": "ACTIVE",
        "replicationState": "ACTIVE",
        "replicationType": "REPLICA",
        "sdkVersion": "",
        "state": "CONNECTED",
        "userId": ""
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetParticipant。

get-public-key

以下代码示例演示了如何使用 get-public-key。

Amazon CLI

获取用于签署阶段参与者令牌的现有公钥

以下 get-public-key 示例获取由提供的 ARN 指定的公钥，用于签署阶段参与者令牌。

```
aws ivs-realtime get-public-key \
    --arn arn:aws:ivs:us-west-2:123456789012:public-key/abcdABC1efg2
```

输出：

```
{
    "publicKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:public-key/abcdABC1efg2",
        "name": "",
        "publicKeyMaterial": "-----BEGIN PUBLIC KEY-----
\nMHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEqVWUtqs6EktQMR1sCYmEzGvRwtaycI16\n9pmzcpiWu/
uhNStGlteJ5odRfRwVkoQUMnSZXTCcbn9bBTTmiWo4mJcFOOAzsthH
\n0UAb8NdD4tUE0At4a9hYP9IETEXAMPLE\n-----END PUBLIC KEY-----",
```

```
        "fingerprint": "12:a3:44:56:bc:7d:e8:9f:10:2g:34:hi:56:78:90:12",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS 实时直播功能用户指南》中的分发参与者令牌。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicKey。

get-stage-session

以下代码示例演示了如何使用 get-stage-session。

Amazon CLI

获取阶段会话

以下 get-stage-session 示例获取指定阶段 ARN（Amazon 资源名称）的指定会话 ID 的阶段会话。

```
aws ivs-realtime get-stage-session \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g
```

输出：

```
{
    "stageSession": {
        "endTime": "2023-04-26T20:36:29+00:00",
        "sessionId": "st-a1b2c3d4e5f6g",
        "startTime": "2023-04-26T20:30:29.602000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStageSession。

get-stage

以下代码示例演示了如何使用 get-stage。

Amazon CLI

获取阶段的配置信息

以下 get-stage 示例获取指定阶段 ARN（Amazon 资源名称）的阶段配置。

```
aws ivs-realtime get-stage \
    --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

输出：

```
{
    "stage": {
        "activeSessionId": "st-a1b2c3d4e5f6g",
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "storageConfigurationArn": "",
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "DISABLED"
            },
            "recordingReconnectWindowSeconds": 0,
            "hlsConfiguration": {
                "targetSegmentDurationSeconds": 6
            },
            "recordParticipantReplicas": true
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://9x0y8z7s6t5u.global-bm.whip.live-video.net"
        },
        "name": "test",
        "tags": {}
```

```
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStage。

`get-storage-configuration`

以下代码示例演示了如何使用 `get-storage-configuration`。

Amazon CLI

获取合成存储配置

以下 `get-storage-configuration` 示例获取由给定 ARN（Amazon 资源名称）指定的合成存储配置。

```
aws ivs-realtime get-storage-configuration \
    --name arn "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
abcdABCDefgh"
```

输出：

```
{
    "storageConfiguration": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
abcdABCDefgh",
        "name": "test-sc",
        "s3": {
            "bucketName": "amzn-s3-demo-bucket"
        },
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStorageConfiguration。

import-public-key

以下代码示例演示了如何使用 import-public-key。

Amazon CLI

导入现有公钥以用于签署阶段参与者令牌

以下 import-public-key 示例从材料文件中导入公钥，用于签署阶段参与者令牌。

```
aws ivs-realtime import-public-key \
    --public-key-material="`cat public.pem`"
```

输出：

```
{
    "publicKey": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:public-key/abcdABC1efg2",
        "name": "",
        "publicKeyMaterial": "-----BEGIN PUBLIC KEY-----
\nMHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEqVWUtqs6EktQMR1sCYmEzGvRwtaycI16\n9pmzcpiWu/
uhNStGlteJ5odRfRwVkoQUMnSZXTCcbn9bBTTmiWo4mJcFOOAzsthH
\n0UAb8NdD4tUE0At4a9hYP9IETEXAMPLE\n-----END PUBLIC KEY-----",
        "fingerprint": "12:a3:44:56:bc:7d:e8:9f:10:2g:34:hi:56:78:90:12",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS 实时直播功能用户指南》中的分发参与者令牌。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportPublicKey。

list-compositions

以下代码示例演示了如何使用 list-compositions。

Amazon CLI

获取合成列表

以下 list-compositions 命令列出在处理 API 请求的 Amazon 区域内，您的 Amazon 账户的所有合成。

```
aws ivs-realtime list-compositions
```

输出：

```
{
    "compositions": [
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/
abcdABCDefgh",
            "destinations": [
                {
                    "id": "AabBCcdDEefF",
                    "startTime": "2023-10-16T23:25:23+00:00",
                    "state": "ACTIVE"
                }
            ],
            "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/
defgABCDabcd",
            "startTime": "2023-10-16T23:25:21+00:00",
            "state": "ACTIVE",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/
ABcdabCDefgh",
            "destinations": [
                {
                    "endTime": "2023-10-16T23:25:00.786512+00:00",
                    "id": "aABbcCDdeEFf",
                    "startTime": "2023-10-16T23:24:01+00:00",
                    "state": "STOPPED"
                },
                {
                    "endTime": "2023-10-16T23:25:00.786512+00:00",
                    "id": "deEFfaABbcCD",
                    "startTime": "2023-10-16T23:24:01+00:00",
                    "state": "STOPPED"
                }
            ],
            "endTime": "2023-10-16T23:25:00+00:00",
            "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/
efghabcdABCD",
            "startTime": "2023-10-16T23:24:00+00:00",
```

```
            "state": "STOPPED",
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCompositions。

`list-encoder-configurations`

以下代码示例演示了如何使用 `list-encoder-configurations`。

Amazon CLI

列出合成编码器配置

以下 `list-encoder-configurations` 命令列出在处理 API 请求的 Amazon 区域内，您的 Amazon 账户的所有合成编码器配置。

```
aws ivs-realtime list-encoder-configurations
```

输出：

```
{
    "encoderConfigurations": [
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
abcdABCDefgh",
            "name": "test-ec-1",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
ABCefgEFGabc",
            "name": "test-ec-2",
            "tags": {}
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEncoderConfigurations。

`list-ingest-configurations`

以下代码示例演示了如何使用 `list-ingest-configurations`。

Amazon CLI

获取有关所有摄取配置的摘要信息

以下 `list-ingest-configurations` 示例列出您的 Amazon 账户在处理 API 请求的 Amazon 区域中的所有摄取配置。

```
aws ivs-realtime list-ingest-configurations
```

输出：

```
{
    "ingestConfigurations": [
        {
            "name": "",
            "arn": "arn:aws:ivs:us-west-2:123456789012:ingest-configuration/
XYZuvwSt4567",
            "ingestProtocol": "RTMPS",
            "stageArn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
            "participnatId": "abC789Xyz456",
            "state": "INACTIVE"
            "userId": "",
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListIngestConfigurations。

list-participant-events

以下代码示例演示了如何使用 list-participant-events。

Amazon CLI

示例 1：获取阶段参与者事件的列表

以下 list-participant-events 示例列出指定阶段 ARN（Amazon 资源名称）的指定参与者 ID 和会话 ID 的所有参与者事件。

```
aws ivs-realtime list-participant-events \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "events": [
        {
            "eventTime": "2023-04-26T20:36:28+00:00",
            "name": "LEFT",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2023-04-26T20:36:28+00:00",
            "name": "PUBLISH_STOPPED",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2023-04-26T20:30:34+00:00",
            "name": "JOINED",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2023-04-26T20:30:34+00:00",
            "name": "PUBLISH_STARTED",
            "participantId": "abCDEf12GHIj"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 2：获取阶段参与者事件的列表，包括参与者复制停止和启动

以下 list-participant-events 示例列出指定阶段 ARN（Amazon 资源名称）的指定会话 ID 的所有参与者事件，其中将某个参与者复制到另一个阶段。

```
aws ivs-realtime list-participant-events \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "events": [
        {
            "eventTime": "2025-04-26T20:36:28+00:00",
            "name": "LEFT",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2025-04-26T20:36:28+00:00",
            "name": "PUBLISH_STOPPED",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2025-04-26T20:30:34+00:00",
            "name": "JOINED",
            "participantId": "abCDEf12GHIj"
        },
        {
            "eventTime": "2025-04-26T20:30:34+00:00",
            "name": "PUBLISH_STARTED",
            "participantId": "abCDEf12GHIj"
        },
        {
            "name": "REPLICATION_STARTED",
            "participantId": "abCDEf12GHIj",
            "eventTime": "2025-04-26T20:30:34+00:00",
            "destinationStageArn": "arn:aws:ivs:us-west-2:12345678901:stage/
ABCDabcdefgh",
```

```
            "destinationSessionId": "st-b1c2d3e4f5g6a"
        },
        {
            "name": "REPLICATION_STOPPED",
            "participantId": "abCDEf12GHIj",
            "eventTime": "2025-04-26T20:32:34+00:00",
            "destinationStageArn": "arn:aws:ivs:us-west-2:12345678901:stage/
ABCDabcdefgh",
            "destinationSessionId": "st-b1c2d3e4f5g6a"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListParticipantEvents。

`list-participant-replicas`

以下代码示例演示了如何使用 `list-participant-replicas`。

Amazon CLI

获取阶段参与者副本的列表

以下 `list-participant-replicas` 示例列出从指定的源阶段 ARN（Amazon 资源名称）复制到另一个阶段的所有阶段参与者。

```
aws ivs-realtime list-participant-replicas \
    --source-stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "replicas": [
        {
            "sourceStageArn": "arn:aws:ivs:us-west-2:123456789012:stage/
abcdABCDefgh",
            "participantId": "abCDEf12GHIj",
            "sourceSessionId": "st-a1b2c3d4e5f6g",
```

```
            "destinationStageArn": "arn:aws:ivs:us-west-2:012345678901:stage/
ABCDabcdefgh",
            "destinationSessionId": "st-b1c2d3e4f5g6a",
            "replicationState": "ACTIVE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Real-Time Streaming User Guide》中的 IVS 参与者复制。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListParticipantReplicas。

`list-participants`

以下代码示例演示了如何使用 `list-participants`。

Amazon CLI

示例 1：获取阶段参与者的列表

以下 `list-participants` 示例列出指定阶段 ARN（Amazon 资源名称）的指定会话 ID 的所有参与者。

```
aws ivs-realtime list-participants \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g
```

输出：

```
{
    "participants": [
        {
            "firstJoinTime": "2023-04-26T20:30:34+00:00",
            "participantId": "abCDEf12GHIj",
            "published": true,
            "recordingState": "STOPPED",
            "state": "DISCONNECTED",
            "userId": ""
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 2：当参与者被复制到另一个阶段时，获取阶段参与者的列表

以下 `list-participants` 示例当将某个参与者复制到另一个阶段时，列出指定阶段 ARN（Amazon 资源名称）的指定会话 ID 的所有参与者。

```
aws ivs-realtime list-participants \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g
```

输出：

```
{
    "participants": [
        {
            "firstJoinTime": "2023-04-26T20:30:34+00:00",
            "participantId": "abCDEf12GHIj",
            "published": true,
            "recordingState": "STOPPED",
            "state": "DISCONNECTED",
            "userId": "",
            "replicationState": "ACTIVE",
            "replicationType": "SOURCE",
            "sourceStageArn": "",
            "sourceSessionId": ""
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 3：当从另一个阶段复制了参与者时，获取阶段参与者的列表

以下 `list-participants` 示例当从另一个阶段复制了某个参与者时，列出指定阶段 ARN（Amazon 资源名称）的指定会话 ID 的所有参与者。

```
aws ivs-realtime list-participants \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --session-id st-a1b2c3d4e5f6g
```

输出：

```
{
    "participants": [
        {
            "firstJoinTime": "2023-04-26T20:30:34+00:00",
            "participantId": "abCDEf12GHIj",
            "published": true,
            "recordingState": "STOPPED",
            "state": "DISCONNECTED",
            "userId": "",
            "replicationState": "ACTIVE",
            "replicationType": "REPLICA",
            "sourceStageArn": "arn:aws:ivs:us-west-2:123456789012:stage/
abcdABCDefgh",
            "sourceSessionId": "st-a1b2c3d4e5f6g"
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListParticipants。

`list-public-keys`

以下代码示例演示了如何使用 `list-public-keys`。

Amazon CLI

列出可用于签署阶段参与者令牌的现有公钥

以下 `list-public-keys` 示例列出在处理 API 请求的 Amazon 区域中，可用于签署阶段参与者令牌的所有公钥。

```
aws ivs-realtime list-public-keys
```

输出：

```
{
```

```
    "publicKeys": [
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:public-key/abcdABC1efg2",
            "name": "",
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:us-west-2:123456789012:public-key/3bcdABCDefg4",
            "name": "",
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon IVS 实时直播功能用户指南》中的分发参与者令牌。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPublicKeys。

list-stage-sessions

以下代码示例演示了如何使用 list-stage-sessions。

Amazon CLI

获取阶段会话列表

以下 list-stage-sessions 示例列出指定阶段 ARN（Amazon 资源名称）的所有会话。

```
aws ivs-realtime list-stage-sessions \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

输出：

```
{
    "stageSessions": [
        {
            "endTime": "2023-04-26T20:36:29+00:00",
            "sessionId": "st-a1b2c3d4e5f6g",
            "startTime": "2023-04-26T20:30:29.602000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStageSessions。

list-stages

以下代码示例演示了如何使用 list-stages。

Amazon CLI

获取有关所有阶段的摘要信息

以下 list-stages 示例列出在处理 API 请求的 Amazon 区域内，您的 Amazon 账户的所有阶段。

```
aws ivs-realtime list-stages
```

输出：

```
{
    "stages": [
        {
            "activeSessionId": "st-a1b2c3d4e5f6g",
            "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
            "name": "stage1",
            "tags": {}
        },
        {
            "activeSessionId": "st-a123bcd456efg",
            "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcd1234ABCD",
            "name": "stage2",
            "tags": {}
        },
        {
            "activeSessionId": "st-abcDEF1234ghi",
            "arn": "arn:aws:ivs:us-west-2:123456789012:stage/ABCD1234efgh",
            "name": "stage3",
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStages。

`list-storage-configurations`

以下代码示例演示了如何使用 `list-storage-configurations`。

Amazon CLI

列出合成存储配置

以下 `list-storage-configurations` 命令列出在处理 API 请求的 Amazon 区域中，您的 Amazon 账户的所有合成存储配置。

```
aws ivs-realtime list-storage-configurations
```

输出：

```
{
    "storageConfigurations": [
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
abcdABCDefgh",
            "name": "test-sc-1",
            "s3": {
                "bucketName": "amzn-s3-demo-bucket-1"
            },
            "tags": {}
        },
        {
            "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
ABCefgEFGabc",
            "name": "test-sc-2",
            "s3": {
                "bucketName": "amzn-s3-demo-bucket-2"
            },
            "tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStorageConfigurations。

start-composition

以下代码示例演示了如何使用 start-composition。

Amazon CLI

示例 1：启动采用默认布局设置的合成

以下 start-composition 示例启动指定阶段的合成，以流式传输到指定位置。

```
aws ivs-realtime start-composition \
    --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd \
    --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg", \
        "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}}, \
        {"s3":{"encoderConfigurationArns":["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
        "recordingConfiguration": {"hlsConfiguration":
 {"targetSegmentDurationSeconds": 5}}, \
        "storageConfigurationArn":"arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE"}}]'
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
```

```
                },
                "id": "AabBCcdDEefF",
                "state": "STARTING"
            },
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 5
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "state": "STARTING"
            }
        ],
        "layout": {
            "grid": {
                "featuredParticipantAttribute": ""
                "gridGap": 2,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "",
                "videoAspectRatio": "VIDEO",
                "videoFillMode": ""
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
```

```
        "state": "STARTING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

示例 2：启动采用 PiP 布局的合成

以下 start-composition 示例启动指定阶段的合成，以流式传输到使用 PiP 布局的指定位置。

```
aws ivs-realtime start-composition \
    --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd \
    --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg", \
        "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}}, \
        {"s3":{"encoderConfigurationArns":["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
        "storageConfigurationArn":"arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE"}}]' \
    --layout pip='{featuredParticipantAttribute="abcdefg"}'
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "state": "STARTING"
            },
```

```
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "state": "STARTING"
            }
        ],
        "layout": {
            "pip": {
                "featuredParticipantAttribute": "abcdefg",
                "gridGap": 0,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "",
                "pipBehavior": "STATIC",
                "pipOffset": 0,
                "pipParticipantAttribute": "",
                "pipPosition": "BOTTOM_RIGHT",
                "videoFillMode": "COVER"
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
        "state": "STARTING",
```

```
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

示例 3：在启用缩略图录制的情况下开始合成

以下 start-composition 示例开始指定阶段的合成，以便在启用缩略图录制的情况下流式传输到指定位置。

```
aws ivs-realtime start-composition \
    --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd \
    --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg", \
        "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}}, \
        {"s3": {"encoderConfigurationArns": ["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
        "storageConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE", \
        "thumbnailConfigurations": [{"storage":
 ["SEQUENTIAL"],"targetIntervalSeconds": 60}]}}]'
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "state": "STARTING"
```

```
            },
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE",
                        "thumbnailConfigurations": [
                           {
                              "targetIntervalSeconds": 60,
                              "storage": [
                                  "SEQUENTIAL"
                              ]
                           }
                        ]
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "state": "STARTING"
            }
        ],
        "layout": {
            "grid": {
                "featuredParticipantAttribute": ""
                "gridGap": 2,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "",
                "videoAspectRatio": "VIDEO",
```

```
                "videoFillMode": ""
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
        "state": "STARTING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的合成录制（实时直播功能）。

示例 4：结合使用网格布局与自定义参与者排序开始合成

以下 `start-composition` 示例结合使用网格布局与自定义参与者排序，来启动指定阶段的合成，以流式传输到指定位置。

```
aws ivs-realtime start-composition \
    --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd \
    --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg", \
        "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}}, \
        {"s3": {"encoderConfigurationArns": ["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
        "storageConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE", \
        "thumbnailConfigurations": [{"storage":
 ["SEQUENTIAL"],"targetIntervalSeconds": 60}]}}]' \
    --layout grid='{participantOrderAttribute="abcdefg"}'
```

输出：

```
{
    "composition": {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
        "destinations": [
            {
                "configuration": {
                    "channel": {
                        "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
```

```
                        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                    },
                    "name": ""
                },
                "id": "AabBCcdDEefF",
                "state": "STARTING"
            },
            {
                "configuration": {
                    "name": "",
                    "s3": {
                        "encoderConfigurationArns": [
                            "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
                        ],
                        "recordingConfiguration": {
                            "format": "HLS",
                            "hlsConfiguration": {
                                "targetSegmentDurationSeconds": 2
                            }
                        },
                        "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE",
                        "thumbnailConfigurations": [
                           {
                              "targetIntervalSeconds": 60,
                              "storage": [
                                  "SEQUENTIAL"
                              ]
                           }
                        ]
                    }
                },
                "detail": {
                    "s3": {
                        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
                    }
                },
                "id": "GHFabcgefABC",
                "state": "STARTING"
            }
        ],
```

```
        "layout": {
            "grid": {
                "featuredParticipantAttribute": ""
                "gridGap": 2,
                "omitStoppedVideo": false,
                "participantOrderAttribute": "abcdefg",
                "videoAspectRatio": "VIDEO",
                "videoFillMode": ""
            }
        },
        "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
        "startTime": "2023-10-16T23:24:00+00:00",
        "state": "STARTING",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 合成录制 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartComposition。

start-participant-replication

以下代码示例演示了如何使用 start-participant-replication。

Amazon CLI

开始将参与者从一个阶段复制到另一个阶段

以下 start-participant-replication 示例将参与者从源阶段复制到目标阶段，每个阶段都由其 ARN（Amazon 资源名称）指定。

```
aws ivs-realtime start-participant-replication \
    --source-stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --destination-stage-arn arn:aws:ivs:us-west-2:234567890123:stage/bcdABCDefghi \
    --participant-id abCDEf12GHIj
```

输出：

```
{
```

```
    "accessControlAllowOrigin": "*",
    "accessControlExposeHeaders": "Access-Control-Allow-Origin,Access-Control-
Expose-Headers,Cache-Control,Content-Length, \
    Content-Security-Policy,Content-Type,date,Strict-Transport-Security,x-amz-apigw-
id,x-amzn-errormessage,x-amzn-errortype, \
    x-amzn-requestid,x-amzn-trace-id,X-Content-Type-Options,X-Frame-Options",
    "cacheControl": "no-store, no-cache",
    "contentSecurityPolicy": "default-src 'self'; upgrade-insecure-requests;",
    "strictTransportSecurity": "max-age:47304000; includeSubDomains",
    "xContentTypeOptions": "nosniff",
    "xFrameOptions": "DENY"
}
```

有关更多信息，请参阅《Amazon IVS Real-Time Streaming User Guide》中的 IVS 参与者复制。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartParticipantReplication。

stop-composition

以下代码示例演示了如何使用 stop-composition。

Amazon CLI

停止合成

以下 stop-composition 命令停止由给定 ARN（Amazon 资源名称）指定的合成。

```
aws ivs-realtime stop-composition \
    --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的在 Amazon IVS Stream 中启用多个主播。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopComposition。

stop-participant-replication

以下代码示例演示了如何使用 stop-participant-replication。

Amazon CLI

停止将参与者从一个阶段复制到另一个阶段

以下 stop-participant-replication 示例停止将参与者从源阶段复制到目标阶段，每个阶段都由其 ARN（Amazon 资源名称）指定。

```
aws ivs-realtime stop-participant-replication \
    --source-stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --destination-stage-arn arn:aws:ivs:us-west-2:234567890123:stage/bcdABCDefghi \
    --participant-id abCDEf12GHIj
```

输出：

```
{
    "accessControlAllowOrigin": "*",
    "accessControlExposeHeaders": "Access-Control-Allow-Origin,Access-Control-
Expose-Headers,Cache-Control,Content-Length, \
    Content-Security-Policy,Content-Type,date,Strict-Transport-Security,x-amz-apigw-
id,x-amzn-errormessage,x-amzn-errortype, \
    x-amzn-requestid,x-amzn-trace-id,X-Content-Type-Options,X-Frame-Options",
    "cacheControl": "no-store, no-cache",
    "contentSecurityPolicy": "default-src 'self'; upgrade-insecure-requests;",
    "strictTransportSecurity": "max-age:47304000; includeSubDomains",
    "xContentTypeOptions": "nosniff",
    "xFrameOptions": "DENY"
}
```

有关更多信息，请参阅《Amazon IVS Real-Time Streaming User Guide》中的 IVS 参与者复制。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopParticipantReplication。

update-ingest-configuration

以下代码示例演示了如何使用 update-ingest-configuration。

Amazon CLI

更新摄取配置

以下 update-inegst-configuration 示例更新摄取配置以将其附加到阶段。

```
aws ivs-realtime update-ingest-configuration \
    --arn arn:aws:ivs:us-west-2:123456789012:ingest-configuration/AbCdEfGh1234 \
    --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

输出：

```
{
    "ingestConfiguration": {
        "name": "ingest1",
        "arn": "arn:aws:ivs:us-west-2:123456789012:ingest-configuration/
AbCdEfGh1234",
        "ingestProtocol": "RTMPS",
        "streamKey": "rt_123456789012_us-
west-2_AbCdEfGh1234_abcd1234efgh5678ijkl9012MNOP34",
        "stageArn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "participantId": "xyZ654abC321",
        "state": "INACTIVE",
        "userId": "",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Interactive Video Service 用户指南》中的 IVS 流摄取 | 实时直播功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateIngestConfiguration。

update-stage

以下代码示例演示了如何使用 update-stage。

Amazon CLI

示例 1：更新阶段的配置

以下 update-stage 示例更新指定阶段 ARN 的阶段以更新阶段名称，并配置启用缩略图录制的单个参与者录制。

```
aws ivs-realtime update-stage \
    --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
```

```
    --auto-participant-recording-configuration '{"mediaTypes":
 ["AUDIO_VIDEO"],"storageConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:storage-configuration/abcdABCDefgh",
 "recordingReconnectWindowSeconds": 100, \
        "thumbnailConfiguration": {"recordingMode": "INTERVAL","storage":
 ["SEQUENTIAL"],"targetIntervalSeconds": 60}} \
        "hlsConfiguration": {"targetSegmentDurationSeconds": 5}}' \
    --name stage1a
```

输出：

```
{
    "stage": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "storageConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:storage-
configuration/abcdABCDefgh",
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "INTERVAL"
            },
            "recordingReconnectWindowSeconds": 100,
            "hlsConfiguration": {
                "targetSegmentDurationSeconds": 5
            },
            "recordParticipantReplicas": true
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://1a2b3c4d5e6f.global-bm.whip.live-video.net"
        },
        "name": "stage1a",
        "tags": {}
```

```
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

示例 2：更新阶段的配置，包括禁用参与者副本录制

以下 update-stage 示例更新指定阶段 ARN 的阶段以更新阶段名称，并在启用缩略图录制和禁用参与者副本录制的情况下配置各个参与者录制。

```
aws ivs-realtime update-stage \
    --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
    --auto-participant-recording-configuration '{"mediaTypes":
 ["AUDIO_VIDEO"],"storageConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:storage-configuration/abcdABCDefgh",
 "recordingReconnectWindowSeconds": 100, \
        "thumbnailConfiguration": {"recordingMode": "INTERVAL","storage":
 ["SEQUENTIAL"],"targetIntervalSeconds": 60}, "recordParticipantReplicas":false} \
        "hlsConfiguration": {"targetSegmentDurationSeconds": 5}}' \
    --name stage1a
```

输出：

```
{
    "stage": {
        "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
        "autoParticipantRecordingConfiguration": {
            "mediaTypes": [
                "AUDIO_VIDEO"
            ],
            "storageConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:storage-
configuration/abcdABCDefgh",
            "thumbnailConfiguration": {
                "targetIntervalSeconds": 60,
                "storage": [
                    "SEQUENTIAL"
                ],
                "recordingMode": "INTERVAL"
            },
            "recordingReconnectWindowSeconds": 100,
            "hlsConfiguration": {
```

```
                "targetSegmentDurationSeconds": 5
            },
            "recordParticipantReplicas": false
        },
        "endpoints": {
            "events": "wss://global.events.live-video.net",
            "rtmp": "rtmp://9x0y8z7s6t5u.global-contribute-staging.live-video.net/
app/",
            "rtmps": "rtmps://9x0y8z7s6t5u.global-contribute-staging.live-
video.net:443/app/",
            "whip": "https://1a2b3c4d5e6f.global-bm.whip.live-video.net"
        },
        "name": "stage1a",
        "tags": {}
    }
}
```

有关更多信息，请参阅《Amazon IVS Low-Latency Streaming User Guide》中的在 Amazon IVS 流中启用多台主机。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStage。

使用 Amazon CLI 的 Amazon Kendra 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Kendra 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-data-source`

以下代码示例演示了如何使用 `create-data-source`。

Amazon CLI

创建 Amazon Kendra 数据来源连接器

以下 `create-data-source` 示例创建和配置 Amazon Kendra 数据来源连接器。您可以使用 `describe-data-source` 查看数据来源连接器的状态，如果状态显示创建数据来源连接器“失败”，则读取相关错误消息。

```
aws kendra create-data-source \
    --name "example data source 1" \
    --description "Example data source 1 for example index 1 contains the first set
 of example documents" \
    --tags '{"Key": "test resources", "Value": "kendra"}, {"Key": "test resources",
 "Value": "aws"}' \
    --role-arn "arn:aws:iam::my-account-id:role/
KendraRoleForS3TemplateConfigDataSource" \
    --index-id exampleindex1 \
    --language-code "es" \
    --schedule "0 0 18 ? * TUE,MON,WED,THU,FRI,SAT *" \
    --configuration '{"TemplateConfiguration": {"Template": file://
s3schemaconfig.json}}' \
    --type "TEMPLATE" \
    --custom-document-enrichment-configuration '{"PostExtractionHookConfiguration":
 {"LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
 "S3Bucket": "s3://amzn-s3-demo-bucket/scanned-image-text-example-docs"}, "RoleArn":
 "arn:aws:iam:my-account-id:role/KendraRoleForCDE"}' \
    --vpc-configuration '{"SecurityGroupIds": ["sg-1234567890abcdef0"], "SubnetIds":
 ["subnet-1c234","subnet-2b134"]}'
```

输出：

```
{
    "Id": "exampledatasource1"
}
```

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataSource。

create-index

以下代码示例演示了如何使用 create-index。

Amazon CLI

创建 Amazon Kendra 索引

以下 create-index 示例创建和配置 Amazon Kendra 索引。您可以使用 describe-index 查看索引的状态，如果状态显示创建索引“失败”，则读取相关错误消息。

```
aws kendra create-index \
    --name "example index 1" \
    --description "Example index 1 contains the first set of example documents" \
    --tags '{"Key": "test resources", "Value": "kendra"}, {"Key": "test resources",
 "Value": "aws"}' \
    --role-arn "arn:aws:iam::my-account-id:role/KendraRoleForExampleIndex" \
    --edition "DEVELOPER_EDITION" \
    --server-side-encryption-configuration '{"KmsKeyId": "my-kms-key-id"}' \
    --user-context-policy "USER_TOKEN" \
    --user-token-configurations '{"JsonTokenTypeConfiguration":
 {"GroupAttributeField": "groupNameField", "UserNameAttributeField":
 "userNameField"}}'
```

输出：

```
{
   "Id": index1
}
```

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIndex。

describe-data-source

以下代码示例演示了如何使用 describe-data-source。

Amazon CLI

获取有关 Amazon Kendra 数据来源连接器的信息

以下 describe-data-source 命令获取有关 Amazon Kendra 数据来源连接器的信息。您可以查看数据来源连接器的配置，如果状态显示创建数据来源连接器“失败”，则读取相关错误消息。

```
aws kendra describe-data-source \
    --id exampledatasource1 \
    --index-id exampleindex1
```

输出：

```
{
    "Configuration": {
        "TemplateConfiguration": {
            "Template": {
                "connectionConfiguration": {
                    "repositoryEndpointMetadata": {
                        "BucketName": "amzn-s3-demo-bucket"
                    }
                },
                "repositoryConfigurations": {
                    "document":{
                        "fieldMappings": [
                            {
                                "indexFieldName":"_document_title",
                                "indexFieldType":"STRING",
                                "dataSourceFieldName": "title"
                            },
                            {
                                "indexFieldName":"_last_updated_at",
                                "indexFieldType":"DATE",
                                "dataSourceFieldName": "modified_date"
                            }
                        ]
                    }
                },
                "additionalProperties": {
                    "inclusionPatterns": [
                        "*.txt",
                        "*.doc",
                        "*.docx"
                    ],
                    "exclusionPatterns": [
                        "*.json"
                    ],
```

```
                    "inclusionPrefixes": [
                        "PublicExampleDocsFolder"
                    ],
                    "exclusionPrefixes": [
                        "PrivateDocsFolder/private"
                    ],
                    "aclConfigurationFilePath": "ExampleDocsFolder/AclConfig.json",
                    "metadataFilesPrefix": "metadata"
                },
                "syncMode": "FULL_CRAWL",
                "type" : "S3",
                "version": "1.0.0"
            }
        }
    },
    "CreatedAt": 2024-02-25T13:30:10+00:00,
    "CustomDocumentEnrichmentConfiguration": {
        "PostExtractionHookConfiguration": {
            "LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
            "S3Bucket": "s3://amzn-s3-demo-bucket/scanned-image-text-example-docs/
function"
        },
        "RoleArn": "arn:aws:iam:my-account-id:role/KendraRoleForCDE"
    }
    "Description": "Example data source 1 for example index 1 contains the first set
 of example documents",
    "Id": exampledatasource1,
    "IndexId": exampleindex1,
    "LanguageCode": "en",
    "Name": "example data source 1",
    "RoleArn": "arn:aws:iam::my-account-id:role/
KendraRoleForS3TemplateConfigDataSource",
    "Schedule": "0 0 18 ? * TUE,MON,WED,THU,FRI,SAT *",
    "Status": "ACTIVE",
    "Type": "TEMPLATE",
    "UpdatedAt": 1709163615,
    "VpcConfiguration": {
        "SecurityGroupIds": ["sg-1234567890abcdef0"],
        "SubnetIds": ["subnet-1c234","subnet-2b134"]
    }
}
```

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDataSource。

describe-index

以下代码示例演示了如何使用 describe-index。

Amazon CLI

获取有关 Amazon Kendra 索引的信息

以下 describe-index 命令获取有关 Amazon Kendra 索引的信息。您可以查看索引的配置，如果状态显示创建索引“失败”，则读取相关错误消息。

```
aws kendra describe-index \
    --id exampleindex1
```

输出：

```
{
    "CapacityUnits": {
        "QueryCapacityUnits": 0,
        "StorageCapacityUnits": 0
    },
    "CreatedAt": 2024-02-25T12:30:10+00:00,
    "Description": "Example index 1 contains the first set of example documents",
    "DocumentMetadataConfigurations": [
        {
            "Name": "_document_title",
            "Relevance": {
                "Importance": 8
            },
            "Search": {
                "Displayable": true,
                "Facetable": false,
                "Searchable": true,
                "Sortable": false
            },
            "Type": "STRING_VALUE"
        },
        {
```

```
            "Name": "_document_body",
            "Relevance": {
                "Importance": 5
            },
            "Search": {
                "Displayable": true,
                "Facetable": false,
                "Searchable": true,
                "Sortable": false
            },
            "Type": "STRING_VALUE"
        },
        {
            "Name": "_last_updated_at",
            "Relevance": {
                "Importance": 6,
                "Duration": "2628000s",
                "Freshness": true
            },
            "Search": {
                "Displayable": true,
                "Facetable": false,
                "Searchable": true,
                "Sortable": true
            },
            "Type": "DATE_VALUE"
        },
        {
            "Name": "department_custom_field",
            "Relevance": {
                "Importance": 7,
                "ValueImportanceMap": {
                    "Human Resources" : 4,
                    "Marketing and Sales" : 2,
                    "Research and innvoation" : 3,
                    "Admin" : 1
                }
            },
            "Search": {
                "Displayable": true,
                "Facetable": true,
                "Searchable": true,
                "Sortable": true
            },
```

```
            "Type": "STRING_VALUE"
        }
    ],
    "Edition": "DEVELOPER_EDITION",
    "Id": "index1",
    "IndexStatistics": {
        "FaqStatistics": {
            "IndexedQuestionAnswersCount": 10
        },
        "TextDocumentStatistics": {
            "IndexedTextBytes": 1073741824,
            "IndexedTextDocumentsCount": 1200
        }
    },
    "Name": "example index 1",
    "RoleArn": "arn:aws:iam::my-account-id:role/KendraRoleForExampleIndex",
    "ServerSideEncryptionConfiguration": {
        "KmsKeyId": "my-kms-key-id"
    },
    "Status": "ACTIVE",
    "UpdatedAt": 1709163615,
    "UserContextPolicy": "USER_TOKEN",
    "UserTokenConfigurations": [
        {
            "JsonTokenTypeConfiguration": {
                "GroupAttributeField": "groupNameField",
                "UserNameAttributeField": "userNameField"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeIndex。

update-data-source

以下代码示例演示了如何使用 update-data-source。

Amazon CLI

更新 Amazon Kendra 数据来源连接器

以下 update-data-source 命令更新 Amazon Kendra 数据来源连接器的配置。如果操作成功，则服务发回 HTTP 状态码 200 而无任何输出，或者 Amazon CLI 返回代码 0。您可以使用 describe-data-source 查看数据来源连接器的配置和状态。

```
aws kendra update-data-source \
    --id exampledatasource1 \
    --index-id exampleindex1 \
    --name "new name for example data source 1" \
    --description "new description for example data source 1" \
    --role-arn arn:aws:iam::my-account-id:role/KendraNewRoleForExampleDataSource \
    --configuration '{"TemplateConfiguration": {"Template": file://
s3schemanewconfig.json}}' \
    --custom-document-enrichment-configuration '{"PostExtractionHookConfiguration":
 {"LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
 "S3Bucket": "s3://amzn-s3-demo-bucket/scanned-image-text-example-docs"}, "RoleArn":
 "arn:aws:iam:my-account-id:role/KendraNewRoleForCDE"}' \
    --language-code "es" \
    --schedule "0 0 18 ? * MON,WED,FRI *" \
    --vpc-configuration '{"SecurityGroupIds": ["sg-1234567890abcdef0"], "SubnetIds":
 ["subnet-1c234","subnet-2b134"]}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDataSource。

update-index

以下代码示例演示了如何使用 update-index。

Amazon CLI

更新 Amazon Kendra 索引

以下 update-index 命令更新 Amazon Kendra 索引的配置。如果操作成功，则服务发回 HTTP 状态码 200 而无任何输出，或者 Amazon CLI 返回代码 0。您可以使用 describe-index 查看索引的配置和状态。

```
aws kendra update-index \
    --id enterpriseindex1 \
    --name "new name for Enterprise Edition index 1" \
    --description "new description for Enterprise Edition index 1" \
    --role-arn arn:aws:iam::my-account-id:role/KendraNewRoleForEnterpriseIndex \
    --capacity-units '{"QueryCapacityUnits": 2, "StorageCapacityUnits": 1}' \
    --document-metadata-configuration-updates '{"Name": "_document_title",
 "Relevance": {"Importance": 6}}, {"Name": "_last_updated_at", "Relevance":
 {"Importance": 8}}' \
    --user-context-policy "USER_TOKEN" \
    --user-token-configurations '{"JsonTokenTypeConfiguration":
 {"GroupAttributeField": "groupNameField", "UserNameAttributeField":
 "userNameField"}}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kendra 开发人员指南》中的 Amazon Kendra 索引和数据来源连接器入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIndex。

使用 Amazon CLI 的 Kinesis 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Kinesis 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-stream`

以下代码示例演示了如何使用 `add-tags-to-stream`。

Amazon CLI

为数据流添加标签

以下 add-tags-to-stream 示例将键为 samplekey 且值为 example 的标签分配给指定流。

```
aws kinesis add-tags-to-stream \
    --stream-name samplestream \
    --tags samplekey=example
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的标记流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToStream。

create-stream

以下代码示例演示了如何使用 create-stream。

Amazon CLI

创建数据流

以下 create-stream 示例创建一个名为 samplestream 的数据流，其中包含 3 个分片。

```
aws kinesis create-stream \
    --stream-name samplestream \
    --shard-count 3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的创建流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStream。

decrease-stream-retention-period

以下代码示例演示了如何使用 decrease-stream-retention-period。

Amazon CLI

缩短数据流保留期

以下 decrease-stream-retention-period 示例将名为 samplestream 的数据流的保留期（数据记录在添加到流中后可供访问的时间长度）缩短到 48 小时。

```
aws kinesis decrease-stream-retention-period \
    --stream-name samplestream \
    --retention-period-hours 48
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的更改数据留存期。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecreaseStreamRetentionPeriod。

delete-stream

以下代码示例演示了如何使用 delete-stream。

Amazon CLI

删除数据流

以下 delete-stream 示例删除指定的数据流。

```
aws kinesis delete-stream \
    --stream-name samplestream
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的删除流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStream。

deregister-stream-consumer

以下代码示例演示了如何使用 deregister-stream-consumer。

Amazon CLI

注销数据流消费端

以下 deregister-stream-consumer 示例注销指定数据流的指定消费端。

```
aws kinesis deregister-stream-consumer \
```

```
    --stream-arn arn:aws:kinesis:us-west-2:123456789012:stream/samplestream \
    --consumer-name KinesisConsumerApplication
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的使用 Kinesis Data Streams API 开发具有增强型扇出功能的消费端。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterStreamConsumer。

describe-limits

以下代码示例演示了如何使用 describe-limits。

Amazon CLI

描述分片限制

以下 describe-limits 示例显示当前 Amazon 账户的分片限制和用量。

```
aws kinesis describe-limits
```

输出：

```
{
    "ShardLimit": 500,
    "OpenShardCount": 29
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的重新对流进行分片。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLimits。

describe-stream-consumer

以下代码示例演示了如何使用 describe-stream-consumer。

Amazon CLI

描述数据流消费端

以下 describe-stream-consumer 示例返回在指定数据流中注册的指定消费端的描述。

```
aws kinesis describe-stream-consumer \
    --stream-arn arn:aws:kinesis:us-west-2:012345678912:stream/samplestream \
    --consumer-name KinesisConsumerApplication
```

输出：

```
{
    "ConsumerDescription": {
        "ConsumerName": "KinesisConsumerApplication",
        "ConsumerARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream/
consumer/KinesisConsumerApplication:1572383852",
        "ConsumerStatus": "ACTIVE",
        "ConsumerCreationTimestamp": 1572383852.0,
        "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream"
    }
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的从 Amazon Kinesis Data Streams 读取数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStreamConsumer。

describe-stream-summary

以下代码示例演示了如何使用 describe-stream-summary。

Amazon CLI

描述数据流摘要

以下 describe-stream-summary 示例提供指定数据流的摘要描述（无分片列表）。

```
aws kinesis describe-stream-summary \
    --stream-name samplestream
```

输出：

```
{
    "StreamDescriptionSummary": {
        "StreamName": "samplestream",
        "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream",
```

```
        "StreamStatus": "ACTIVE",
        "RetentionPeriodHours": 48,
        "StreamCreationTimestamp": 1572297168.0,
        "EnhancedMonitoring": [
            {
                "ShardLevelMetrics": []
            }
        ],
        "EncryptionType": "NONE",
        "OpenShardCount": 3,
        "ConsumerCount": 0
    }
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的创建和管理流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStreamSummary。

describe-stream

以下代码示例演示了如何使用 describe-stream。

Amazon CLI

描述数据流

以下 describe-stream 示例返回指定数据流的详细信息。

```
aws kinesis describe-stream \
    --stream-name samplestream
```

输出：

```
{
    "StreamDescription": {
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "StartingHashKey": "0",
                    "EndingHashKey": "113427455640312821154458202477256070484"
                },
                "SequenceNumberRange": {
```

```
                    "StartingSequenceNumber":
"49600871682957036442365024926191073437251060580128653314"
                }
            },
            {
                "ShardId": "shardId-000000000001",
                "HashKeyRange": {
                    "StartingHashKey": "113427455640312821154458202477256070485",
                    "EndingHashKey": "226854911280625642308916404954512140969"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
"49600871682979337187563555549332609155523708941634633746"
                }
            },
            {
                "ShardId": "shardId-000000000002",
                "HashKeyRange": {
                    "StartingHashKey": "226854911280625642308916404954512140970",
                    "EndingHashKey": "340282366920938463463374607431768211455"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
"49600871683001637932762086172474144873796357303140614178"
                }
            }
        ],
        "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream",
        "StreamName": "samplestream",
        "StreamStatus": "ACTIVE",
        "RetentionPeriodHours": 24,
        "EnhancedMonitoring": [
            {
                "ShardLevelMetrics": []
            }
        ],
        "EncryptionType": "NONE",
        "KeyId": null,
        "StreamCreationTimestamp": 1572297168.0
    }
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的创建和管理流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStream。

disable-enhanced-monitoring

以下代码示例演示了如何使用 disable-enhanced-monitoring。

Amazon CLI

禁用对分片级别指标进行增强型监控

以下 disable-enhanced-monitoring 示例禁用对分片级别指标进行增强型 Kinesis 数据流监控。

```
aws kinesis disable-enhanced-monitoring \
    --stream-name samplestream --shard-level-metrics ALL
```

输出：

```
{
    "StreamName": "samplestream",
    "CurrentShardLevelMetrics": [
        "IncomingBytes",
        "OutgoingRecords",
        "IteratorAgeMilliseconds",
        "IncomingRecords",
        "ReadProvisionedThroughputExceeded",
        "WriteProvisionedThroughputExceeded",
        "OutgoingBytes"
    ],
    "DesiredShardLevelMetrics": []
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的在 Amazon Kinesis Data Streams 中监控流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableEnhancedMonitoring。

enable-enhanced-monitoring

以下代码示例演示了如何使用 enable-enhanced-monitoring。

Amazon CLI

启用对分片级别指标进行增强型监控

以下 enable-enhanced-monitoring 示例启用对分片级别指标进行增强型 Kinesis 数据流监控。

```
aws kinesis enable-enhanced-monitoring \
    --stream-name samplestream \
    --shard-level-metrics ALL
```

输出：

```
{
    "StreamName": "samplestream",
    "CurrentShardLevelMetrics": [],
    "DesiredShardLevelMetrics": [
        "IncomingBytes",
        "OutgoingRecords",
        "IteratorAgeMilliseconds",
        "IncomingRecords",
        "ReadProvisionedThroughputExceeded",
        "WriteProvisionedThroughputExceeded",
        "OutgoingBytes"
    ]
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的在 Amazon Kinesis Data Streams 中监控流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableEnhancedMonitoring。

get-records

以下代码示例演示了如何使用 get-records。

Amazon CLI

获取分片中的记录

以下 get-records 示例使用指定分片迭代器从 Kinesis 数据流的分片中获取数据记录。

```
aws kinesis get-records \
    --shard-iterator AAAAAAAAAAF7/0mWD7IuHj1yGv/
TKuNgx2ukD5xipCY4cy4gU96orWwZwcSXh3K9tAmGYeOZyLZrvzzeOFVf9iN99hUPw/w/
b0YWYeehfNvnf1DYt5XpDJghLKr3DzgznkTmMymDP3R+3wRKeuEw6/kdxY2yKJH0veaiekaVc4N2VwK/
GvaGP2Hh9Fg7N++q0Adg6fIDQPt4p8RpavDbk+A4sL9SWGE1
```

输出：

```
{
    "Records": [],
    "MillisBehindLatest": 80742000
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的通过 Amazon SDK for Java 使用 Kinesis Data Streams API 开发消费端。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRecords。

get-shard-iterator

以下代码示例演示了如何使用 get-shard-iterator。

Amazon CLI

获取分片迭代器

以下 get-shard-iterator 示例使用 AT_SEQUENCE_NUMBER 分片迭代器类型并生成分片迭代器，以便从指定序列号所表示的位置开始精确读取数据记录。

```
aws kinesis get-shard-iterator \
    --stream-name samplestream \
    --shard-id shardId-000000000001 \
    --shard-iterator-type LATEST
```

输出：

```
{
    "ShardIterator": "AAAAAAAAAAFEvJjIYI+3jw/4aqgH9FifJ+n48XWTh/
IFIsbILP6o5eDueD39NXNBfpZ10WL5K6ADXk8w+5H+Qhd9cFA9k268CPXCz/kebq1TGYI7Vy
+lUkA9BuN3xvATxMBGxRY3zYK05gqgvaIRn94O8SqeEqwhigwZxNWxID3Ej7YYYcxQi8Q/fIrCjGAy/
n2r5Z9G864YpWDfN9upNNQAR/iiOWKs"
```

```
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的通过 Amazon SDK for Java 使用 Kinesis Data Streams API 开发消费端。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetShardIterator。

`increase-stream-retention-period`

以下代码示例演示了如何使用 `increase-stream-retention-period`。

Amazon CLI

延长数据流保留期

以下 `increase-stream-retention-period` 示例将指定数据流的保留期（数据记录在添加到流中后可供访问的时间长度）延长到 168 小时。

```
aws kinesis increase-stream-retention-period \
    --stream-name samplestream \
    --retention-period-hours 168
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的更改数据留存期。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IncreaseStreamRetentionPeriod。

`list-shards`

以下代码示例演示了如何使用 `list-shards`。

Amazon CLI

列出数据流中的分片

以下 `list-shards` 示例列出指定流中从 ID 紧接指定 `exclusive-start-shard-id` 的 `shardId-000000000000` 后的分片开始的所有分片。

```
aws kinesis list-shards \
```

```
    --stream-name samplestream \
    --exclusive-start-shard-id shardId-000000000000
```

输出：

```
{
    "Shards": [
        {
            "ShardId": "shardId-000000000001",
            "HashKeyRange": {
                "StartingHashKey": "113427455640312821154458202477256070485",
                "EndingHashKey": "226854911280625642308916404954512140969"
            },
            "SequenceNumberRange": {
                "StartingSequenceNumber":
 "496008716829793371875635555493326091555237089416346337 46"
            }
        },
        {
            "ShardId": "shardId-000000000002",
            "HashKeyRange": {
                "StartingHashKey": "226854911280625642308916404954512140970",
                "EndingHashKey": "340282366920938463463374607431768211455"
            },
            "SequenceNumberRange": {
                "StartingSequenceNumber":
 "496008716830016379327620861724741448737963573031406141 78"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的列出分片。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListShards。

list-streams

以下代码示例演示了如何使用 list-streams。

Amazon CLI

列出数据流

以下 `list-streams` 示例列出了当前账户和区域中的所有活动数据流。

```
aws kinesis list-streams
```

输出：

```
{
    "StreamNames": [
        "samplestream",
        "samplestream1"
    ]
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的列出流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreams。

`list-tags-for-stream`

以下代码示例演示了如何使用 `list-tags-for-stream`。

Amazon CLI

列出数据流的标签

以下 `list-tags-for-stream` 示例列出附加到指定数据流的标签：

```
aws kinesis list-tags-for-stream \
    --stream-name samplestream
```

输出：

```
{
    "Tags": [
        {
            "Key": "samplekey",
            "Value": "example"
        }
    ],
    "HasMoreTags": false
```

```
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的标记流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForStream。

merge-shards

以下代码示例演示了如何使用 merge-shards。

Amazon CLI

合并分片

以下 merge-shards 示例将指定数据流中两个相邻的且 ID 分别为 shardId-000000000000 和 shardId-000000000001 的分片合并成一个分片。

```
aws kinesis merge-shards \
    --stream-name samplestream \
    --shard-to-merge shardId-000000000000 \
    --adjacent-shard-to-merge shardId-000000000001
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的合并两个分片。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MergeShards。

put-record

以下代码示例演示了如何使用 put-record。

Amazon CLI

将记录写入数据流

以下 put-record 示例使用指定的分区键将单个数据记录写入指定的数据流。

```
aws kinesis put-record \
    --stream-name samplestream \
    --data sampledatarecord \
```

```
    --partition-key samplepartitionkey
```

输出：

```
{
    "ShardId": "shardId-000000000009",
    "SequenceNumber": "496009022733575409159899312569015062438784078352975136 18",
    "EncryptionType": "KMS"
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的通过 Amazon SDK for Java 使用 Amazon Kinesis Data Streams API 开发创建器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRecord。

put-records

以下代码示例演示了如何使用 put-records。

Amazon CLI

将多条记录写入一个数据流中

以下 put-records 示例在一次调用中使用指定的分区键写入一条数据记录，并使用不同的分区键写入另一条数据记录。

```
aws kinesis put-records \
    --stream-name samplestream \
    --
records Data=blob1,PartitionKey=partitionkey1 Data=blob2,PartitionKey=partitionkey2
```

输出：

```
{
    "FailedRecordCount": 0,
    "Records": [
        {
            "SequenceNumber":
 "49600883331171471519674795588238531498465399900093808706",
            "ShardId": "shardId-000000000004"
        },
        {
```

```
            "SequenceNumber":
 "49600902273357540915989931256902715169698037101720764562",
            "ShardId": "shardId-000000000009"
        }
    ],
    "EncryptionType": "KMS"
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的通过 Amazon SDK for Java 使用 Amazon Kinesis Data Streams API 开发创建器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutRecords。

register-stream-consumer

以下代码示例演示了如何使用 register-stream-consumer。

Amazon CLI

注册数据流消费端

以下 register-stream-consumer 示例将名为 KinesisConsumerApplication 的消费端注册到指定数据流。

```
aws kinesis register-stream-consumer \
    --stream-arn arn:aws:kinesis:us-west-2:012345678912:stream/samplestream \
    --consumer-name KinesisConsumerApplication
```

输出：

```
{
    "Consumer": {
        "ConsumerName": "KinesisConsumerApplication",
        "ConsumerARN": "arn:aws:kinesis:us-west-2: 123456789012:stream/samplestream/
consumer/KinesisConsumerApplication:1572383852",
        "ConsumerStatus": "CREATING",
        "ConsumerCreationTimestamp": 1572383852.0
    }
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的使用 Kinesis Data Streams API 开发具有增强型扇出功能的消费端。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterStreamConsumer。

`remove-tags-from-stream`

以下代码示例演示了如何使用 `remove-tags-from-stream`。

Amazon CLI

从数据流移除标签

以下 `remove-tags-from-stream` 示例从指定数据流中移除具有指定键的标签。

```
aws kinesis remove-tags-from-stream \
    --stream-name samplestream \
    --tag-keys samplekey
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的标记流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromStream。

`split-shard`

以下代码示例演示了如何使用 `split-shard`。

Amazon CLI

拆分分片

以下 `split-shard` 示例使用新的起始哈希键 10 将指定分片拆分为两个新分片。

```
aws kinesis split-shard \
    --stream-name samplestream \
    --shard-to-split shardId-000000000000 \
    --new-starting-hash-key 10
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的拆分分片。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SplitShard。

start-stream-encryption

以下代码示例演示了如何使用 start-stream-encryption。

Amazon CLI

启用数据流加密

以下 start-stream-encryption 示例使用指定 Amazon KMS 密钥为指定流的服务器端加密。

```
aws kinesis start-stream-encryption \
    --encryption-type KMS \
    --key-id arn:aws:kms:us-west-2:012345678912:key/a3c4a7cd-728b-45dd-
b334-4d3eb496e452 \
    --stream-name samplestream
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的 Amazon Kinesis Data Streams 中的数据保护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartStreamEncryption。

stop-stream-encryption

以下代码示例演示了如何使用 stop-stream-encryption。

Amazon CLI

禁用数据流加密

以下 stop-stream-encryption 示例禁止使用指定 Amazon KMS 密钥为指定流的服务器端加密。

```
aws kinesis start-stream-encryption \
    --encryption-type KMS \
    --key-id arn:aws:kms:us-west-2:012345678912:key/a3c4a7cd-728b-45dd-
b334-4d3eb496e452 \
    --stream-name samplestream
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的 Amazon Kinesis Data Streams 中的数据保护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopStreamEncryption。

update-shard-count

以下代码示例演示了如何使用 update-shard-count。

Amazon CLI

更新数据流中的分片计数

以下 update-shard-count 示例将指定数据流的分片计数更新为 6。此示例使用统一缩放，创建同等大小的分片。

```
aws kinesis update-shard-count \
    --stream-name samplestream \
    --scaling-type UNIFORM_SCALING \
    --target-shard-count 6
```

输出：

```
{
    "StreamName": "samplestream",
    "CurrentShardCount": 3,
    "TargetShardCount": 6
}
```

有关更多信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的重新对流进行分片。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateShardCount。

使用 Amazon CLI 的 Amazon KMS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon KMS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`cancel-key-deletion`

以下代码示例演示了如何使用 `cancel-key-deletion`。

Amazon CLI

取消对客户托管的 KMS 密钥的预定删除

以下 `cancel-key-deletion` 示例取消对客户托管的 KMS 密钥的预定删除。

```
aws kms cancel-key-deletion \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

`cancel-key-deletion` 命令成功后，预定的删除即被取消。但是，KMS 密钥的密钥状态为 `Disabled`，因此您不能在加密操作中使用该 KMS 密钥。要恢复其功能，请使用 `enable-key` 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的预定和取消密钥删除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelKeyDeletion。

`connect-custom-key-store`

以下代码示例演示了如何使用 `connect-custom-key-store`。

Amazon CLI

连接自定义密钥存储

以下 connect-custom-key-store 示例重新连接指定的自定义密钥存储。您可以使用这样的命令首次连接自定义密钥存储，或重新连接已断连的密钥存储。

您可以使用此命令连接 Amazon CloudHSM 密钥存储或外部密钥存储。

```
aws kms connect-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0
```

此命令不返回任何输出。要验证命令是否有效，请使用 describe-custom-key-stores 命令。

有关连接 Amazon CloudHSM 密钥存储的信息，请参阅《Amazon 密钥管理服务开发人员指南》中的连接和断开连接 Amazon CloudHSM 密钥存储。

有关连接外部密钥存储的信息，请参阅《Amazon 密钥管理服务开发人员指南》中的连接和断开连接外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ConnectCustomKeyStore。

create-alias

以下代码示例演示了如何使用 create-alias。

Amazon CLI

为 KMS 密钥创建别名

以下 create-alias 命令为由密钥 ID 1234abcd-12ab-34cd-56ef-1234567890ab 标识的 KMS 密钥创建名为 example-alias 的别名。

别名名称必须以 alias/ 开头。请勿使用以 alias/aws 开头的别名名称；这些名称被保留以供 Amazon 使用。

```
aws kms create-alias \
    --alias-name alias/example-alias \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不返回任何输出。要查看新别名，请使用 list-aliases 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的使用别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAlias。

create-custom-key-store

以下代码示例演示了如何使用 create-custom-key-store。

Amazon CLI

示例 1：创建 Amazon CloudHSM 密钥存储

以下 create-custom-key-store 示例使用所需参数创建由 Amazon CloudHSM 集群支持的 Amazon CloudHSM 密钥存储。您也可以添加 custom-key-store-type``parameter with the default value: ``AWS_CLOUDHSM。

要在 Amazon CLI 中为 trust-anchor-certificate 命令指定文件输入，需要 file:// 前缀。

```
aws kms create-custom-key-store \
    --custom-key-store-name ExampleCloudHSMKeyStore \
    --cloud-hsm-cluster-id cluster-1a23b4cdefg \
    --key-store-password kmsPswd \
    --trust-anchor-certificate file://customerCA.crt
```

输出：

```
{
    "CustomKeyStoreId": cks-1234567890abcdef0
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的创建 Amazon CloudHSM 密钥存储。

示例 2：创建使用公有端点连接的外部密钥存储

以下 create-custom-key-store 示例创建一个外部密钥存储（XKS），它通过互联网与 Amazon KMS 通信。

在此示例中，XksProxyUriPath 使用可选前缀 example-prefix。

注意：如果使用 Amazon CLI 版本 1.0，请先运行以下命令，再指定具有 HTTP 或 HTTPS 值的参数，例如参数 XksProxyUriEndpoint。

```
aws configure set cli_follow_urlparam false
```

否则，Amazon CLI 版本 1.0 会将参数值替换为在该 URI 地址找到的内容。

```
aws kms create-custom-key-store \
    --custom-key-store-name ExamplePublicEndpointXKS \
    --custom-key-store-type EXTERNAL_KEY_STORE \
    --xks-proxy-connectivity PUBLIC_ENDPOINT \
    --xks-proxy-uri-endpoint "https://myproxy.xks.example.com" \
    --xks-proxy-uri-path "/example-prefix/kms/xks/v1" \
    --xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,
 RawSecretAccessKey=DXjSUawnel2fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```

输出：

```
{
    "CustomKeyStoreId": cks-2234567890abcdef0
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的创建外部密钥存储。

示例 3：创建使用 VPC 端点服务连接的外部密钥存储

以下 `create-custom-key-store` 示例创建一个外部密钥存储（XKS），它使用 Amazon VPC 端点服务与 Amazon KMS 通信。

注意：如果使用 Amazon CLI 版本 1.0，请先运行以下命令，再指定具有 HTTP 或 HTTPS 值的参数，例如参数 XksProxyUriEndpoint。

```
aws configure set cli_follow_urlparam false
```

否则，Amazon CLI 版本 1.0 会将参数值替换为在该 URI 地址找到的内容。

```
aws kms create-custom-key-store \
    --custom-key-store-name ExampleVPCEndpointXKS \
    --custom-key-store-type EXTERNAL_KEY_STORE \
    --xks-proxy-connectivity VPC_ENDPOINT_SERVICE \
    --xks-proxy-uri-endpoint "https://myproxy-private.xks.example.com" \
    --xks-proxy-uri-path "/kms/xks/v1" \
    --xks-proxy-vpc-endpoint-service-name "com.amazonaws.vpce.us-east-1.vpce-svc-
example1" \
```

```
    --xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,
 RawSecretAccessKey=DXjSUawnel2fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```

输出：

```
{
    "CustomKeyStoreId": cks-3234567890abcdef0
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的创建外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomKeyStore。

create-grant

以下代码示例演示了如何使用 create-grant。

Amazon CLI

创建授权

以下 create-grant 示例将创建授权，以允许 exampleUser 用户对 1234abcd-12ab-34cd-56ef-1234567890ab KMS 密钥示例使用 decrypt 命令。停用主体是 adminRole 角色。该授权使用 EncryptionContextSubset 授权约束，以便仅在 decrypt 请求中的加密上下文包含 "Department": "IT" 键值对时才允许此权限。

```
aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::123456789012:user/exampleUser \
    --operations Decrypt \
    --constraints EncryptionContextSubset={Department=IT} \
    --retiring-principal arn:aws:iam::123456789012:role/adminRole
```

输出：

```
{
    "GrantId": "1a2b3c4d2f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
    "GrantToken": "<grant token here>"
}
```

要查看有关授权的详细信息，请使用 list-grants 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGrant。

create-key

以下代码示例演示了如何使用 create-key。

Amazon CLI

示例 1：在 Amazon KMS 中创建客户托管 KMS 密钥

以下 create-key 示例创建对称加密 KMS 密钥。

要创建基本 KMS 密钥（对称加密密钥），您无需指定任何参数。这些参数的默认值会创建对称加密密钥。

由于此命令未指定密钥策略，因此，KMS 密钥将获得以编程方式创建的 KMS 密钥的默认密钥策略。要查看密钥策略，请使用 get-key-policy 命令。要更改密钥策略，请使用 put-key-policy 命令。

```
aws kms create-key
```

create-key 命令会返回密钥元数据，包括新 KMS 密钥的密钥 ID 和 ARN。您可以使用这些值来识别其他 Amazon KMS 操作中的 KMS 密钥。输出不包括标签。要查看 KMS 密钥的标签，请使用 list-resource-tags command。

输出：

```
{
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2017-07-05T14:04:55-07:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_KMS"
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

注意：create-key 命令不允许您指定别名。要为新 KMS 密钥创建别名，请使用 create-alias 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的创建密钥。

示例 2：创建用于加密和解密的非对称 RSA KMS 密钥

以下 create-key 示例创建了一个 KMS 密钥，其中包含用于加密和解密的非对称 RSA 密钥对。创建密钥后，无法更改密钥规格和密钥用法：

```
aws kms create-key \
   --key-spec RSA_4096 \
   --key-usage ENCRYPT_DECRYPT
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2021-04-05T14:04:55-07:00",
        "CustomerMasterKeySpec": "RSA_4096",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
            "RSAES_OAEP_SHA_1",
            "RSAES_OAEP_SHA_256"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
        "KeyManager": "CUSTOMER",
        "KeySpec": "RSA_4096",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_KMS"
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

示例 3：创建用于签名和验证的非对称椭圆曲线 KMS 密钥

创建用于签名和验证的非对称 KMS 密钥，其中包含非对称椭圆曲线（ECC）密钥对。尽管 SIGN_VERIFY 是 ECC KMS 密钥的唯一有效值，但 --key-usage 参数也是必需的。创建密钥后，无法更改密钥规格和密钥用法：

```
aws kms create-key \
    --key-spec ECC_NIST_P521 \
    --key-usage SIGN_VERIFY
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2019-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "ECC_NIST_P521",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "ECC_NIST_P521",
        "KeyState": "Enabled",
        "KeyUsage": "SIGN_VERIFY",
        "MultiRegion": false,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ECDSA_SHA_512"
```

```
        ]
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

示例 4：创建用于签名和验证的非对称 ML-DSA KMS 密钥

此示例创建了一个用于签名和验证的模格数字签名算法（ML-DSA）密钥。即使 SIGN_VERIFY 是 ML-DSA 密钥的唯一有效值，key-usage 参数也是必需的。

```
aws kms create-key \
    --key-spec ML_DSA_65 \
    --key-usage SIGN_VERIFY
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2019-12-02T07:48:55-07:00",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "ML_DSA_65",
        "KeyState": "Enabled",
        "KeyUsage": "SIGN_VERIFY",
        "MultiRegion": false,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ML_DSA_SHAKE_256"
        ]
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

示例 5：创建 HMAC KMS 密钥

以下 create-key 示例将创建 384 位 HMAC KMS 密钥。尽管 --key-usage 参数的 GENERATE_VERIFY_MAC 值是 HMAC KMS 密钥的唯一有效值，但该值也是必需的。

```
aws kms create-key \
    --key-spec HMAC_384 \
    --key-usage GENERATE_VERIFY_MAC
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2022-04-05T14:04:55-07:00",
        "CustomerMasterKeySpec": "HMAC_384",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "HMAC_384",
        "KeyState": "Enabled",
        "KeyUsage": "GENERATE_VERIFY_MAC",
        "MacAlgorithms": [
            "HMAC_SHA_384"
        ],
        "MultiRegion": false,
        "Origin": "AWS_KMS"
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的 HMAC 密钥。

示例 6：创建多区域主 KMS 密钥

以下 create-key 示例创建多区域主对称加密密钥。由于所有参数的默认值都会创建对称加密密钥，因此，此 KMS 密钥只需要 --multi-region 参数。在 Amazon CLI 中，要指示布尔参数为 true，只需指定参数名称即可。

```
aws kms create-key \
    --multi-region
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef12345678990ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2021-09-02T016:15:21-09:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
          "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "mrk-1234abcd12ab34cd56ef12345678990ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": true,
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "PRIMARY",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef12345678990ab",
                "Region": "us-west-2"
            },
            "ReplicaKeys": []
        },
        "Origin": "AWS_KMS"
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

示例 7：为导入的密钥材料创建 KMS 密钥

以下 create-key 示例创建一个不带密钥材料的 KMS 密钥。操作完成后，您可以将自己的密钥材料导入 KMS 密钥。要创建此 KMS 密钥，请将 --origin 参数设置为 EXTERNAL。

```
aws kms create-key \
    --origin EXTERNAL
```

输出：

```
{
     "KeyMetadata": {
         "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
         "AWSAccountId": "111122223333",
         "CreationDate": "2019-12-02T07:48:55-07:00",
         "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
         "Description": "",
         "Enabled": false,
         "EncryptionAlgorithms": [
             "SYMMETRIC_DEFAULT"
         ],
         "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
         "KeyManager": "CUSTOMER",
         "KeySpec": "SYMMETRIC_DEFAULT",
         "KeyState": "PendingImport",
         "KeyUsage": "ENCRYPT_DECRYPT",
         "MultiRegion": false,
         "Origin": "EXTERNAL"
     }
 }
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的将密钥材料导入到 Amazon KMS 密钥中。

示例 6：在 Amazon CloudHSM 密钥存储中创建 KMS 密钥

以下 create-key 示例在指定的 Amazon CloudHSM 密钥存储中创建一个 KMS 密钥。该操作在 Amazon KMS 中创建 KMS 密钥及其元数据，并在与自定义密钥存储关联的 Amazon CloudHSM 集群中创建密钥材料。--custom-key-store-id 和 --origin 参数是必需的。

```
aws kms create-key \
    --origin AWS_CLOUDHSM \
```

```
    --custom-key-store-id cks-1234567890abcdef0
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CloudHsmClusterId": "cluster-1a23b4cdefg",
        "CreationDate": "2019-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "CustomKeyStoreId": "cks-1234567890abcdef0",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_CLOUDHSM"
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon CloudHSM 密钥存储。

示例 8：在外部密钥存储中创建 KMS 密钥

以下 create-key 示例在指定的外部密钥存储中创建一个 KMS 密钥。在此命令中需要使用 --custom-key-store-id、--origin 和 --xks-key-id 参数。

--xks-key-id 参数指定外部密钥管理器中现有对称加密密钥的 ID。此密钥用作 KMS 密钥的外部密钥材料。--origin 参数的值必须为 EXTERNAL_KEY_STORE。custom-key-store-id 参数必须识别连接到其外部密钥存储代理的外部密钥存储。

```
aws kms create-key \
    --origin EXTERNAL_KEY_STORE \
```

```
    --custom-key-store-id cks-9876543210fedcba9 \
    --xks-key-id bb8562717f809024
```

输出：

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2022-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "CustomKeyStoreId": "cks-9876543210fedcba9",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "EXTERNAL_KEY_STORE",
        "XksKeyConfiguration": {
            "Id": "bb8562717f809024"
        }
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKey。

`decrypt`

以下代码示例演示了如何使用 `decrypt`。

Amazon CLI

示例 1：使用对称 KMS 密钥解密加密消息（Linux 和 macOS）

以下 decrypt 命令示例演示了使用 Amazon CLI 解密数据的推荐方法。此版本演示了如何使用对称 KMS 密钥解密数据。

在文件中提供加密文字。在 --ciphertext-blob 参数的值中，使用 fileb:// 前缀，它将指示 CLI 从二进制文件中读取数据。如果文件不在当前目录中，请键入文件的完整路径。有关从文件读取 Amazon CLI 参数值的更多信息，请参阅以下内容：《Amazon 命令行界面用户指南》中的“从文件中加载 Amazon CLI 参数”<https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html>，以及《Amazon 命令行工具博客》中的“本地文件参数最佳实践”<https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/>。指定 KMS 密钥来解密加密文字。使用对称 KMS 密钥进行解密时不需要 --key-id参数。AmazonKMS 可以获取用于加密加密文字元数据中数据的 KMS 密钥的密钥 ID。但是，指定您正在使用的 KMS 密钥始终是最佳实践。此做法可确保您使用预期的 KMS 密钥，并防止您意外使用不信任的 KMS 密钥解密加密文字。以文本值请求明文输出。--query 参数指示 CLI 仅从输出中获取 Plaintext 字段的值。--output 参数以 text.base64 解码格式返回明文输出并将其保存在文件中。以下示例将 Plaintext 参数的值传送 (|) 给 Base64 实用程序，该程序负责对其进行解码。然后，它将解码后的输出重定向 (>) 到 ExamplePlaintext 文件。

在运行此命令之前，将密钥 ID 示例替换为您 Amazon 账户中的有效密钥 ID。

```
aws kms decrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --output text \
    --query Plaintext | base64 \
    --decode > ExamplePlaintextFile
```

此命令不生成任何输出。decrypt 命令的输出经过 base64 解码并保存在文件中。

有关更多信息，请参阅《Amazon 密钥管理服务 API Reference》中的解密。

示例 2：使用对称 KMS 密钥解密加密消息 (Windows 命令提示符)

以下示例与前一个示例相同，不同之处在于，它使用 certutil 实用程序对明文数据进行 Base64 解码。此过程需要两个命令，如以下示例所示。

在运行此命令之前，将密钥 ID 示例替换为您 Amazon 账户中的有效密钥 ID。

```
aws kms decrypt ^
    --ciphertext-blob fileb://ExampleEncryptedFile ^
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^
```

```
    --output text ^
    --query Plaintext > ExamplePlaintextFile.base64
```

运行 certutil 命令。

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

输出：

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

有关更多信息，请参阅《Amazon 密钥管理服务 API Reference》中的解密。

示例 3：使用非对称 KMS 密钥解密加密消息（Linux 和 macOS）

以下 decrypt 命令示例演示如何解密在 RSA 非对称 KMS 密钥下加密的数据。

使用非对称 KMS 密钥时，需要 encryption-algorithm 参数，以指定用于加密明文的算法。

在运行此命令之前，将密钥 ID 示例替换为您 Amazon 账户中的有效密钥 ID。

```
aws kms decrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --encryption-algorithm RSAES_OAEP_SHA_256 \
    --output text \
    --query Plaintext | base64 \
    --decode > ExamplePlaintextFile
```

此命令不生成任何输出。decrypt 命令的输出经过 base64 解码并保存在文件中。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的解密。

delete-alias

以下代码示例演示了如何使用 delete-alias。

Amazon CLI

删除 Amazon KMS 别名

以下 delete-alias 示例将删除别名 alias/example-alias。别名名称必须以 alias/ 开头。

```
aws kms delete-alias \
    --alias-name alias/example-alias
```

此命令不生成任何输出。要查找别名，请使用 list-aliases 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的删除别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAlias。

delete-custom-key-store

以下代码示例演示了如何使用 delete-custom-key-store。

Amazon CLI

删除自定义密钥存储

以下 delete-custom-key-store 示例删除指定自定义密钥存储。

删除 Amazon CloudHSM 密钥存储不会影响关联的 CloudHSM 集群。删除外部密钥存储不会影响关联的外部密钥存储代理、外部密钥管理器或外部密钥。

注意：要删除一个自定义密钥存储，必须先计划删除该自定义密钥存储中的所有 KMS 密钥，然后等待这些 KMS 密钥被删除。之后，必须断开连接该自定义密钥存储。有关在自定义密钥存储中查找 KMS 密钥的帮助，请参阅《Amazon 密钥管理服务开发人员指南》中的删除 Amazon CloudHSM 密钥存储（API）。

```
delete-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0
```

此命令不返回任何输出。要确认是否删除了该自定义密钥存储，请使用 describe-custom-key-stores 命令。

有关删除 Amazon CloudHSM 密钥存储的信息，请参阅《Amazon 密钥管理服务开发人员指南》中的删除 Amazon CloudHSM 密钥存储。

有关删除外部密钥存储的信息，请参阅《Amazon 密钥管理服务开发人员指南》中的删除外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCustomKeyStore。

delete-imported-key-material

以下代码示例演示了如何使用 delete-imported-key-material。

Amazon CLI

从 KMS 密钥中删除导入的密钥材料

以下 delete-imported-key-material 示例删除导入到 KMS 密钥的密钥材料。

```
aws kms delete-imported-key-material \
   --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

有关更多信息，请参阅《Amazon Key Management Service Developer Guide》中的 Deleting imported key material。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteImportedKeyMaterial。

derive-shared-secret

以下代码示例演示了如何使用 derive-shared-secret。

Amazon CLI

派生共享密钥

以下 derive-shared-secret 示例使用密钥协议算法派生共享密钥。

您必须使用 NIST 推荐的非对称椭圆曲线（ECC）或 SM2（仅限中国区域）KMS 密钥对且 KeyUsage 值为 KEY_AGREEMENT 来调用 DeriveSharedSecret。

```
aws kms derive-shared-secret \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-agreement-algorithm ECDH \
    --public-
key "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvH3Yj0wbkLEpUl95Cv1cJVjsVNSjwGq3tCLnzXfhVwV
```

输出：

```
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "SharedSecret": "MEYCIQCKZLWyTk5runarx6XiAkU9gv3lbwPO/pHa
+DXFehzdDwIhANwpsIV2g/9SPWLLsF6p/hiSskuIXMTRwqrMdVKWTMHG",
    "KeyAgreementAlgorithm": "ECDH",
    "KeyOrigin": "AWS_KMS"
}
```

有关更多信息，请参阅《Amazon 密钥管理服务 API 参考》中的 DeriveSharedSecret。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeriveSharedSecret。

describe-custom-key-stores

以下代码示例演示了如何使用 describe-custom-key-stores。

Amazon CLI

示例 1：获取有关 Amazon CloudHSM 密钥存储的详细信息

以下 describe-custom-key-store 示例显示有关指定 Amazon CloudHSM 密钥存储的详细信息。该命令对于所有类型的自定义密钥存储都是一样的，只是输出因密钥存储类型而异，并且对于外部密钥存储来说，还因其连接选项而异。

默认情况下，该命令显示有关账户和区域中的所有自定义密钥存储的信息。要显示有关特定自定义密钥存储的信息，请使用 custom-key-store-name 或 custom-key-store-id 参数。

```
aws kms describe-custom-key-stores \
    --custom-key-store-name ExampleCloudHSMKeyStore
```

该命令的输出中包括有关 Amazon CloudHSM 密钥存储的有用详细信息，包括其连接状态（ConnectionState）。如果连接状态为 FAILED，则输出中还包含描述问题的 ConnectionErrorCode 字段。

输出：

```
{
    "CustomKeyStores": [
        {
            "CloudHsmClusterId": "cluster-1a23b4cdefg",
            "ConnectionState": "CONNECTED",
            "CreationDate": "2022-04-05T14:04:55-07:00",
            "CustomKeyStoreId": "cks-1234567890abcdef0",
            "CustomKeyStoreName": "ExampleExternalKeyStore",
            "TrustAnchorCertificate": "<certificate appears here>"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的查看 Amazon CloudHSM 密钥存储。

示例 2：获取使用公有端点连接的外部密钥存储的详细信息

以下 describe-custom-key-store 示例显示有关指定外部密钥存储的详细信息。该命令对于所有类型的自定义密钥存储都是一样的，只是输出因密钥存储类型而异，并且对于外部密钥存储来说，还因其连接选项而异。

默认情况下，该命令显示有关账户和区域中的所有自定义密钥存储的信息。要显示有关特定自定义密钥存储的信息，请使用 custom-key-store-name 或 custom-key-store-id 参数。

```
aws kms describe-custom-key-stores \
    --custom-key-store-id cks-9876543210fedcba9
```

该命令的输出中包括有关外部密钥存储的有用详细信息，包括其连接状态（ConnectionState）。如果连接状态为 FAILED，则输出中还包含描述问题的 ConnectionErrorCode 字段。

输出：

```
{
```

```
    "CustomKeyStores": [
        {
            "CustomKeyStoreId": "cks-9876543210fedcba9",
            "CustomKeyStoreName": "ExampleXKS",
            "ConnectionState": "CONNECTED",
            "CreationDate": "2022-12-02T07:48:55-07:00",
            "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
            "XksProxyConfiguration": {
                "AccessKeyId": "ABCDE12345670EXAMPLE",
                "Connectivity": "PUBLIC_ENDPOINT",
                "UriEndpoint": "https://myproxy.xks.example.com",
                "UriPath": "/example-prefix/kms/xks/v1"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的查看外部密钥存储。

示例 3：获取使用 VPC 端点服务连接的外部密钥存储的详细信息

以下 describe-custom-key-store 示例显示有关指定外部密钥存储的详细信息。该命令对于所有类型的自定义密钥存储都是一样的，只是输出因密钥存储类型而异，并且对于外部密钥存储来说，还因其连接选项而异。

默认情况下，该命令显示有关账户和区域中的所有自定义密钥存储的信息。要显示有关特定自定义密钥存储的信息，请使用 custom-key-store-name 或 custom-key-store-id 参数。

```
aws kms describe-custom-key-stores \
    --custom-key-store-id cks-2234567890abcdef0
```

该命令的输出中包括有关外部密钥存储的有用详细信息，包括其连接状态（ConnectionState）。如果连接状态为 FAILED，则输出中还包含描述问题的 ConnectionErrorCode 字段。

输出：

```
{
    "CustomKeyStores": [
        {
            "CustomKeyStoreId": "cks-3234567890abcdef0",
```

```
            "CustomKeyStoreName": "ExampleVPCExternalKeyStore",
            "ConnectionState": "CONNECTED",
            "CreationDate": "2022-12-22T07:48:55-07:00",
            "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
            "XksProxyConfiguration": {
                "AccessKeyId": "ABCDE12345670EXAMPLE",
                "Connectivity": "VPC_ENDPOINT_SERVICE",
                "UriEndpoint": "https://myproxy-private.xks.example.com",
                "UriPath": "/kms/xks/v1",
                "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-
example1"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的查看外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCustomKeyStores。

describe-key

以下代码示例演示了如何使用 describe-key。

Amazon CLI

示例 1：查找有关 KMS 密钥的详细信息

以下 describe-key 示例获取有关账户示例和区域中 Amazon S3 Amazon 管理密钥的详细信息。您可以使用此命令来查找有关 Amazon 管理密钥和客户托管密钥的详细信息。

要指定 KMS 密钥，请使用 key-id 参数。此示例使用别名名称值，但您可以在此命令中使用密钥 ID、密钥 ARN、别名名称或别名 ARN。

```
aws kms describe-key \
    --key-id alias/aws/s3
```

输出：

```
{
    "KeyMetadata": {
```

```
        "AWSAccountId": "846764612917",
        "KeyId": "b8a9477d-836c-491f-857e-07937918959b",
        "Arn": "arn:aws:kms:us-west-2:846764612917:key/
b8a9477d-836c-491f-857e-07937918959b",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CreationDate": 2017-06-30T21:44:32.140000+00:00,
        "Enabled": true,
        "Description": "Default KMS key that protects my S3 objects when no other
 key is defined",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "AWS",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的查看密钥。

示例 2：获取有关 RSA 非对称 KMS 密钥的详细信息

以下 describe-key 示例获取有关用于签名和验证的非对称 RSA KMS 密钥的详细信息。

```
aws kms describe-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2019-12-02T19:47:14.861000+00:00",
        "CustomerMasterKeySpec": "RSA_2048",
        "Enabled": false,
        "Description": "",
        "KeyState": "Disabled",
```

```
        "Origin": "AWS_KMS",
        "MultiRegion": false,
        "KeyManager": "CUSTOMER",
        "KeySpec": "RSA_2048",
        "KeyUsage": "SIGN_VERIFY",
        "SigningAlgorithms": [
            "RSASSA_PKCS1_V1_5_SHA_256",
            "RSASSA_PKCS1_V1_5_SHA_384",
            "RSASSA_PKCS1_V1_5_SHA_512",
            "RSASSA_PSS_SHA_256",
            "RSASSA_PSS_SHA_384",
            "RSASSA_PSS_SHA_512"
        ]
    }
}
```

示例 3：获取有关多区域副本密钥的详细信息

以下 describe-key 示例获取多区域副本密钥的元数据。此多区域密钥是对称加密密钥。任何多区域密钥的 describe-key 命令输出都会返回有关主密钥及其所有副本的信息。

```
aws kms describe-key \
    --key-id arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab
```

输出：

```
{
    "KeyMetadata": {
        "MultiRegion": true,
        "AWSAccountId": "111122223333",
        "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "CreationDate": "2021-06-28T21:09:16.114000+00:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "Description": "",
        "Enabled": true,
        "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
```

```
        "Origin": "AWS_KMS",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "PRIMARY",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                "Region": "us-west-2"
            },
            "ReplicaKeys": [
                {
                    "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "eu-west-1"
                },
                {
                    "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "ap-northeast-1"
                },
                {
                    "Arn": "arn:aws:kms:sa-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "sa-east-1"
                }
            ]
        }
    }
}
```

示例 4：获取有关 HMAC KMS 密钥的详细信息

以下 `describe-key` 示例获取有关 HMAC KMS 密钥的详细信息。

```
aws kms describe-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
```

```
    "KeyMetadata": {
        "AWSAccountId": "123456789012",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Arn": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2022-04-03T22:23:10.194000+00:00",
        "Enabled": true,
        "Description": "Test key",
        "KeyUsage": "GENERATE_VERIFY_MAC",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "HMAC_256",
        "MacAlgorithms": [
            "HMAC_SHA_256"
        ],
        "MultiRegion": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeKey。

`disable-key-rotation`

以下代码示例演示了如何使用 `disable-key-rotation`。

Amazon CLI

禁用自动轮换 KMS 密钥

以下 `disable-key-rotation` 示例禁用自动轮换客户托管的 KMS 密钥。要重新启用自动轮换，请使用 `enable-key-rotation` 命令。

```
aws kms disable-key-rotation \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。要验证是否对 KMS 密钥禁用自动轮换，请使用 `get-key-rotation-status` 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableKeyRotation。

disable-key

以下代码示例演示了如何使用 disable-key。

Amazon CLI

暂时禁用 KMS 密钥

以下 disable-key 命令禁用客户自主管理型 KMS 密钥。要重新启用 KMS 密钥，请使用 enable-key 命令。

```
aws kms disable-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的启用和禁用密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableKey。

disconnect-custom-key-store

以下代码示例演示了如何使用 disconnect-custom-key-store。

Amazon CLI

断开连接自定义密钥存储

以下 disconnect-custom-key-store 示例断开一个自定义密钥存储与其 Amazon CloudHSM 集群的连接。您可以断开连接一个密钥存储，以便解决问题、更新其设置或防止该密钥存储中的 KMS 密钥被用于加密操作。

此命令对于所有自定义密钥存储都是一样的，包括 Amazon CloudHSM 密钥存储和外部密钥存储。

在运行此命令之前，请将示例自定义密钥存储 ID 替换为有效 ID。

```
$ aws kms disconnect-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0
```

此命令不生成任何输出。要验证命令是否有效，请使用 describe-custom-key-stores 命令。

有关断开连接 Amazon CloudHSM 密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的连接和断开连接 Amazon CloudHSM 密钥存储。

有关断开连接外部密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的连接和断开连接外部密钥存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisconnectCustomKeyStore。

enable-key-rotation

以下代码示例演示了如何使用 enable-key-rotation。

Amazon CLI

启用自动轮换 KMS 密钥

以下 enable-key-rotation 示例启用自动轮换客户托管的 KMS 密钥，轮换周期为 180 天。KMS 密钥将自该命令完成之日起一年（大约 365 天）以及此后每年进行轮换。

--key-id 参数标识 KMS 密钥。此示例使用密钥 ARN 值，但您可以使用 KMS 密钥的密钥 ID 或 ARN。--rotation-period-in-days 参数指定每次轮换日期之间的天数。可指定 90 到 2560 天之间的值。如果未指定值，则默认值为 365 天。

```
aws kms enable-key-rotation \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --rotation-period-in-days 180
```

此命令不生成任何输出。要验证 KMS 密钥是否已启用，请使用 get-key-rotation-status 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableKeyRotation。

enable-key

以下代码示例演示了如何使用 enable-key。

Amazon CLI

启用 KMS 密钥

以下 enable-key 示例启用客户托管密钥。您可以使用类似的命令来启用您通过 disable-key 命令暂时禁用的 KMS 密钥。您还可以使用它来启用已被禁用的 KMS 密钥，因为已计划删除该密钥且删除已取消。

要指定 KMS 密钥，请使用 key-id 参数。此示例使用密钥 ID 值，但您可以在此命令中使用密钥 ID 或密钥 ARN 值。

在运行此命令之前，将密钥 ID 示例替换为有效 ID。

```
aws kms enable-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。要验证 KMS 密钥是否已启用，请使用 describe-key 命令。查看 describe-key 输出中 KeyState 和 Enabled 字段的值。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的启用和禁用密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableKey。

encrypt

以下代码示例演示了如何使用 encrypt。

Amazon CLI

示例 1：在 Linux 或 macOS 上对文件内容进行加密

以下 encrypt 命令演示了使用 Amazon CLI 解密数据的推荐方法。

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob | base64 \
    --decode > ExampleEncryptedFile
```

该命令可以执行以下几项操作：

使用 --plaintext 参数来指示要加密的数据。此参数值必须经过 Base64 编码。plaintext 参数的值必须经过 base64 编码，或者您必须使用前缀 fileb://，以指示 Amazon CLI 从文件中读取二进制数据。如果文件不在当前目录中，请键入文件的完整路径。例如：fileb:///var/tmp/ExamplePlaintextFile 或 fileb://C:\Temp\ExamplePlaintextFile。有关从文件中读取 Amazon CLI 参数值的更多信息，请参阅《Amazon 命令行界面用户指南》中的从文件加载参数，以及 Amazon 命令行工具博客上的 Best Practices for Local File Parameters。使用 --output 和 --query 参数控制命令的输出。这些参数会从该命令的输出中提取称为加密文字的加密数据。有关控制输出的更多信息，请参阅《Amazon 命令行界面用户指南》中的控制命令输出。使用 base64 实用程序将提取的输出解码为二进制数据。成功的 encrypt 命令所返回的加密文字是 base64 编码的文本。必须先解码此文本，然后才能使用 Amazon CLI 对其进行解密。将二进制加密文字保存到文件中。命令（> ExampleEncryptedFile）的最后一部分将二进制加密文字保存到文件中，以便于解密。有关使用 Amazon CLI 解密数据的命令示例，请参阅解密示例。

示例 2：使用 Amazon CLI 在 Windows 上加密数据

此示例与前一个示例相同，不同之处在于，它使用 certutil 工具代替 base64。此过程需要两个命令，如以下示例所示。

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob > C:\Temp\ExampleEncryptedFile.base64

certutil -decode C:\Temp\ExampleEncryptedFile.base64 C:\Temp\ExampleEncryptedFile
```

示例 3：使用非对称 KMS 密钥进行加密

以下 encrypt 命令显示如何使用非对称 KMS 密钥加密明文。--encryption-algorithm 参数是必需的。与所有 encrypt CLI 命令一样，plaintext 参数必须经过 base64 编码，或者您必须使用 fileb:// 前缀，因为它指示 Amazon CLI 从文件中读取二进制数据。

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encryption-algorithm RSAES_OAEP_SHA_256 \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob | base64 \
    --decode > ExampleEncryptedFile
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Encrypt。

generate-data-key-pair-without-plaintext

以下代码示例演示了如何使用 generate-data-key-pair-without-plaintext。

Amazon CLI

生成 ECC NIST P384 非对称数据密钥对

以下 generate-data-key-pair-without-plaintext 示例请求一个 ECC NIST P384 密钥对以供在 Amazon 外部使用。

该命令在指定 KMS 密钥下返回一个明文公钥和一个加密私钥副本。它不返回明文私钥。您可以安全地将加密私钥与加密数据一起存储，并在需要使用它时调用 Amazon KMS 来解密私钥。

要请求 ECC NIST P384 非对称数据密钥对，请使用 key-pair-spec 参数且值设为 ECC_NIST_P384。

您指定的 KMS 密钥必须是对称加密 KMS 密钥，即 KeySpec 值为 SYMMETRIC_DEFAULT 的 KMS 密钥。

注意：本示例输出中的值被截断，便于显示。

```
aws kms generate-data-key-pair-without-plaintext \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-pair-spec ECC_NIST_P384
```

输出：

```
{
    "PrivateKeyCiphertextBlob": "AQIDAHi6LtupRpdKl2aJTzkK6FbhOtQkMlQJJH3PdtHvS/y
+hAFFxmiD134doUDzMGmfCEtcAAAHaTCCB2UGCSqGSIb3DQEHBqCCB1...",
    "PublicKey":
 "MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEA3A3eGMyPrvSn7+LdlJE1oUoQV5HpEuHAVbdOyND
+NmYDH/mL1OSIEuLrcdZ5hrMH4pk83r40l...",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
```

```
    "KeyPairSpec": "ECC_NIST_P384"
}
```

PublicKey 和 PrivateKeyCiphertextBlob 以 base64 编码的格式返回。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的数据密钥对。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateDataKeyPairWithoutPlaintext。

generate-data-key-pair

以下代码示例演示了如何使用 generate-data-key-pair。

Amazon CLI

生成 2048 位 RSA 非对称数据密钥对

以下 generate-data-key-pair 示例请求 2048 位 RSA 非对称数据密钥以供在 Amazon 外部使用。该命令在指定 KMS 密钥下返回一个明文公钥和一个明文私钥以供立即使用和删除，另外还返回一个加密私钥的副本。您可以安全地将加密私钥与加密数据一起存储。

要请求 2048 位 RSA 非对称数据密钥对，请使用 key-pair-spec 参数且值设为 RSA_2048。

您指定的 KMS 密钥必须是对称加密 KMS 密钥，即 KeySpec 值为 SYMMETRIC_DEFAULT 的 KMS 密钥。

注意：本示例输出中的值被截断，便于显示。

```
aws kms generate-data-key-pair \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-pair-spec RSA_2048
```

输出：

```
{
    "PrivateKeyCiphertextBlob": "AQIDAHi6LtupRpdKl2aJTzkK6FbhOtQkMlQJJH3PdtHvS/y
+hAFFxmiD134doUDzMGmfCEtcAAAHaTCCB2UGCSqGSIb3DQEHBqCCB1...",
    "PrivateKeyPlaintext": "MIIG/
QIBADANBgkqhkiG9w0BAQEFAASCBucwggbjAgEAAoIBgQDcDd4YzI
+u9Kfv4t2UkTWhShBXkekS4cBVt07I0P42ZgMf+YvU5IgS4ut...",
```

```
    "PublicKey":
 "MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEA3A3eGMyPrvSn7+LdlJE1oUoQV5HpEuHAVbdOyND
+NmYDH/mL1OSIEuLrcdZ5hrMH4pk83r40l...",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
    "KeyPairSpec": "RSA_2048"
}
```

PublicKey、PrivateKeyPlaintext 和 PrivateKeyCiphertextBlob 以 base64 编码的格式返回。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的数据密钥对。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateDataKeyPair。

generate-data-key-without-plaintext

以下代码示例演示了如何使用 generate-data-key-without-plaintext。

Amazon CLI

生成不带明文密钥的 256 位对称数据密钥

以下 generate-data-key-without-plaintext 示例请求 256 位对称数据密钥的加密副本以供在 Amazon 外部使用。准备好使用数据密钥时，可以调用 Amazon KMS 对其进行解密。

要请求 256 位数据密钥，请使用值为 AES_256 的 key-spec 参数。要请求 128 位数据密钥，请使用值为 AES_128 的 key-spec 参数。对于所有其他数据密钥长度，请使用 number-of-bytes 参数。

您指定的 KMS 密钥必须是对称加密 KMS 密钥，即 KeySpec 值为 SYMMETRIC_DEFAULT 的 KMS 密钥。

```
aws kms generate-data-key-without-plaintext \
    --key-id "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" \
    --key-spec AES_256
```

输出：

```
{
    "CiphertextBlob":
 "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgkqhk
    "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

CiphertextBlob（加密数据密钥）以 base64 编码的格式返回。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的数据密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateDataKeyWithoutPlaintext。

generate-data-key

以下代码示例演示了如何使用 generate-data-key。

Amazon CLI

示例 1：生成 256 位对称数据密钥

以下 generate-data-key 示例请求 256 位对称数据密钥以供在 Amazon 外部使用。该命令返回一个明文数据密钥以供立即使用和删除，以及以指定 KMS 密钥加密的该数据密钥的副本。您可以安全地将加密的数据密钥与加密的数据一起存储。

要请求 256 位数据密钥，请使用值为 AES_256 的 key-spec 参数。要请求 128 位数据密钥，请使用值为 AES_128 的 key-spec 参数。对于所有其他数据密钥长度，请使用 number-of-bytes 参数。

您指定的 KMS 密钥必须是对称加密 KMS 密钥，即 KeySpec 值为 SYMMETRIC_DEFAULT 的 KMS 密钥。

```
aws kms generate-data-key \
    --key-id alias/ExampleAlias \
    --key-spec AES_256
```

输出：

```
{
```

```
    "Plaintext": "VdzKNHGzUAzJeRBVY+uUmofUGGiDzyB3+i9fVkh3piw=",
    "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
    "CiphertextBlob":
 "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgkqhk
+YdhV8MrkBQPeac0ReRVNDt9qleAt+SHgIRF8P0H+7U="
}
```

Plaintext（明文数据密钥）和 CiphertextBlob（加密数据密钥）均以 base64 编码的格式返回。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的数据密钥。示例 2：生成 512 位对称数据密钥

以下 generate-data-key 示例请求用于加密和解密的 512 位对称数据密钥。该命令返回一个明文数据密钥以供立即使用和删除，以及以指定 KMS 密钥加密的该数据密钥的副本。您可以安全地将加密的数据密钥与加密的数据一起存储。

要请求 128 或 256 位以外的密钥长度，请使用 number-of-bytes 参数。为了请求 512 位数据密钥，以下示例使用值为 64（字节）的 number-of-bytes 参数。

您指定的 KMS 密钥必须是对称加密 KMS 密钥，即密钥规格值为 SYMMETRIC_DEFAULT 的 KMS 密钥。

注意：本示例输出中的值被截断，便于显示。

```
aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --number-of-bytes 64
```

输出：

```
{
    "CiphertextBlob": "AQIBAHi6LtupRpdKl2aJTzkK6FbhOtQkMlQJJH3PdtHvS/y+hAEnX/
QQNmMwDfg2korNMEc8AAACaDCCAmQGCSqGSIb3DQEHBqCCAlUwggJRAgEAMIICSgYJKoZ...",
    "Plaintext": "ty8Lr0Bk6OF07M2BWt6qbFdNB
+G00ZLtf5MSEb4al3R2UKWGOp06njAwy2n72VRm2m7z/
Pm9Wpbvttz6a4lSo9hgPvKhZ5y6RTm4OovEXiVfBveyX3DQxDzRSwbKDPk/...",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

Plaintext（明文数据密钥）和 CiphertextBlob（加密数据密钥）均以 base64 编码的格式返回。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的数据密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateDataKey。

generate-mac

以下代码示例演示了如何使用 generate-mac。

Amazon CLI

示例 1：为消息生成 HMAC

以下 generate-mac 命令为消息、HMAC KMS 密钥和 MAC 算法生成 HMAC。该算法必须受指定的 HMAC KMS 密钥支持。

在 Amazon CLI v2 中，message 参数的值必须采用 Base64 编码。或者，您可以将消息保存在文件中并使用 fileb:// 前缀，它告诉 Amazon CLI 从文件中读取二进制数据。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。密钥 ID 必须表示密钥用法为 GENERATE_VERIFY_MAC 的 HMAC KMS 密钥。

```
msg=(echo 'Hello World' | base64)

aws kms generate-mac \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://Message \
    --mac-algorithm HMAC_SHA_384
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "Mac": "<HMAC_TAG>",
```

```
    "MacAlgorithm": "HMAC_SHA_384"
}
```

有关在 Amazon KMS 中使用 HMAC KMS 密钥的更多信息，请参阅《Amazon Key Management Service Developer Guide》中的 HMAC keys in Amazon KMS。

示例 2：将 HMAC 保存在文件中（Linux 和 macOs）

以下 `generate-mac` 示例为一条存储在本地文件中的短消息生成 HMAC。该命令还从响应中获取 `Mac` 属性，对其进行 Base64 解码并将其保存在 ExampleMac 文件中。您可以在验证 MAC 的 `verify-mac` 命令中使用该 MAC 文件。

`generate-mac` 命令需要一条以 Base64 编码的消息和您的 HMAC KMS 密钥支持的 MAC 算法。要获取 KMS 密钥支持的 MAC 算法，请使用 `describe-key` 命令。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。密钥 ID 必须表示密钥用法为 GENERATE_VERIFY_MAC 的非对称 KMS 密钥。

```
echo 'hello world' | base64 > EncodedMessage

aws kms generate-mac \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://EncodedMessage \
    --mac-algorithm HMAC_SHA_384 \
    --output text \
    --query Mac | base64 --decode > ExampleMac
```

此命令不生成任何输出。此示例提取输出的 `Mac` 属性并将其保存在文件中。

有关在 Amazon KMS 中使用 HMAC KMS 密钥的更多信息，请参阅《Amazon Key Management Service Developer Guide》中的 HMAC keys in Amazon KMS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateMac。

`generate-random`

以下代码示例演示了如何使用 `generate-random`。

Amazon CLI

示例 1：生成 256 位随机字节字符串（Linux 或 macOS）

以下 generate-random 示例生成 256 位（32 字节）、以 base64 编码的随机字节字符串。该示例对字节字符串进行解码并将其保存在随机文件中。

运行此命令时，您必须使用 number-of-bytes 参数指定随机值的长度（以字节为单位）。

在运行此命令时，无需指定 KMS 密钥。随机字节字符串与任何 KMS 密钥无关。

默认情况下，Amazon KMS 会生成随机数。但是，如果您指定 custom key store，则将在与该自定义密钥存储关联的 Amazon CloudHSM 集群中生成随机字节字符串。

本示例使用以下参数和值：

它使用值为 32 的必需 --number-of-bytes 参数来请求 32 字节（256 位）字符串。它使用值为 text 的 --output 参数来指示 Amazon CLI 将输出作为文本（而不是 JSON）返回。它使用 --query parameter 从响应中提取 Plaintext 属性的值。它将命令的输出传送（|）到 base64 实用程序，该程序负责对提取的输出进行解码。它使用重定向运算符（>）将解码后的字节字符串保存到 ExampleRandom 文件中。它使用重定向运算符（>）将二进制加密文字保存到文件。

```
aws kms generate-random \
    --number-of-bytes 32 \
    --output text \
    --query Plaintext | base64 --decode > ExampleRandom
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 密钥管理服务 API Reference》中的 GenerateRandom。

示例 2：生成 256 位随机数（Windows 命令提示符）

以下示例使用 generate-random 命令生成 256 位（32 字节）、以 base64 编码的随机字节字符串。该示例对字节字符串进行解码并将其保存在随机文件中。此示例与前面的示例相同，不同之处在于：它在将随机字节字符串保存到文件之前，使用 Windows 中的 certutil 实用工具对随机字节字符串进行 base64 解码。

首先，生成一个 base64 编码的随机字节字符串并将其保存在临时文件 ExampleRandom.base64 中。

```
aws kms generate-random \
    --number-of-bytes 32 \
    --output text \
    --query Plaintext > ExampleRandom.base64
```

由于 generate-random 命令的输出保存在文件中，因此，此示例不生成任何输出。

现在，使用 certutil -decode 命令解码 ExampleRandom.base64 文件中以 base64 编码的字节字符串。然后，它将解码后的字节字符串保存在 ExampleRandom 文件中。

```
certutil -decode ExampleRandom.base64 ExampleRandom
```

输出：

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

有关更多信息，请参阅《Amazon 密钥管理服务 API Reference》中的 GenerateRandom。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateRandom。

get-key-policy

以下代码示例演示了如何使用 get-key-policy。

Amazon CLI

将密钥策略从一个 KMS 密钥复制到另一个 KMS 密钥

以下 get-key-policy 示例从一个 KMS 密钥获取密钥策略并将其保存在文本文件中。然后，它使用文本文件作为策略输入替换其他 KMS 密钥的策略。

由于 put-key-policy 的 --policy 参数需要字符串，因此，您必须使用 --output text 选项将输出作为文本字符串（而不是 JSON）返回。

```
aws kms get-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --query Policy \
    --output text > policy.txt

aws kms put-key-policy \
    --policy-name default \
    --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --policy file://policy.txt
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon KMS API Reference》中的 PutKeyPolicy。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetKeyPolicy。

get-key-rotation-status

以下代码示例演示了如何使用 get-key-rotation-status。

Amazon CLI

检索 KMS 密钥的轮换状态

以下 get-key-rotation-status 示例返回有关指定 KMS 密钥轮换状态的信息，包括是否启用自动轮换、轮换周期以及下个预定轮换日期。您可以对客户托管的 KMS 密钥和 Amazon 托管的 KMS 密钥使用此命令。不过，所有 Amazon 托管的 KMS 密钥每年都会自动轮换。

```
aws kms get-key-rotation-status \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyRotationEnabled": true,
    "NextRotationDate": "2024-02-14T18:14:33.587000+00:00",
    "RotationPeriodInDays": 365
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetKeyRotationStatus。

get-parameters-for-import

以下代码示例演示了如何使用 get-parameters-for-import。

Amazon CLI

获取将密钥材料导入 KMS 密钥所需的事项

以下 get-parameters-for-import 示例获取将密钥材料导入 KMS 密钥所需的公钥和导入令牌。在使用 import-key-material 命令时，请务必使用同一 get-parameters-for-import 命令中返回的导入令牌和由公钥加密的密钥材料。此外，您在此命令中指定的包装算法必须和使用公钥加密密钥材料的算法一样。

要指定 KMS 密钥，请使用 key-id 参数。此示例使用密钥 ID，但您可以在此命令中使用密钥 ID 或密钥 ARN。

```
aws kms get-parameters-for-import \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --wrapping-algorithm RSAES_OAEP_SHA_256 \
    --wrapping-key-spec RSA_2048
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "PublicKey": "<public key base64 encoded data>",
    "ImportToken": "<import token base64 encoded data>",
    "ParametersValidTo": 1593893322.32
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的下载公钥和导入令牌。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetParametersForImport。

get-public-key

以下代码示例演示了如何使用 get-public-key。

Amazon CLI

示例 1：下载非对称 KMS 密钥的公钥

以下 get-public-key 示例下载非对称 KMS 密钥的公钥。

除了返回公钥外，输出中还包括在 Amazon KMS 之外安全使用公钥所需的信息，包括密钥用法和支持的加密算法。

```
aws kms get-public-key \
```

```
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "PublicKey": "jANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAl5epvg1/
QtJhxSi2g9SDEVg8QV/...",
    "CustomerMasterKeySpec": "RSA_4096",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "EncryptionAlgorithms": [
        "RSAES_OAEP_SHA_1",
        "RSAES_OAEP_SHA_256"
    ]
}
```

有关在 Amazon KMS 中使用非对称 KMS 密钥的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。示例 2：将公钥转换为 DER 格式（Linux 和 macOS）

以下 `get-public-key` 示例下载非对称 KMS 密钥的公钥并将其保存在 DER 文件中。

当您在 Amazon CLI 中使用 `get-public-key` 命令时，它会返回一个经过 Base64 编码的 DER 编码的 X.509 公钥。此示例获取文本形式的 `PublicKey` 属性的值。它对 `PublicKey` 进行 Base64 解码并将其保存在 `public_key.der` 文件中。`output` 参数返回文本形式的输出，而不是 JSON。`--query` 参数仅获取 `PublicKey` 属性，而不获取在 Amazon KMS 之外安全使用公钥所需的属性。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。

```
aws kms get-public-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --output text \
    --query PublicKey | base64 --decode > public_key.der
```

此命令不生成任何输出。

有关在 Amazon KMS 中使用非对称 KMS 密钥的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicKey。

import-key-material

以下代码示例演示了如何使用 import-key-material。

Amazon CLI

将密钥材料导入 KMS 密钥

以下 import-key-material 示例将密钥材料上传到创建时不含任何密钥材料的 KMS 密钥中。KMS 密钥的密钥状态必须为 PendingImport。

此命令使用的密钥材料是您用 get-parameters-for-import 命令返回的公钥加密过的材料。它还使用来自同一 get-parameters-for-import 命令的导入令牌。

expiration-model 参数表示密钥材料会在 valid-to 参数指定的日期和时间自动过期。当密钥材料过期后，Amazon KMS 会删除密钥材料，KMS 密钥的密钥状态变为 Pending import，而 KMS 密钥变为不可用。要还原 KMS 密钥，您必须重新导入这一密钥材料。要使用不同的密钥材料，您必须创建新的 KMS 密钥。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID 或密钥 ARN。

```
aws kms import-key-material \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
    --import-token fileb://ImportToken.bin \
    --expiration-model KEY_MATERIAL_EXPIRES \
    --valid-to 2021-09-21T19:00:00Z
```

输出：

```
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

有关导入密钥材料的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的导入密钥材料。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportKeyMaterial。

list-aliases

以下代码示例演示了如何使用 list-aliases。

Amazon CLI

示例 1：列出 Amazon 账户和区域中的所有别名

以下示例使用 list-aliases 命令列出 Amazon 账户默认区域中的所有别名。输出包含与 Amazon 托管的 KMS 密钥和客户托管 KMS 密钥关联的别名。

```
aws kms list-aliases
```

输出：

```
{
    "Aliases": [
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/testKey",
            "AliasName": "alias/testKey",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/FinanceDept",
            "AliasName": "alias/FinanceDept",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
        },
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
            "AliasName": "alias/aws/dynamodb",
            "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
        },
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
            "AliasName": "alias/aws/ebs",
            "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef"
        },
        ...
    ]
}
```

示例 2：列出特定 KMS 密钥的所有别名

以下示例使用 `list-aliases` 命令及其 `key-id` 参数列出与特定 KMS 密钥关联的所有别名。

每个别名都仅与一个 KMS 密钥关联，但一个 KMS 密钥可以有多个别名。此命令非常有用，因为 Amazon KMS 控制台仅列出每个 KMS 密钥的一个别名。要查找 KMS 密钥的所有别名，您必须使用 `list-aliases` 命令。

此示例使用 KMS 密钥的密钥 ID 作为 `--key-id` 参数，但您可以在此命令中使用密钥 ID、密钥 ARN、别名名称或别名 ARN。

```
aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "Aliases": [
        {
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/oregon-test-key",
            "AliasName": "alias/oregon-test-key"
        },
        {
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project121-test",
            "AliasName": "alias/project121-test"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的使用别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAliases。

`list-grants`

以下代码示例演示了如何使用 `list-grants`。

Amazon CLI

查看 Amazon KMS 密钥的授权

以下 list-grants 示例显示了您账户中针对 Amazon DynamoDB 的指定 Amazon 托管的 KMS 密钥的所有授权。该授权允许 DynamoDB 在将 DynamoDB 表写入磁盘之前代表您使用 KMS 密钥对其进行加密。您可以使用此类命令来查看 Amazon 账户和区域中 Amazon 托管的 KMS 密钥和客户托管 KMS 密钥的授权。

此命令使用带有密钥 ID 的 key-id 参数来识别 KMS 密钥。您可以使用密钥 ID 或密钥 ARN 来识别 KMS 密钥。要获取 Amazon 托管的 KMS 密钥的密钥 ID 或密钥 ARN，请使用 list-keys 或 list-aliases 命令。

```
aws kms list-grants \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出显示，该授权向 Amazon DynamoDB 授予了使用 KMS 密钥进行加密操作的权限，并向其授予了查看有关 KMS 密钥的详细信息（DescribeKey）和停用授权（RetireGrant）的权限。EncryptionContextSubset 约束将这些权限限制为包含指定加密上下文对的请求。因此，授权中的权限仅对指定账户和 DynamoDB 表有效。

```
{
    "Grants": [
        {
            "Constraints": {
                "EncryptionContextSubset": {
                    "aws:dynamodb:subscriberId": "123456789012",
                    "aws:dynamodb:tableName": "Services"
                }
            },
            "IssuingAccount": "arn:aws:iam::123456789012:root",
            "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
            "Operations": [
                "Decrypt",
                "Encrypt",
                "GenerateDataKey",
                "ReEncryptFrom",
                "ReEncryptTo",
                "RetireGrant",
                "DescribeKey"
            ],
            "GrantId":
 "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59",
            "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
```

```
            "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
            "CreationDate": "2021-05-13T18:32:45.144000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGrants。

`list-key-policies`

以下代码示例演示了如何使用 `list-key-policies`。

Amazon CLI

获取 KMS 密钥的密钥策略名称

以下 `list-key-policies` 示例获取示例账户和区域中客户托管密钥的密钥策略名称。您可以使用此命令来查找有关 Amazon 托管的密钥和客户托管密钥的密钥策略名称。

由于唯一有效的密钥策略名称是 `default`，因此，此命令没有用。

要指定 KMS 密钥，请使用 `key-id` 参数。此示例使用密钥 ID 值，但您可以在此命令中使用密钥 ID 或密钥 ARN。

```
aws kms list-key-policies \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "PolicyNames": [
    "default"
    ]
}
```

有关 Amazon KMS 密钥策略的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的在 Amazon KMS 中使用密钥策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListKeyPolicies。

list-key-rotations

以下代码示例演示了如何使用 list-key-rotations。

Amazon CLI

检索有关所有已完成的密钥材料轮换的信息

以下 list-key-rotations 示例列出指定 KMS 密钥的所有已完成密钥材料轮换的信息。

```
aws kms list-key-rotations \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "Rotations": [
        {
            "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "RotationDate": "2024-03-02T10:11:36.564000+00:00",
            "RotationType": "AUTOMATIC"
        },
        {
            "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "RotationDate": "2024-04-05T15:14:47.757000+00:00",
            "RotationType": "ON_DEMAND"
        }
    ],
    "Truncated": false
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListKeyRotations。

list-keys

以下代码示例演示了如何使用 list-keys。

Amazon CLI

获取账户和区域中的 KMS 密钥

以下 `list-keys` 示例获取账户和区域中的 KMS 密钥。此命令返回 Amazon 托管的密钥和客户托管密钥。

```
aws kms list-keys
```

输出：

```
{
    "Keys": [
        {
            "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
            "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
        },
        {
            "KeyArn": "arn:aws:kms:us-
east-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",
            "KeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的查看密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListKeys。

`list-resource-tags`

以下代码示例演示了如何使用 `list-resource-tags`。

Amazon CLI

获取 KMS 密钥上的标签

以下 `list-resource-tags` 示例获取一个 KMS 密钥的标签。要在 KMS 密钥上添加或更换资源标签，请使用 `tag-resource` 命令。输出中显示此 KMS 密钥有两个资源标签，每个标签都有一个键和值。

要指定 KMS 密钥，请使用 key-id 参数。此示例使用密钥 ID 值，但您可以在此命令中使用密钥 ID 或密钥 ARN。

```
aws kms list-resource-tags \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "Tags": [
    {
        "TagKey": "Dept",
        "TagValue": "IT"
    },
    {
        "TagKey": "Purpose",
        "TagValue": "Test"
    }
    ],
    "Truncated": false
}
```

有关在 Amazon KMS 中使用标签的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的标记密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceTags。

list-retirable-grants

以下代码示例演示了如何使用 list-retirable-grants。

Amazon CLI

查看主体可以停用的授权

以下 list-retirable-grants 示例显示在一个 Amazon 账户和区域中，ExampleAdmin 用户可以停用的 KMS 密钥的所有授权。您可以使用此类命令来查看有关 Amazon 账户和区域中任何账户主体都可以停用的 KMS 密钥的授权。

retiring-principal 参数为必需的，且其值必须是账户、用户或角色的 Amazon 资源名称（ARN）。

即使服务可以是停用主体，您也不能在此命令中指定服务作为 retiring-principal 的值。要查找在哪些授权中特定服务为停用主体，请使用 list-grants 命令。

输出显示在该账户和区域中，ExampleAdmin 用户有权停用两个不同 KMS 密钥的授权。除了停用主体外，该账户还有权停用账户中的任何授权。

```
aws kms list-retirable-grants \
    --retiring-principal arn:aws:iam::111122223333:user/ExampleAdmin
```

输出：

```
{
    "Grants": [
        {
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "GrantId":
 "156b69c63cb154aa21f59929ff19760717be8d9d82b99df53e18b94a15a5e88e",
            "Name": "",
            "CreationDate": 2021-01-14T20:17:36.419000+00:00,
            "GranteePrincipal": "arn:aws:iam::111122223333:user/ExampleUser",
            "RetiringPrincipal": "arn:aws:iam::111122223333:user/ExampleAdmin",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "Operations": [
                "Encrypt"
            ],
            "Constraints": {
                "EncryptionContextSubset": {
                    "Department": "IT"
                }
            }
        },
        {
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
            "GrantId":
 "8c94d1f12f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
            "Name": "",
            "CreationDate": "2021-02-02T19:49:49.638000+00:00",
            "GranteePrincipal": "arn:aws:iam::111122223333:role/ExampleRole",
            "RetiringPrincipal": "arn:aws:iam::111122223333:user/ExampleAdmin",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "Operations": [
```

```
                "Decrypt"
            ],
            "Constraints": {
                "EncryptionContextSubset": {
                    "Department": "IT"
                }
            }
        }
    ],
    "Truncated": false
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRetirableGrants。

put-key-policy

以下代码示例演示了如何使用 put-key-policy。

Amazon CLI

更改 KMS 密钥的密钥策略

以下 put-key-policy 示例更改客户托管密钥的密钥策略。

首先，创建密钥策略并将其保存在本地 JSON 文件中。在本示例中，该文件为 key_policy.json。您也可以将密钥策略指定为 policy 参数的字符串值。

此密钥策略中的第一条语句向 Amazon 账户授予使用 IAM 策略来控制对 KMS 密钥的访问的权限。第二条语句向 test-user 用户授予针对 KMS 密钥运行 describe-key 和 list-keys 命令的权限。

的内容key_policy.json：

```
{
    "Version":"2012-10-17",
    "Id" : "key-default-1",
    "Statement" : [
        {
            "Sid" : "Enable IAM User Permissions",
            "Effect" : "Allow",
            "Principal" : {
```

```
                "AWS" : "arn:aws:iam::111122223333:root"
            },
            "Action" : "kms:*",
            "Resource" : "*"
        },
        {
            "Sid" : "Allow Use of Key",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:user/test-user"
            },
            "Action" : [
                "kms:DescribeKey",
                "kms:ListKeys"
            ],
            "Resource" : "*"
        }
    ]
}
```

为了标识 KMS 密钥，此示例使用了密钥 ID，但您也可以使用密钥 ARN。为了指定密钥策略，该命令使用 policy 参数。为了表示策略位于文件中，它使用所需的 file:// 前缀。需要使用此前缀来识别所有受支持操作系统上的文件。最后，该命令使用值为 default 的 policy-name 参数。如果未指定策略名称，则默认值为 default。唯一有效值为 default。

```
aws kms put-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --policy file://key_policy.json
```

此命令不生成任何输出。要验证命令是否有效，请使用 get-key-policy 命令。以下示例命令获取相同 KMS 密钥的密钥策略。值为 text 的 output 参数返回一种易于读取的文本格式。

```
aws kms get-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --output text
```

输出：

```
{
```

```
    "Version":"2012-10-17",
    "Id" : "key-default-1",
    "Statement" : [
        {
            "Sid" : "Enable IAM User Permissions",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:root"
            },
            "Action" : "kms:*",
            "Resource" : "*"
            },
            {
            "Sid" : "Allow Use of Key",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:user/test-user"
            },
            "Action" : [ "kms:Describe", "kms:List" ],
            "Resource" : "*"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的更改密钥策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutKeyPolicy。

`re-encrypt`

以下代码示例演示了如何使用 `re-encrypt`。

Amazon CLI

示例 1：使用不同的对称 KMS 密钥重新加密加密消息（Linux 和 macOS）。

以下 `re-encrypt` 命令示例演示了使用 Amazon CLI 重新加密数据的推荐方法。

在文件中提供加密文字。在 `--ciphertext-blob` 参数的值中，使用 `fileb://` 前缀，它将指示 CLI 从二进制文件中读取数据。如果文件不在当前目录中，请键入文件的完整路径。有关从文件中读取 Amazon CLI 参数值的更多信息，请参阅以下内容：《Amazon Command Line Interface 用户指南》中的从文件加载 Amazon CLI 参数和《Amazon Command Line Tool Blog》中的 Best Practices for Local File Parameters。指定用于解密加密文字的源 KMS 密钥。使用对称加密 KMS

密钥解密时，不需要 --source-key-id 参数。AmazonKMS 可以获取用于加密加密文字 Blob 元数据中数据的 KMS 密钥。但是，指定您正在使用的 KMS 密钥始终是最佳实践。此做法可确保您使用预期的 KMS 密钥，并防止您意外使用不信任的 KMS 密钥解密加密文字。指定目标 KMS 密钥来重新加密数据。--destination-key-id 参数始终为必需项。此示例使用密钥 ARN，但您可以使用任何有效的密钥标识符。将明文输出请求为文本值。--query 参数指示 CLI 仅从输出中获取 Plaintext 字段的值。--output 参数以 text.base64 解码格式返回明文输出并将其保存在文件中。以下示例将 Plaintext 参数的值传送（|）给 Base64 实用工具，该程序负责对其进行解码。然后，它将解码后的输出重定向（>）到 ExamplePlaintext 文件。

在运行此命令之前，将示例密钥 ID 替换为您 Amazon 账户中的有效密钥标识符。

```
aws kms re-encrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --query CiphertextBlob \
    --output text | base64 --decode > ExampleReEncryptedFile
```

此命令不生成任何输出。re-encrypt 命令的输出经过 base64 解码并保存在文件中。

有关更多信息，请参阅《Amazon Key Management Service API Reference》中的 ReEncrypt。

示例 2：使用不同的对称 KMS 密钥重新加密加密消息（Windows 命令提示符）。

以下 re-encrypt 命令示例与前一个示例相同，不同之处在于，它使用 certutil 实用工具对明文数据进行 Base64 解码。此过程需要两个命令，如以下示例所示。

在运行此命令之前，将密钥 ID 示例替换为您 Amazon 账户中的有效密钥 ID。

```
aws kms re-encrypt ^
    --ciphertext-blob fileb://ExampleEncryptedFile ^
    --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^
    --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 ^
    --query CiphertextBlob ^
    --output text > ExampleReEncryptedFile.base64
```

然后使用 certutil 实用工具

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

输出：

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

有关更多信息，请参阅《Amazon Key Management Service API Reference》中的 ReEncrypt。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReEncrypt。

`retire-grant`

以下代码示例演示了如何使用 `retire-grant`。

Amazon CLI

停用对客户主密钥的授权

以下 `retire-grant` 示例从 KMS 密钥中删除授权。

以下示例命令指定 `grant-id` 和 `key-id` 参数。`key-id` 参数值必须为 KMS 密钥的密钥 ARN。

```
aws kms retire-grant \
    --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。要确认授权是否已停用，请使用 `list-grants` 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的停用和撤销授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetireGrant。

`revoke-grant`

以下代码示例演示了如何使用 `revoke-grant`。

Amazon CLI

撤销对客户主密钥的授权

以下 `revoke-grant` 示例从 KMS 密钥中删除授权。以下示例命令指定 `grant-id` 和 `key-id` 参数。`key-id` 参数的值可以是 KMS 密钥的密钥 ID 或密钥 ARN。

```
aws kms revoke-grant \
    --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。要确认授权是否已撤销，请使用 `list-grants` 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的停用和撤销授权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeGrant。

`rotate-key-on-demand`

以下代码示例演示了如何使用 `rotate-key-on-demand`。

Amazon CLI

执行按需轮换 KMS 密钥

以下 `rotate-key-on-demand` 示例立即启动指定 KMS 密钥的密钥材料轮换。

```
aws kms rotate-key-on-demand \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

输出：

```
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的如何执行按需密钥轮换。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RotateKeyOnDemand。

`schedule-key-deletion`

以下代码示例演示了如何使用 `schedule-key-deletion`。

Amazon CLI

计划删除客户托管的 KMS 密钥

以下 `schedule-key-deletion` 示例计划在 15 天后删除指定的客户托管 KMS 密钥。

--key-id 参数识别 KMS 密钥。此示例使用密钥 ARN 值，但您可以使用 KMS 密钥的密钥 ID 或 ARN。--pending-window-in-days 参数指定 7-30 天的等待期限。默认的等待期限为 30 天。此示例指定的值为 15，用于指示 Amazon 在命令完成 15 天后永久删除 KMS 密钥。

```
aws kms schedule-key-deletion \
    --key-id arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --pending-window-in-days 15
```

响应包括密钥 ARN、密钥状态、等待期（PendingWindowInDays）和删除日期（以 Unix 时间表示）。要以当地时间查看删除日期，请使用 Amazon KMS 控制台。无法在加密操作中使用密钥状态为 PendingDeletion 的 KMS 密钥。

```
{
    "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "DeletionDate": "2022-06-18T23:43:51.272000+00:00",
    "KeyState": "PendingDeletion",
    "PendingWindowInDays": 15
}
```

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的删除密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ScheduleKeyDeletion。

sign

以下代码示例演示了如何使用 sign。

Amazon CLI

示例 1：为消息生成数字签名

以下 sign 示例为一条短消息生成加密签名。该命令的输出包括一个 base-64 编码的 Signature 字段，您可以使用 verify 命令对其进行验证。

您必须指定要签名的消息以及您的非对称 KMS 密钥支持的签名算法。要获取 KMS 密钥的签名算法，请使用 describe-key 命令。

在 Amazon CLI v2 中，message 参数的值必须采用 Base64 编码。或者，您可以将消息保存在文件中并使用 fileb:// 前缀，它告诉 Amazon CLI 从文件中读取二进制数据。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。密钥 ID 必须表示密钥用法为 SIGN_VERIFY 的非对称 KMS 密钥。

```
msg=(echo 'Hello World' | base64)

aws kms sign \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://UnsignedMessage \
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "Signature": "ABCDEFhpyVYyTxbafE74ccSvEJLJr3zuoV1Hfymz4qv+/
fxmxNLA7SE1SiF8lHw80fKZZ3bJ...",
    "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

有关在 Amazon KMS 中使用非对称 KMS 密钥的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

示例 2：将数字签名保存在文件中（Linux 和 macOS）

以下 `sign` 示例为一条存储在本地文件中的短消息生成加密签名。该命令还从响应中获取 `Signature` 属性，对其进行 Base64 解码并将其保存在 ExampleSignature 文件中。您可以在验证签名的 `verify` 命令中使用该签名文件。

`sign` 命令需要一条 Base64 编码的消息和您的非对称 KMS 密钥支持的签名算法。要获取 KMS 密钥支持的签名算法，请使用 `describe-key` 命令。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。密钥 ID 必须表示密钥用法为 SIGN_VERIFY 的非对称 KMS 密钥。

```
echo 'hello world' | base64 > EncodedMessage

aws kms sign \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://EncodedMessage \
```

```
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
    --output text \
    --query Signature | base64 --decode > ExampleSignature
```

此命令不生成任何输出。此示例提取输出的 Signature 属性并将其保存在文件中。

有关在 Amazon KMS 中使用非对称 KMS 密钥的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的 Amazon KMS 中的非对称密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Sign。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为 KMS 密钥添加标签

以下 tag-resource 示例为客户托管的 KMS 密钥添加 "Purpose":"Test" 和 "Dept":"IT" 标签。您可以使用此类标签来标记 KMS 密钥并创建 KMS 密钥类别以进行权限管理和审计。

要指定 KMS 密钥，请使用 key-id 参数。此示例使用密钥 ID 值，但您可以在此命令中使用密钥 ID 或密钥 ARN。

```
aws kms tag-resource \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --tags TagKey='Purpose',TagValue='Test' TagKey='Dept',TagValue='IT'
```

此命令不生成任何输出。要查看 Amazon KMS 密钥上的标签，请使用 list-resource-tags 命令。

有关在 Amazon KMS 中使用标签的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的标记密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从 KMS 密钥中删除标签

以下 `untag-resource` 示例从客户托管的 KMS 密钥中删除具有 `"Purpose"` 键的标签。

要指定 KMS 密钥，请使用 `key-id` 参数。此示例使用密钥 ID 值，但您可以在此命令中使用密钥 ID 或密钥 ARN。在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。

```
aws kms untag-resource \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --tag-key 'Purpose'
```

此命令不生成任何输出。要查看 Amazon KMS 密钥上的标签，请使用 `list-resource-tags` 命令。

有关在 Amazon KMS 中使用标签的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的标记密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-alias

以下代码示例演示了如何使用 `update-alias`。

Amazon CLI

将别名与其他 KMS 密钥关联

以下 `update-alias` 示例将别名 `alias/test-key` 与其他 KMS 密钥关联起来。

`--alias-name` 参数指定别名。别名名称值必须以 `alias/` 开头。`--target-key-id` 参数指定要与别名关联的 KMS 密钥。您不需要为别名指定当前 KMS 密钥。

```
aws kms update-alias \
    --alias-name alias/test-key \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不生成任何输出。要查找别名，请使用 `list-aliases` 命令。

有关更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的更新别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAlias。

update-custom-key-store

以下代码示例演示了如何使用 update-custom-key-store。

Amazon CLI

示例 1：编辑自定义密钥存储的友好名称

以下 update-custom-key-store 示例更改自定义密钥存储的名称。此示例适用于 Amazon CloudHSM 密钥存储或外部密钥存储。

使用 custom-key-store-id 来标识密钥存储。使用 new-custom-key-store-name 参数指定新的友好名称。

要更新 Amazon CloudHSM 密钥存储的友好名称，必须先断连密钥存储，例如使用 disconnect-custom-key-store 命令来断连。在连接或断连外部密钥存储的情况下，都可以更新其友好名称。要查找自定义密钥存储的连接状态，请使用 describe-custom-key-store 命令。

```
aws kms update-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0 \
    --new-custom-key-store-name ExampleKeyStore
```

此命令不返回任何数据。要验证此命令是否有效，请使用 describe-custom-key-stores 命令。

有关更新 Amazon CloudHSM 密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑 Amazon CloudHSM 密钥存储设置。

有关更新外部密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑外部密钥存储属性。

示例 2：编辑 Amazon CloudHSM 密钥存储的 kmsuser 密码

以下 update-custom-key-store 示例将 kmsuser 密码的值更新为与指定密钥存储关联的 CloudHSM 集群中 kmsuser 的当前密码。此命令不会更改集群的 kmsuser 密码。它只是告知 Amazon KMS 当前的密码。如果 KMS 没有当前 kmsuser 密码，则无法连接到 Amazon CloudHSM 密钥存储。

注意：在更新一个 Amazon CloudHSM 密钥存储之前，必须先将其断开连接。使用 disconnect-custom-key-store 命令。命令完成后，您可以重新连接该 Amazon CloudHSM 密钥存储。使用 connect-custom-key-store 命令。

```
aws kms update-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0 \
    --key-store-password ExamplePassword
```

此命令不返回任何输出。要验证更改是否生效，请使用 describe-custom-key-stores 命令。

有关更新 Amazon CloudHSM 密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑 Amazon CloudHSM 密钥存储设置。

示例 3：编辑 Amazon CloudHSM 密钥存储的 Amazon CloudHSM 集群

以下示例将与一个 Amazon CloudHSM 密钥存储关联的 Amazon CloudHSM 集群更改为一个相关集群，例如同一集群的其他备份。

注意：在更新一个 Amazon CloudHSM 密钥存储之前，必须先将其断开连接。使用 disconnect-custom-key-store 命令。命令完成后，您可以重新连接该 Amazon CloudHSM 密钥存储。使用 connect-custom-key-store 命令。

```
aws kms update-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0 \
    --cloud-hsm-cluster-id cluster-1a23b4cdefg
```

此命令不返回任何输出。要验证更改是否生效，请使用 describe-custom-key-stores 命令。

有关更新 Amazon CloudHSM 密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑 Amazon CloudHSM 密钥存储设置。

示例 4：编辑外部密钥存储的代理身份验证凭证

以下示例更新您的外部密钥存储的代理身份验证凭证。您必须同时指定 raw-secret-access-key 和 access-key-id，即使只是更改其中一个值。您可以使用此功能修复无效凭证，或者在外部密钥存储代理轮换凭证时更改凭证。

在外部密钥存储上为 Amazon KMS 建立代理身份验证凭证。然后使用此命令向 Amazon KMS 提供凭证。AmazonKMS 使用此凭证来签署其对于您的外部密钥存储代理的请求。

在连接或断开连接外部密钥存储的情况下，都可以更新代理身份验证凭证。要查找自定义密钥存储的连接状态，请使用 describe-custom-key-store 命令。

```
aws kms update-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0 \
    --xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,
 RawSecretAccessKey=DXjSUawnel2fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```

此命令不返回任何输出。要验证更改是否生效，请使用 describe-custom-key-stores 命令。

有关更新外部密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑外部密钥存储属性。

示例 5：编辑外部密钥存储的代理连接

以下示例将外部密钥存储代理连接选项从公有端点连接更改为 VPC 端点服务连接。除了更改 xks-proxy-connectivity 值外，您还必须更改 xks-proxy-uri-endpoint 值，以反映与 VPC 端点服务关联的私有 DNS 名称。您还必须添加一个 xks-proxy-vpc-endpoint-service-name 值。

注意：在更新一个外部存储的代理连接之前，必须断开其连接。使用 disconnect-custom-key-store 命令。命令完成后，您可以使用 connect-custom-key-store 命令重新连接该外部密钥存储。

```
aws kms update-custom-key-store \
    --custom-key-store-id cks-1234567890abcdef0 \
    --xks-proxy-connectivity VPC_ENDPOINT_SERVICE \
    --xks-proxy-uri-endpoint "https://myproxy-private.xks.example.com" \
    --xks-proxy-vpc-endpoint-service-name "com.amazonaws.vpce.us-east-1.vpce-svc-
example"
```

此命令不返回任何输出。要验证更改是否生效，请使用 describe-custom-key-stores 命令。

有关更新外部密钥存储的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的编辑外部密钥存储属性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCustomKeyStore。

update-key-description

以下代码示例演示了如何使用 update-key-description。

Amazon CLI

示例 1：为客户托管的 KMS 密钥添加或更改描述

以下 update-key-description 示例为客户托管的 KMS 密钥添加描述。您可以使用同样的命令更改现有描述。

--key-id 参数在该命令中标识 KMS 密钥。此示例使用密钥 ARN 值，但您可以使用 KMS 密钥的密钥 ID 或密钥 ARN。--description 参数指定新的描述。此参数的值替换 KMS 密钥的当前描述（如有）。

```
aws kms update-key-description \
    --key-id arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --description "IT Department test key"
```

此命令不生成任何输出。要查看 KMS 密钥的描述，请使用 describe-key 命令。

有关更多信息，请参阅《Amazon 密钥管理服务 API 参考》中的 UpdateKeyDescription。

示例 2：删除客户托管的 KMS 密钥的描述

以下 update-key-description 示例删除客户托管的 KMS 密钥的描述。

--key-id 参数在该命令中标识 KMS 密钥。此示例使用密钥 ID 值，但您可以使用 KMS 密钥的密钥 ID 或密钥 ARN。采用空字符串值（''）的 --description 参数会删除现有描述。

```
aws kms update-key-description \
    --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --description ''
```

此命令不生成任何输出。要查看 KMS 密钥的描述，请使用 describe-key 命令。

有关更多信息，请参阅《Amazon 密钥管理服务 API 参考》中的 UpdateKeyDescription。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateKeyDescription。

verify-mac

以下代码示例演示了如何使用 verify-mac。

Amazon CLI

示例 1：验证 HMAC

以下 verify-mac 命令验证特定消息、HMAC KMS 密钥和 MAC 算法的 HMAC。响应的 MacValid 值中的值为“true”表示 HMAC 有效。

在 Amazon CLI v2 中，message 参数的值必须采用 Base64 编码。或者，您可以将消息保存在文件中并使用 fileb:// 前缀，它告诉 Amazon CLI 从文件中读取二进制数据。

您指定的 MAC 不得采用 base64 编码。如需解码 generate-mac 命令返回的 MAC 的帮助，请参阅 generate-mac 命令示例。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。密钥 ID 必须表示密钥用法为 GENERATE_VERIFY_MAC 的 HMAC KMS 密钥。

```
msg=(echo 'Hello World' | base64)

aws kms verify-mac \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://Message \
    --mac-algorithm HMAC_SHA_384 \
    --mac fileb://ExampleMac
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "MacValid": true,
    "MacAlgorithm": "HMAC_SHA_384"
}
```

有关在 Amazon KMS 中使用 HMAC KMS 密钥的更多信息，请参阅《Amazon Key Management Service Developer Guide》中的 HMAC keys in Amazon KMS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyMac。

verify

以下代码示例演示了如何使用 verify。

Amazon CLI

验证数字签名

以下 verify 命令验证一条 Base64 编码短消息的加密签名。密钥 ID、消息、消息类型和签名算法必须与用于签名该消息的相同。

在 Amazon CLI v2 中，message 参数的值必须采用 Base64 编码。或者，您可以将消息保存在文件中并使用 fileb:// 前缀，它告诉 Amazon CLI 从文件中读取二进制数据。

您指定的签名不得采用 base64 编码。如需解码 sign 命令返回的签名的帮助，请参阅 sign 命令示例。

该命令的输出包括一个布尔 SignatureValid 字段，表示签名已通过验证。如果签名验证失败，verify 命令也会失败。

在运行此命令之前，请将例子中的密钥 ID 替换为您的 Amazon 账户中的有效密钥 ID。

```
aws kms verify \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://EncodedMessage \
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
    --signature fileb://ExampleSignature
```

输出：

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "SignatureValid": true,
    "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

有关在 Amazon KMS 中使用非对称 KMS 密钥的更多信息，请参阅《Amazon 密钥管理服务开发人员指南》中的使用非对称密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Verify。

使用 Amazon CLI 的 Lake Formation 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Lake Formation 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-lf-tags-to-resource

以下代码示例演示了如何使用 add-lf-tags-to-resource。

Amazon CLI

将一个或多个 LF 标签附加到现有资源

以下 add-lf-tags-to-resource 示例将给定 LF 标签附加到表资源。

```
aws lakeformation add-lf-tags-to-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Resource": {
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "Name": "dl_tpc_promotion"
        }
    },
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "analyst"
        ]
    }]
}
```

输出：

```
{
    "Failures": []
```

```
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的为数据目录资源分配 LF 标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddLfTagsToResource。

batch-grant-permissions

以下代码示例演示了如何使用 batch-grant-permissions。

Amazon CLI

向主体批量授予资源权限

以下 batch-grant-permissions 示例批量向主体授予对指定资源的访问权限。

```
aws lakeformation batch-grant-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Entries": [{
            "Id": "1",
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "Name": "dl_tpc_promotion"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": [
                "ALL"
            ]
```

```
        },
        {
            "Id": "2",
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "Name": "dl_tpc_customer"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": [
                "ALL"
            ]
        },
        {
            "Id": "3",
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
business-analyst"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "Name": "dl_tpc_promotion"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": [
                "ALL"
            ]
        },
        {
            "Id": "4",
            "Principal": {
```

```
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
            },
            "Resource": {
                "DataCellsFilter": {
                    "TableCatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "TableName": "dl_tpc_item",
                    "Name": "developer_item"
                }
            },
            "Permissions": [
                "SELECT"
            ],
            "PermissionsWithGrantOption": []
        }
    ]
}
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGrantPermissions。

batch-revoke-permissions

以下代码示例演示了如何使用 batch-revoke-permissions。

Amazon CLI

批量撤销主体对资源的权限

以下 batch-revoke-permissions 示例批量撤销主体对指定资源的访问权限。

```
aws lakeformation batch-revoke-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Entries": [{
            "Id": "1",
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "Name": "dl_tpc_promotion"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": [
                "ALL"
            ]
        },
        {
            "Id": "2",
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
business-analyst"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "tpc",
                    "Name": "dl_tpc_promotion"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": [
                "ALL"
            ]
        }
```

```
    ]
}
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchRevokePermissions。

`cancel-transaction`

以下代码示例演示了如何使用 `cancel-transaction`。

Amazon CLI

取消事务

以下 `cancel-transaction` 示例取消事务。

```
aws lakeformation cancel-transaction \
    --transaction-id='b014d972ca8347b89825e33c5774aec4'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelTransaction。

`commit-transaction`

以下代码示例演示了如何使用 `commit-transaction`。

Amazon CLI

提交事务

以下 `commit-transaction` 示例提交事务。

```
aws lakeformation commit-transaction \
    --transaction-id='b014d972ca8347b89825e33c5774aec4'
```

输出：

```
{
    "TransactionStatus": "committed"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CommitTransaction。

create-data-cells-filter

以下代码示例演示了如何使用 create-data-cells-filter。

Amazon CLI

示例 1：创建数据单元格筛选器

以下 create-data-cells-filter 示例创建一个数据单元格筛选器，以允许用户根据行条件授予对某些列的访问权限。

```
aws lakeformation create-data-cells-filter \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "TableData": {
        "ColumnNames": ["p_channel_details", "p_start_date_sk", "p_promo_name"],
        "DatabaseName": "tpc",
        "Name": "developer_promotion",
        "RowFilter": {
            "FilterExpression": "p_promo_name='ese'"
        },
        "TableCatalogId": "123456789111",
        "TableName": "dl_tpc_promotion"
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的 Lake Formation 中的数据筛选和单元格级安全性。

示例 2：创建列筛选器

以下 create-data-cells-filter 示例创建一个数据筛选器，以允许用户授予对某些列的访问权限。

```
aws lakeformation create-data-cells-filter \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "TableData": {
        "ColumnNames": ["p_channel_details", "p_start_date_sk", "p_promo_name"],
        "DatabaseName": "tpc",
        "Name": "developer_promotion_allrows",
        "RowFilter": {
            "AllRowsWildcard": {}
        },
        "TableCatalogId": "123456789111",
        "TableName": "dl_tpc_promotion"
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的 Lake Formation 中的数据筛选和单元格级安全性。

示例 3：使用排除列创建数据筛选器

以下 create-data-cells-filter 示例创建一个数据筛选器，以允许用户授予对除了提到的列之外的所有列的访问权限。

```
aws lakeformation create-data-cells-filter \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "TableData": {
        "ColumnWildcard": {
            "ExcludedColumnNames": ["p_channel_details", "p_start_date_sk"]
        },
        "DatabaseName": "tpc",
        "Name": "developer_promotion_excludecolumn",
        "RowFilter": {
            "AllRowsWildcard": {}
        },
        "TableCatalogId": "123456789111",
        "TableName": "dl_tpc_promotion"
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的 Lake Formation 中的数据筛选和单元格级安全性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataCellsFilter。

`create-lf-tag`

以下代码示例演示了如何使用 `create-lf-tag`。

Amazon CLI

创建 LF 标签

以下 `create-lf-tag` 示例使用指定名称和值创建一个 LF 标签。

```
aws lakeformation create-lf-tag \
    --catalog-id '123456789111' \
    --tag-key 'usergroup' \
    --tag-values '["developer","analyst","campaign"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 LF 标签以进行元数据访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLfTag。

delete-data-cells-filter

以下代码示例演示了如何使用 delete-data-cells-filter。

Amazon CLI

删除数据单元格筛选器

以下 delete-data-cells-filter 示例删除给定的数据单元格筛选器。

```
aws lakeformation delete-data-cells-filter \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "TableCatalogId": "123456789111",
    "DatabaseName": "tpc",
    "TableName": "dl_tpc_promotion",
    "Name": "developer_promotion"
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的 Lake Formation 中的数据筛选和单元格级安全性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDataCellsFilter。

delete-lf-tag

以下代码示例演示了如何使用 delete-lf-tag。

Amazon CLI

删除 LF 标签定义

以下 delete-lf-tag 示例删除 LF 标签定义。

```
aws lakeformation delete-lf-tag \
```

```
    --catalog-id '123456789111' \
    --tag-key 'usergroup'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 LF 标签以进行元数据访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLfTag。

delete-objects-on-cancel

以下代码示例演示了如何使用 delete-objects-on-cancel。

Amazon CLI

取消事务时删除对象

以下 delete-objects-on-cancel 示例在取消事务时删除列出的 s3 对象。

```
aws lakeformation delete-objects-on-cancel \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "012345678901",
    "DatabaseName": "tpc",
    "TableName": "dl_tpc_household_demographics_gov",
    "TransactionId": "1234d972ca8347b89825e33c5774aec4",
    "Objects": [{
        "Uri": "s3://lf-data-lake-012345678901/target/
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-
ff26b17504414fe88b302cd795eabd00.parquet",
        "ETag": "1234ab1fc50a316b149b4e1f21a73800"
    }]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObjectsOnCancel。

deregister-resource

以下代码示例演示了如何使用 deregister-resource。

Amazon CLI

注销数据湖存储

以下 deregister-resource 示例注销由 Lake Formation 托管的资源。

```
aws lakeformation deregister-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123"
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的向数据湖添加 Amazon S3 位置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterResource。

describe-transaction

以下代码示例演示了如何使用 describe-transaction。

Amazon CLI

检索事务详细信息

以下 describe-transaction 示例返回单个事务的详细信息。

```
aws lakeformation describe-transaction \
    --transaction-id='8cb4b1a7cc8d486fbaca9a64e7d9f5ce'
```

输出：

```
{
    "TransactionDescription": {
        "TransactionId": "12345972ca8347b89825e33c5774aec4",
        "TransactionStatus": "committed",
        "TransactionStartTime": "2022-08-10T14:29:04.046000+00:00",
        "TransactionEndTime": "2022-08-10T14:29:09.681000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTransaction。

extend-transaction

以下代码示例演示了如何使用 extend-transaction。

Amazon CLI

扩展事务

以下 extend-transaction 示例扩展事务。

```
aws lakeformation extend-transaction \
    --transaction-id='8cb4b1a7cc8d486fbaca9a64e7d9f5ce'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExtendTransaction。

get-data-lake-settings

以下代码示例演示了如何使用 get-data-lake-settings。

Amazon CLI

检索 Amazon Lake Formation 托管的数据湖设置

以下 get-data-lake-settings 示例检索数据湖管理员列表和其他数据湖设置。

```
aws lakeformation get-data-lake-settings \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111"
}
```

输出：

```
{
    "DataLakeSettings": {
        "DataLakeAdmins": [{
            "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-admin"
        }],
        "CreateDatabaseDefaultPermissions": [],
        "CreateTableDefaultPermissions": [
                {
                        "Principal": {
                            "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                        },
                        "Permissions": [
                            "ALL"
                        ]
                }
        ],
        "TrustedResourceOwners": [],
        "AllowExternalDataFiltering": true,
        "ExternalDataFilteringAllowList": [{
            "DataLakePrincipalIdentifier": "123456789111"
        }],
        "AuthorizedSessionTagValueList": [
            "Amazon EMR"
        ]
    }
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的更改数据湖的默认安全设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataLakeSettings。

get-effective-permissions-for-path

以下代码示例演示了如何使用 get-effective-permissions-for-path。

Amazon CLI

检索位于特定路径的资源的权限

以下 get-effective-permissions-for-path 示例返回位于 Amazon S3 中某个路径上的指定表或数据库资源的 Lake Formation 权限。

```
aws lakeformation get-effective-permissions-for-path \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111"
}
```

输出：

```
{
    "Permissions": [{
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
campaign-manager"
            },
            "Resource": {
                "Database": {
                    "Name": "tpc"
                }
            },
            "Permissions": [
                "DESCRIBE"
            ],
            "PermissionsWithGrantOption": []
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/EMR-
RuntimeRole"
```

```
            },
            "Resource": {
                "Database": {
                    "Name": "tpc"
                }
            },
            "Permissions": [
                "ALL"
            ],
            "PermissionsWithGrantOption": []
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:saml-
provider/oktaSAMLProvider:user/emr-developer"
            },
            "Resource": {
                "Database": {
                    "Name": "tpc"
                }
            },
            "Permissions": [
                "ALL",
                "DESCRIBE"
            ],
            "PermissionsWithGrantOption": []
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
            },
            "Resource": {
                "Database": {
                    "Name": "tpc"
                }
            },
            "Permissions": [
                "ALL",
                "ALTER",
                "CREATE_TABLE",
                "DESCRIBE",
                "DROP"
            ],
```

```
            "PermissionsWithGrantOption": [
                "ALL",
                "ALTER",
                "CREATE_TABLE",
                "DESCRIBE",
                "DROP"
            ]
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/LF-
GlueServiceRole"
            },
            "Resource": {
                "Database": {
                    "Name": "tpc"
                }
            },
            "Permissions": [
                "CREATE_TABLE"
            ],
            "PermissionsWithGrantOption": []
        }
    ],
    "NextToken":
 "E5SlJDSTZleUp6SWpvaU9UQTNORE0zTXpFeE5Ua3pJbjE5TENKbGVIQnBjbUYwYVc5dUlqcDdJbk5sWTI5dVpITWlP
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 Lake Formation 权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEffectivePermissionsForPath。

get-lf-tag

以下代码示例演示了如何使用 get-lf-tag。

Amazon CLI

检索 LF 标签定义

以下 get-lf-tag 示例检索 LF 标签定义。

```
aws lakeformation get-lf-tag \
    --catalog-id '123456789111' \
    --tag-key 'usergroup'
```

输出：

```
{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
        "analyst",
        "campaign",
        "developer"
    ]
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 LF 标签以进行元数据访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLfTag。

get-query-state

以下代码示例演示了如何使用 get-query-state。

Amazon CLI

检索已提交查询的状态

以下 get-query-state 示例返回先前提交的查询的状态。

```
aws lakeformation get-query-state \
    --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

输出：

```
{
    "State": "FINISHED"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的事务性数据操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueryState。

get-query-statistics

以下代码示例演示了如何使用 get-query-statistics。

Amazon CLI

检索查询统计信息

以下 get-query-statistics 示例检索有关查询计划和执行的统计信息。

```
aws lakeformation get-query-statistics \
    --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

输出：

```
{
    "ExecutionStatistics": {
        "AverageExecutionTimeMillis": 0,
        "DataScannedBytes": 0,
        "WorkUnitsExecutedCount": 0
    },
    "PlanningStatistics": {
        "EstimatedDataToScanBytes": 43235,
        "PlanningTimeMillis": 2377,
        "QueueTimeMillis": 440,
        "WorkUnitsGeneratedCount": 1
    },
    "QuerySubmissionTime": "2022-08-11T02:14:38.641870+00:00"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的事务性数据操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueryStatistics。

get-resource-lf-tags

以下代码示例演示了如何使用 get-resource-lf-tags。

Amazon CLI

列出 LF 标签

以下 `list-lf-tags` 示例返回请求者有权查看的 LF 标签列表。

```
aws lakeformation list-lf-tags \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "ResourceShareType": "ALL",
    "MaxResults": 2
}
```

输出：

```
{
"LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "category",
        "TagValues": [
            "private",
            "public"
        ]
    },
    {
        "CatalogId": "123456789111",
        "TagKey": "group",
        "TagValues": [
            "analyst",
            "campaign",
            "developer"
        ]
    }],
    "NextToken": "kIiwiZXhwaXJhdGlvbiI6eyJzZWNvbmRzIjoxNjYwMDY4dCI6ZmFsc2V9"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 LF 标签以进行元数据访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceLfTags。

get-table-objects

以下代码示例演示了如何使用 get-table-objects。

Amazon CLI

列出受控表格的对象

以下 get-table-objects 示例返回构成指定受控表格的一组 Amazon S3 对象。

```
aws lakeformation get-table-objects \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "012345678901",
    "DatabaseName": "tpc",
    "TableName": "dl_tpc_household_demographics_gov",
    "QueryAsOfTime": "2022-08-10T15:00:00"
}
```

输出：

```
{
    "Objects": [{
        "PartitionValues": [],
        "Objects": [{
            "Uri": "s3://lf-data-lake-012345678901/target/
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-
ff26b17504414fe88b302cd795eabd00.parquet",
            "ETag": "12345b1fc50a316b149b4e1f21a73800",
            "Size": 43235
        }]
    }]
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTableObjects。

get-work-unit-results

以下代码示例演示了如何使用 get-work-unit-results。

Amazon CLI

检索给定查询的工作单位

以下 get-work-unit-results 示例返回从查询得到的工作单位。

```
aws lakeformation get-work-units \
    --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b' \
    --work-unit-id '0' \
    --work-unit-token 'B2fMSdmQXe9umX8Ux8XCo4=' outfile
```

输出：

```
outfile with Blob content.
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的事务性数据操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWorkUnitResults。

get-work-units

以下代码示例演示了如何使用 get-work-units。

Amazon CLI

检索工作单位

以下 get-work-units 示例检索 StartQueryPlanning 操作生成的工作单位。

```
aws lakeformation get-work-units \
    --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

输出：

```
{
```

```
    "WorkUnitRanges": [{
        "WorkUnitIdMax": 0,
        "WorkUnitIdMin": 0,
        "WorkUnitToken":
 "1234eMAk4kLO4umqEL4Z5WuxL04AXwABABVhd3MtY3J5cHRvLXB1YmxpYy1rZXkAREEwYm9QbkhINmFYTWphbmMxZW
+f88jzGrYq22gE6jkQlpOB
+Oet2eqNUmFudAAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMCOEWRda
wAAAAEAAAAAAAAAAAAAAAEAAACX3/w5h75QAPomfKH+cyEKYU1yccUmBl
+VSojiGOtdsUk7vcjYXUUboYm3dvqRqX2s4gROMOn
+Ij8R0/8jYmnHkpvyAFNVRPyETyIKg7k5Z9+5I1c2d3446Jw/moWGGxjH8AEG9h27ytmOhozxDOEi/
F2ZoXz6wlGDfGUo/2WxCkYOhTyNaw6TM
+7drTM7yrW4iNVLUM0LX0xnFjIAhLhooWJek6vjQZUAZzBlAjBH8okRtYP8R7AY2Wls/
hqFBhG0V4l42AC0LxsuZbMQrE2SzWZUZ0E9Uew7/n0cyX4CMQDR79INyv4ysMByW9kKGGKyba+cCNklExMR
+btBQBmMuB2fMSdmQXe9umX8Ux8XCo4="
    }],
    "QueryId": "1234273f-4a62-4cda-8d98-69615ee8be9b"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的事务性数据操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWorkUnits。

grant-permissions

以下代码示例演示了如何使用 grant-permissions。

Amazon CLI

示例 1：向主体授予使用 LF 标签的资源的权限

以下 grant-permissions 示例向主体授予与 LF 标签策略匹配的数据库资源的所有权限。

```
aws lakeformation grant-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-admin"
    },
```

```
    "Resource": {
        "LFTagPolicy": {
            "CatalogId": "123456789111",
            "ResourceType": "DATABASE",
            "Expression": [{
                "TagKey": "usergroup",
                "TagValues": [
                    "analyst",
                    "developer"
                ]
            }]
        }
    },
    "Permissions": [
        "ALL"
    ],
    "PermissionsWithGrantOption": [
        "ALL"
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

示例 2：向主体授予列级权限

以下 `grant-permissions` 示例向主体授予选择的特定列的权限。

```
aws lakeformation grant-permissions \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
    },
    "Resource": {
        "TableWithColumns": {
```

```
            "CatalogId": "123456789111",
            "ColumnNames": ["p_end_date_sk"],
            "DatabaseName": "tpc",
            "Name": "dl_tpc_promotion"
        }
    },
    "Permissions": [
        "SELECT"
    ],
    "PermissionsWithGrantOption": []
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

示例 3：向主体授予表权限

以下 grant-permissions 示例向主体授予对给定数据库中所有表的选择权限。

```
aws lakeformation grant-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
    },
    "Resource": {
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "TableWildcard": {}
        }
    },
    "Permissions": [
        "SELECT"
    ],
    "PermissionsWithGrantOption": []
```

```
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

示例 4：向主体授予 LF 标签权限

以下 `grant-permissions` 示例向主体授予 LF 标签的关联权限。

```
aws lakeformation grant-permissions \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
    },
    "Resource": {
        "LFTag": {
            "CatalogId": "123456789111",
            "TagKey": "category",
            "TagValues": [
                "private", "public"
            ]
        }

    },
    "Permissions": [
        "ASSOCIATE"
    ],
    "PermissionsWithGrantOption": []
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

示例 5：向主体授予数据位置权限

以下 grant-permissions 示例向主体授予数据位置的权限。

```
aws lakeformation grant-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
    },
    "Resource": {
        "DataLocation": {
            "CatalogId": "123456789111",
            "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111"
        }
    },
    "Permissions": [
        "DATA_LOCATION_ACCESS"
    ],
    "PermissionsWithGrantOption": []
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GrantPermissions。

list-data-cells-filter

以下代码示例演示了如何使用 list-data-cells-filter。

Amazon CLI

列出数据单元格筛选器

以下 list-data-cells-filter 示例列出给定表的数据单元格筛选器。

```
aws lakeformation list-data-cells-filter \
```

```
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "MaxResults": 2,
    "Table": {
        "CatalogId": "123456789111",
        "DatabaseName": "tpc",
        "Name": "dl_tpc_promotion"
    }
}
```

输出：

```
{
    "DataCellsFilters": [{
            "TableCatalogId": "123456789111",
            "DatabaseName": "tpc",
            "TableName": "dl_tpc_promotion",
            "Name": "developer_promotion",
            "RowFilter": {
                "FilterExpression": "p_promo_name='ese'"
            },
            "ColumnNames": [
                "p_channel_details",
                "p_start_date_sk",
                "p_purpose",
                "p_promo_id",
                "p_promo_name",
                "p_end_date_sk",
                "p_discount_active"
            ]
        },
        {
            "TableCatalogId": "123456789111",
            "DatabaseName": "tpc",
            "TableName": "dl_tpc_promotion",
            "Name": "developer_promotion_allrows",
            "RowFilter": {
                "FilterExpression": "TRUE",
                "AllRowsWildcard": {}
            },
```

```
            "ColumnNames": [
                "p_channel_details",
                "p_start_date_sk",
                "p_promo_name"
            ]
        }
    ],
    "NextToken": "2MDA2MTgwNiwibmFub3MiOjE0MDAwMDAwMH19"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的 Lake Formation 中的数据筛选和单元格级安全性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDataCellsFilter。

list-permissions

以下代码示例演示了如何使用 list-permissions。

Amazon CLI

示例 1：检索资源的主体权限列表

以下 list-permissions 示例返回数据库资源的主体权限列表。

```
aws lakeformation list-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "ResourceType": "DATABASE",
    "MaxResults": 2
}
```

输出：

```
{
    "PrincipalResourcePermissions": [{
        "Principal": {
```

```
            "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
campaign-manager"
        },
        "Resource": {
            "Database": {
                "CatalogId": "123456789111",
                "Name": "tpc"
            }
        },
        "Permissions": [
            "DESCRIBE"
        ],
        "PermissionsWithGrantOption": []
    }],
    "NextToken":
 "E5SlJDSTZleUp6SWpvaU9UQTNORE0zTXpFeE5Ua3pJbjE5TENKbGVIQnBjbUYwYVc5dUlqcDdJbk5sWTI5dVpITWlP
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 Lake Formation 权限。

示例 2：检索具有数据筛选器的表的主体权限列表

以下 `list-permissions` 示例列出向主体授予的具有相关数据筛选器的表的权限。

```
aws lakeformation list-permissions \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "Resource": {
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "Name": "dl_tpc_customer"
        }
    },
    "IncludeRelated": "TRUE",
    "MaxResults": 10
}
```

输出：

```
{
    "PrincipalResourcePermissions": [{
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
            },
            "Resource": {
                "Table": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "customer",
                    "Name": "customer_invoice"
                }
            },
            "Permissions": [
                "ALL",
                "ALTER",
                "DELETE",
                "DESCRIBE",
                "DROP",
                "INSERT"
            ],
            "PermissionsWithGrantOption": [
                "ALL",
                "ALTER",
                "DELETE",
                "DESCRIBE",
                "DROP",
                "INSERT"
            ]
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
            },
            "Resource": {
                "TableWithColumns": {
                    "CatalogId": "123456789111",
                    "DatabaseName": "customer",
                    "Name": "customer_invoice",
                    "ColumnWildcard": {}
                }
```

```
            },
            "Permissions": [
                "SELECT"
            ],
            "PermissionsWithGrantOption": [
                "SELECT"
            ]
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
            },
            "Resource": {
                "DataCellsFilter": {
                    "TableCatalogId": "123456789111",
                    "DatabaseName": "customer",
                    "TableName": "customer_invoice",
                    "Name": "dl_us_customer"
                }
            },
            "Permissions": [
                "DESCRIBE",
                "SELECT",
                "DROP"
            ],
            "PermissionsWithGrantOption": []
        }
    ],
    "NextToken": "VyeUFjY291bnRQZXJtaXNzaW9ucyI6ZmFsc2V9"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 Lake Formation 权限。

示例 3：检索 LF 标签的主体权限列表

以下 `list-permissions` 示例列出授予主体的 LF 标签权限。

```
aws lakeformation list-permissions \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "Resource": {
        "LFTag": {
            "CatalogId": "123456789111",
            "TagKey": "category",
            "TagValues": [
                "private"
            ]
        }
    },
    "MaxResults": 10
}
```

输出：

```
{
    "PrincipalResourcePermissions": [{
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
            },
            "Resource": {
                "LFTag": {
                    "CatalogId": "123456789111",
                    "TagKey": "category",
                    "TagValues": [
                        "*"
                    ]
                }
            },
            "Permissions": [
                "DESCRIBE"
            ],
            "PermissionsWithGrantOption": [
                "DESCRIBE"
            ]
        },
        {
            "Principal": {
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
            },
```

```
            "Resource": {
                "LFTag": {
                    "CatalogId": "123456789111",
                    "TagKey": "category",
                    "TagValues": [
                        "*"
                    ]
                }
            },
            "Permissions": [
                "ASSOCIATE"
            ],
            "PermissionsWithGrantOption": [
                "ASSOCIATE"
            ]
        }
    ],
    "NextToken": "EJwY21GMGFXOXVJanA3SW5Ocm1pc3Npb25zIjpmYWxzZX0="
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 Lake Formation 权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPermissions。

list-resources

以下代码示例演示了如何使用 list-resources。

Amazon CLI

列出 Lake Formation 托管的资源

以下 list-resources 示例列出 Lake Formation 托管的并与条件相匹配的资源。

```
aws lakeformation list-resources \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "FilterConditionList": [{
        "Field": "ROLE_ARN",
```

```
        "ComparisonOperator": "CONTAINS",
        "StringValueList": [
            "123456789111"
        ]
    }],
    "MaxResults": 10
}
```

输出：

```
{
    "ResourceInfoList": [{
            "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111",
            "RoleArn": "arn:aws:iam::123456789111:role/LF-GlueServiceRole",
            "LastModified": "2022-07-21T02:12:46.669000+00:00"
        },
        {
            "ResourceArn": "arn:aws:s3:::lf-emr-test-123456789111",
            "RoleArn": "arn:aws:iam::123456789111:role/EMRLFS3Role",
            "LastModified": "2022-07-29T16:22:03.211000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 Lake Formation 权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResources。

`list-transactions`

以下代码示例演示了如何使用 `list-transactions`。

Amazon CLI

列出所有事务详细信息

以下 `list-transactions` 示例返回有关事务的元数据及其状态。

```
aws lakeformation list-transactions \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "StatusFilter": "ALL",
    "MaxResults": 3
}
```

输出：

```
{
    "Transactions": [{
            "TransactionId": "1234569f08804cb790d950d4d0fe485e",
            "TransactionStatus": "committed",
            "TransactionStartTime": "2022-08-10T14:32:29.220000+00:00",
            "TransactionEndTime": "2022-08-10T14:32:33.751000+00:00"
        },
        {
            "TransactionId": "12345972ca8347b89825e33c5774aec4",
            "TransactionStatus": "committed",
            "TransactionStartTime": "2022-08-10T14:29:04.046000+00:00",
            "TransactionEndTime": "2022-08-10T14:29:09.681000+00:00"
        },
        {
            "TransactionId": "12345daf6cb047dbba8ad9b0414613b2",
            "TransactionStatus": "committed",
            "TransactionStartTime": "2022-08-10T13:56:51.261000+00:00",
            "TransactionEndTime": "2022-08-10T13:56:51.547000+00:00"
        }
    ],
    "NextToken": "77X1ebypsI7os+X2lhHsZLGNCDK3nNGpwRdFpicSOHgcX1/
QMoniUAKcpR3kj3ts3PVdMA=="
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTransactions。

put-data-lake-settings

以下代码示例演示了如何使用 put-data-lake-settings。

Amazon CLI

设置 Amazon Lake Formation 托管的数据湖设置

以下 put-data-lake-settings 示例设置数据湖管理员列表和其他数据湖设置。

```
aws lakeformation put-data-lake-settings \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "DataLakeSettings": {
        "DataLakeAdmins": [{
                "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
            }
        ],
        "CreateDatabaseDefaultPermissions": [],
        "CreateTableDefaultPermissions": [],
        "TrustedResourceOwners": [],
        "AllowExternalDataFiltering": true,
        "ExternalDataFilteringAllowList": [{
            "DataLakePrincipalIdentifier ": "123456789111"
        }],
        "AuthorizedSessionTagValueList": ["Amazon EMR"]
    }
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的更改数据湖的默认安全设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutDataLakeSettings。

register-resource

以下代码示例演示了如何使用 register-resource。

Amazon CLI

示例 1：使用服务关联角色注册数据湖存储

以下 register-resource 示例使用服务关联角色注册由 Lake Formation 托管的资源。

```
aws lakeformation register-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123",
    "UseServiceLinkedRole": true
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的向数据湖添加 Amazon S3 位置。

示例 2：使用自定义角色注册数据湖存储

以下 register-resource 示例使用自定义角色注册由 Lake Formation 托管的资源。

```
aws lakeformation register-resource \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123",
    "UseServiceLinkedRole": false,
    "RoleArn": "arn:aws:iam::123456789111:role/LF-GlueServiceRole"
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的向数据湖添加 Amazon S3 位置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterResource。

remove-lf-tags-from-resource

以下代码示例演示了如何使用 remove-lf-tags-from-resource。

Amazon CLI

从资源中移除 LF 标签

以下 `remove-lf-tags-from-resource` 示例移除与表资源关联的 LF 标签。

```
aws lakeformation remove-lf-tags-from-resource \
    --cli-input-json file://input.json
```

的内容`input.json`：

```
{
    "CatalogId": "123456789111",
    "Resource": {
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "Name": "dl_tpc_promotion"
        }
    },
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
}
```

输出：

```
{
    "Failures": []
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的为数据目录资源分配 LF 标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveLfTagsFromResource。

revoke-permissions

以下代码示例演示了如何使用 revoke-permissions。

Amazon CLI

撤销主体对资源的权限

以下 revoke-permissions 示例撤销主体对于给定数据库的特定表的访问权限。

```
aws lakeformation revoke-permissions \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "123456789111",
    "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
    },
    "Resource": {
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "Name": "dl_tpc_promotion"
        }
    },
    "Permissions": [
        "ALL"
    ],
    "PermissionsWithGrantOption": []
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的授予和撤销对数据目录资源的权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokePermissions。

search-databases-by-lf-tags

以下代码示例演示了如何使用 search-databases-by-lf-tags。

Amazon CLI

通过 LF 标签搜索数据库资源

以下 search-databases-by-lf-tags 示例搜索与 LF 标签表达式匹配的数据库资源。

```
aws lakeformation search-databases-by-lf-tags \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "MaxResults": 1,
    "CatalogId": "123456789111",
    "Expression": [{
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
}
```

输出：

```
{
    "DatabaseList": [{
        "Database": {
            "CatalogId": "123456789111",
            "Name": "tpc"
        },
        "LFTags": [{
            "CatalogId": "123456789111",
            "TagKey": "usergroup",
            "TagValues": [
                "developer"
            ]
        }]
    }]
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的查看分配得到某个 LF 标签的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchDatabasesByLfTags。

search-tables-by-lf-tags

以下代码示例演示了如何使用 search-tables-by-lf-tags。

Amazon CLI

通过 LF 标签搜索表资源

以下 search-tables-by-lf-tags 示例搜索与 LF 标签表达式匹配的表资源。

```
aws lakeformation search-tables-by-lf-tags \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "MaxResults": 2,
    "CatalogId": "123456789111",
    "Expression": [{
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
}
```

输出：

```
{
    "NextToken": "c2VhcmNoQWxsVGFnc0luVGFibGVzIjpmYWxzZX0=",
    "TableList": [{
        "Table": {
            "CatalogId": "123456789111",
            "DatabaseName": "tpc",
            "Name": "dl_tpc_item"
        },
        "LFTagOnDatabase": [{
            "CatalogId": "123456789111",
            "TagKey": "usergroup",
            "TagValues": [
```

```
                "developer"
            ]
        }],
        "LFTagsOnTable": [{
            "CatalogId": "123456789111",
            "TagKey": "usergroup",
            "TagValues": [
                "developer"
            ]
        }],
        "LFTagsOnColumns": [{
                "Name": "i_item_desc",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_container",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_wholesale_cost",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_manufact_id",
                "LFTags": [{
                    "CatalogId": "123456789111",
```

```
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_brand_id",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_formulation",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_current_price",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_size",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
```

```
            }]
        },
        {
            "Name": "i_rec_start_date",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_manufact",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_item_sk",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_manager_id",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_item_id",
```

```
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_class_id",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_class",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_category",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            },
            {
                "Name": "i_category_id",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
```

```
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_brand",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_units",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_rec_end_date",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
        {
            "Name": "i_color",
            "LFTags": [{
                "CatalogId": "123456789111",
                "TagKey": "usergroup",
                "TagValues": [
                    "developer"
                ]
            }]
        },
```

```
            {
                "Name": "i_product_name",
                "LFTags": [{
                    "CatalogId": "123456789111",
                    "TagKey": "usergroup",
                    "TagValues": [
                        "developer"
                    ]
                }]
            }
        ]
    }]
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的查看分配得到某个 LF 标签的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchTablesByLfTags。

start-query-planning

以下代码示例演示了如何使用 start-query-planning。

Amazon CLI

处理查询语句

以下 start-query-planning 示例提交了处理查询语句的请求。

```
aws lakeformation start-query-planning \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "QueryPlanningContext": {
        "CatalogId": "012345678901",
        "DatabaseName": "tpc"
    },
    "QueryString": "select * from dl_tpc_household_demographics_gov where
 hd_income_band_sk=9"
}
```

输出：

```
{
    "QueryId": "772a273f-4a62-4cda-8d98-69615ee8be9b"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartQueryPlanning。

start-transaction

以下代码示例演示了如何使用 start-transaction。

Amazon CLI

启动新事务

以下 start-transaction 示例启动一个新事务并返回其事务 ID。

```
aws lakeformation start-transaction \
    --transaction-type = 'READ_AND_WRITE'
```

输出：

```
{
    "TransactionId": "b014d972ca8347b89825e33c5774aec4"
}
```

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartTransaction。

update-lf-tag

以下代码示例演示了如何使用 update-lf-tag。

Amazon CLI

更新 LF 标签定义

以下 update-lf-tag 示例更新 LF 标签定义。

```
aws lakeformation update-lf-tag \
    --catalog-id '123456789111' \
    --tag-key 'usergroup' \
    --tag-values-to-add '["admin"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的管理 LF 标签以进行元数据访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLfTag。

update-table-objects

以下代码示例演示了如何使用 update-table-objects。

Amazon CLI

修改受控表格的对象

以下 update-table-objects 示例将提供的 S3 对象添加到指定的受控表中。

```
aws lakeformation update-table-objects \
    --cli-input-json file://input.json
```

的内容input.json：

```
{
    "CatalogId": "012345678901",
    "DatabaseName": "tpc",
    "TableName": "dl_tpc_household_demographics_gov",
    "TransactionId": "12347a9f75424b9b915f6ff201d2a190",
    "WriteOperations": [{
        "AddObject": {
            "Uri": "s3://lf-data-lake-012345678901/target/
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-
ff26b17504414fe88b302cd795eabd00.parquet",
            "ETag": "1234ab1fc50a316b149b4e1f21a73800",
            "Size": 42200
        }
    }]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lake Formation 开发人员指南》中的在事务内读写数据湖。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTableObjects。

使用 Amazon CLI 的 Lambda 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Lambda 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-layer-version-permission`

以下代码示例演示了如何使用 `add-layer-version-permission`。

Amazon CLI

向层版本添加权限

以下 `add-layer-version-permission` 示例授权指定账户使用层 `my-layer` 的版本 1。

```
aws lambda add-layer-version-permission \
    --layer-name my-layer \
    --statement-id xaccount \
    --action lambda:GetLayerVersion  \
    --principal 123456789012 \
    --version-number 1
```

输出：

```
{
    "RevisionId": "35d87451-f796-4a3f-a618-95a3671b0a0c",
    "Statement":
    {
        "Sid":"xaccount",
        "Effect":"Allow",
        "Principal":{
            "AWS":"arn:aws:iam::210987654321:root"
        },
        "Action":"lambda:GetLayerVersion",
        "Resource":"arn:aws:lambda:us-east-2:123456789012:layer:my-layer:1"
    }
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddLayerVersionPermission。

add-permission

以下代码示例演示了如何使用 add-permission。

Amazon CLI

向现有 Lambda 函数添加权限

以下 add-permission 示例授权 Amazon SNS 服务调用名为 my-function 的函数。

```
aws lambda add-permission \
    --function-name my-function \
    --action lambda:InvokeFunction \
    --statement-id sns \
    --principal sns.amazonaws.com
```

输出：

```
{
    "Statement":
    {
        "Sid":"sns",
        "Effect":"Allow",
        "Principal":{
```

```
            "Service":"sns.amazonaws.com"
        },
        "Action":"lambda:InvokeFunction",
        "Resource":"arn:aws:lambda:us-east-2:123456789012:function:my-function"
    }
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的对 Amazon Lambda 使用基于资源的策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddPermission。

`create-alias`

以下代码示例演示了如何使用 `create-alias`。

Amazon CLI

为 Lambda 函数创建别名

以下 `create-alias` 示例会创建名为 `LIVE` 的别名，该别名指向 `my-function` Lambda 函数的版本 1。

```
aws lambda create-alias \
    --function-name my-function \
    --description "alias for live version of function" \
    --function-version 1 \
    --name LIVE
```

输出：

```
{
    "FunctionVersion": "1",
    "Name": "LIVE",
    "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",
    "RevisionId": "873282ed-4cd3-4dc8-a069-d0c647e470c6",
    "Description": "alias for live version of function"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAlias。

create-event-source-mapping

以下代码示例演示了如何使用 create-event-source-mapping。

Amazon CLI

创建事件源与 Amazon Lambda 函数之间的映射

以下 create-event-source-mapping 示例创建 SQS 队列和 my-function Lambda 函数之间的映射。

```
aws lambda create-event-source-mapping \
    --function-name my-function \
    --batch-size 5 \
    --event-source-arn arn:aws:sqs:us-west-2:123456789012:mySQSqueue
```

输出：

```
{
    "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "StateTransitionReason": "USER_INITIATED",
    "LastModified": 1569284520.333,
    "BatchSize": 5,
    "State": "Creating",
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 事件源映射。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEventSourceMapping。

create-function

以下代码示例演示了如何使用 create-function。

Amazon CLI

创建 Lambda 函数

以下 create-function 示例创建一个名为 my-function 的 Lambda 函数。

```
aws lambda create-function \
    --function-name my-function \
    --runtime nodejs22.x \
    --zip-file fileb://my-function.zip \
    --handler my-function.handler \
    --role arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-tges6bf4
```

的内容my-function.zip：

```
This file is a deployment package that contains your function code and any
 dependencies.
```

输出：

```
{
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "CodeSha256": "PFn4S+er27qk+UuZSTKEQfNKG/XNn7QJs90mJgq6oH8=",
    "FunctionName": "my-function",
    "CodeSize": 308,
    "RevisionId": "873282ed-4cd3-4dc8-a069-d0c647e470c6",
    "MemorySize": 128,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "Version": "$LATEST",
    "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-
zgur6bf4",
    "Timeout": 3,
    "LastModified": "2025-10-14T22:26:11.234+0000",
    "Handler": "my-function.handler",
    "Runtime": "nodejs22.x",
    "Description": ""
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 配置 Lambda 函数内存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFunction。

delete-alias

以下代码示例演示了如何使用 delete-alias。

Amazon CLI

删除 Lambda 函数别名

以下 `delete-alias` 示例从 `my-function` Lambda 函数中删除了名为 `LIVE` 的别名。

```
aws lambda delete-alias \
    --function-name my-function \
    --name LIVE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAlias。

`delete-event-source-mapping`

以下代码示例演示了如何使用 `delete-event-source-mapping`。

Amazon CLI

删除事件源和 Amazon Lambda 函数之间的映射

以下 `delete-event-source-mapping` 示例删除 SQS 队列和 `my-function` Lambda 函数之间的映射。

```
aws lambda delete-event-source-mapping \
    --uuid  a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

输出：

```
{
    "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "StateTransitionReason": "USER_INITIATED",
    "LastModified": 1569285870.271,
    "BatchSize": 5,
    "State": "Deleting",
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 事件源映射。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEventSourceMapping。

`delete-function-concurrency`

以下代码示例演示了如何使用 `delete-function-concurrency`。

Amazon CLI

从函数中删除预留的并发执行限制

以下 `delete-function-concurrency` 示例从 `my-function` 函数中删除了预留的并发执行限制。

```
aws lambda delete-function-concurrency \
    --function-name  my-function
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的为 Lambda 函数预留并发。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFunctionConcurrency。

`delete-function-event-invoke-config`

以下代码示例演示了如何使用 `delete-function-event-invoke-config`。

Amazon CLI

删除异步调用配置

以下 `delete-function-event-invoke-config` 示例删除指定函数的 GREEN 别名的异步调用配置。

```
aws lambda delete-function-event-invoke-config --function-name my-function:GREEN
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFunctionEventInvokeConfig。

`delete-function`

以下代码示例演示了如何使用 `delete-function`。

Amazon CLI

示例 1：按函数名称删除 Lambda 函数

以下 `delete-function` 示例删除通过指定函数名称命名为 `my-function` 的 Lambda 函数。

```
aws lambda delete-function \
    --function-name my-function
```

此命令不生成任何输出。

示例 2：按函数 ARN 删除 Lambda 函数

以下 `delete-function` 示例删除通过指定函数 ARN 命名为 `my-function` 的 Lambda 函数。

```
aws lambda delete-function \
    --function-name arn:aws:lambda:us-west-2:123456789012:function:my-function
```

此命令不生成任何输出。

示例 3：按部分函数 ARN 删除 Lambda 函数

以下 `delete-function` 示例删除通过指定函数的部分 ARN 命名为 `my-function` 的 Lambda 函数。

```
aws lambda delete-function \
    --function-name 123456789012:function:my-function
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 函数配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFunction。

`delete-layer-version`

以下代码示例演示了如何使用 `delete-layer-version`。

Amazon CLI

删除 Lambda 层的某个版本

以下 `delete-layer-version` 示例删除名为 `my-layer` 的层的版本 2。

```
aws lambda delete-layer-version \
    --layer-name my-layer \
    --version-number 2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLayerVersion。

delete-provisioned-concurrency-config

以下代码示例演示了如何使用 delete-provisioned-concurrency-config。

Amazon CLI

删除预置并发配置

以下 delete-provisioned-concurrency-config 示例删除了指定函数 GREEN 别名的预置并发配置。

```
aws lambda delete-provisioned-concurrency-config \
    --function-name my-function \
    --qualifier GREEN
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProvisionedConcurrencyConfig。

get-account-settings

以下代码示例演示了如何使用 get-account-settings。

Amazon CLI

在 Amazon 区域中检索关于账户的详细信息

以下 get-account-settings 示例展示了账户的 Lambda 限制和使用信息。

```
aws lambda get-account-settings
```

输出：

```
{
    "AccountLimit": {
       "CodeSizeUnzipped": 262144000,
       "UnreservedConcurrentExecutions": 1000,
       "ConcurrentExecutions": 1000,
       "CodeSizeZipped": 52428800,
       "TotalCodeSize": 80530636800
    },
    "AccountUsage": {
       "FunctionCount": 4,
       "TotalCodeSize": 9426
    }
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 限制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccountSettings。

get-alias

以下代码示例演示了如何使用 get-alias。

Amazon CLI

检索关于函数别名的详细信息

以下 get-alias 示例展示了 my-function Lambda 函数中名为 LIVE 的别名的详细信息。

```
aws lambda get-alias \
    --function-name my-function \
    --name LIVE
```

输出：

```
{
    "FunctionVersion": "3",
    "Name": "LIVE",
    "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",
    "RevisionId": "594f41fb-b85f-4c20-95c7-6ca5f2a92c93",
    "Description": "alias for live version of function"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAlias。

get-event-source-mapping

以下代码示例演示了如何使用 get-event-source-mapping。

Amazon CLI

检索有关事件源映射的详细信息

以下 get-event-source-mapping 示例显示 SQS 队列和 my-function Lambda 函数之间的映射的详细信息。

```
aws lambda get-event-source-mapping \
    --uuid "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

输出：

```
{
    "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "StateTransitionReason": "USER_INITIATED",
    "LastModified": 1569284520.333,
    "BatchSize": 5,
    "State": "Enabled",
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 事件源映射。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEventSourceMapping。

get-function-concurrency

以下代码示例演示了如何使用 get-function-concurrency。

Amazon CLI

查看函数的预留并发设置

以下 get-function-concurrency 示例检索了指定函数的预留并发设置。

```
aws lambda get-function-concurrency \
    --function-name my-function
```

输出：

```
{
    "ReservedConcurrentExecutions": 250
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunctionConcurrency。

get-function-configuration

以下代码示例演示了如何使用 get-function-configuration。

Amazon CLI

检索 Lambda 函数的版本特定设置

以下 get-function-configuration 示例展示了 my-function 函数版本 2 的设置。

```
aws lambda get-function-configuration \
    --function-name  my-function:2
```

输出：

```
{
    "FunctionName": "my-function",
    "LastModified": "2019-09-26T20:28:40.438+0000",
    "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",
    "MemorySize": 256,
    "Version": "2",
    "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qyq",
    "Timeout": 3,
    "Runtime": "nodejs10.x",
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "CodeSha256": "5tT2qgzYUHaqwR716pZ2dpkn/0J1FrzJmlKidWoaCgk=",
```

```
    "Description": "",
    "VpcConfig": {
        "SubnetIds": [],
        "VpcId": "",
        "SecurityGroupIds": []
    },
    "CodeSize": 304,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:2",
    "Handler": "index.handler"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 函数配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunctionConfiguration。

get-function-event-invoke-config

以下代码示例演示了如何使用 get-function-event-invoke-config。

Amazon CLI

查看异步调用配置

以下 get-function-event-invoke-config 示例检索指定函数的 BLUE 别名的异步调用配置。

```
aws lambda get-function-event-invoke-config \
    --function-name my-function:BLUE
```

输出：

```
{
    "LastModified": 1577824396.653,
    "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:BLUE",
    "MaximumRetryAttempts": 0,
    "MaximumEventAgeInSeconds": 3600,
    "DestinationConfig": {
        "OnSuccess": {},
        "OnFailure": {
            "Destination": "arn:aws:sqs:us-east-2:123456789012:failed-invocations"
        }
```

```
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunctionEventInvokeConfig。

get-function

以下代码示例演示了如何使用 get-function。

Amazon CLI

检索有关函数的信息

以下 get-function 示例显示有关 my-function 函数的信息：

```
aws lambda get-function \
    --function-name  my-function
```

输出：

```
{
    "Concurrency": {
        "ReservedConcurrentExecutions": 100
    },
    "Code": {
        "RepositoryType": "S3",
        "Location": "https://awslambda-us-west-2-tasks.s3.us-west-2.amazonaws.com/
snapshots/123456789012/my-function..."
    },
    "Configuration": {
        "TracingConfig": {
            "Mode": "PassThrough"
        },
        "Version": "$LATEST",
        "CodeSha256": "5tT2qgzYUHoqwR616pZ2dpkn/0J1FrzJmlKidWaaCgk=",
        "FunctionName": "my-function",
        "VpcConfig": {
            "SubnetIds": [],
            "VpcId": "",
            "SecurityGroupIds": []
        },
        "MemorySize": 128,
```

```
        "RevisionId": "28f0fb31-5c5c-43d3-8955-03e76c5c1075",
        "CodeSize": 304,
        "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
        "Handler": "index.handler",
        "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-role-
uy3l9qyq",
        "Timeout": 3,
        "LastModified": "2025-09-24T18:20:35.054+0000",
        "Runtime": "nodejs22.x",
        "Description": ""
    }
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 配置 Lambda 函数内存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFunction。

get-layer-version-by-arn

以下代码示例演示了如何使用 get-layer-version-by-arn。

Amazon CLI

检索有关 Lambda 层版本的信息

以下 get-layer-version-by-arn 示例显示具有指定 Amazon 资源名称（ARN）的层版本信息。

```
aws lambda get-layer-version-by-arn \
    --arn "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-Python311-
SciPy1x:2"
```

输出：

```
{
    "LayerVersionArn": "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-
Python311-SciPy1x:2",
    "Description": "AWS Lambda SciPy layer for Python 3.11 (scipy-1.1.0,
 numpy-1.15.4) https://github.com/scipy/scipy/releases/tag/v1.1.0 https://
github.com/numpy/numpy/releases/tag/v1.15.4",
    "CreatedDate": "2023-10-12T10:09:38.398+0000",
    "LayerArn": "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-Python311-
SciPy1x",
```

```
    "Content": {
        "CodeSize": 41784542,
        "CodeSha256": "GGmv8ocUw4cly0T8HL0Vx/f5V4RmSCGNjDIslY4VskM=",
        "Location": "https://awslambda-us-west-2-layers.s3.us-west-2.amazonaws.com/
snapshots/123456789012/..."
    },
    "Version": 2,
    "CompatibleRuntimes": [
        "python3.11"
    ],
    "LicenseInfo": "SciPy: https://github.com/scipy/scipy/blob/main/LICENSE.txt,
 NumPy: https://github.com/numpy/numpy/blob/main/LICENSE.txt"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLayerVersionByArn。

get-layer-version-policy

以下代码示例演示了如何使用 get-layer-version-policy。

Amazon CLI

检索 Lambda 层版本的权限策略

以下 get-layer-version-policy 示例显示名为 my-layer 的层的版本 1 的策略信息。

```
aws lambda get-layer-version-policy \
    --layer-name my-layer \
    --version-number 1
```

输出：

```
{
    "Policy": {
        "Version":"2012-10-17",
        "Id":"default",
        "Statement":
        [
            {
                "Sid":"xaccount",
```

```
                "Effect":"Allow",
                "Principal": {"AWS":"arn:aws:iam::123456789012:root"},
                "Action":"lambda:GetLayerVersion",
                "Resource":"arn:aws:lambda:us-west-2:123456789012:layer:my-layer:1"
            }
        ]
    },
    "RevisionId": "c68f21d2-cbf0-4026-90f6-1375ee465cd0"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLayerVersionPolicy。

get-layer-version

以下代码示例演示了如何使用 get-layer-version。

Amazon CLI

检索有关 Lambda 层版本的信息

以下 get-layer-version 示例显示名为 my-layer 的层的版本 1 的策略信息。

```
aws lambda get-layer-version \
    --layer-name my-layer \
    --version-number 1
```

输出：

```
{
    "Content": {
        "Location": "https://awslambda-us-east-2-layers.s3.us-east-2.amazonaws.com/
snapshots/123456789012/my-layer-4aaa2fbb-ff77-4b0a-ad92-5b78a716a96a?
versionId=27iWyA73cCAYqyH...",
        "CodeSha256": "tv9jJO+rPbXUUXuRKi7CwHzKtLDkDRJLB3cC3Z/ouXo=",
        "CodeSize": 169
    },
    "LayerArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer",
    "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer:1",
    "Description": "My Python layer",
    "CreatedDate": "2018-11-14T23:03:52.894+0000",
```

```
    "Version": 1,
    "LicenseInfo": "MIT",
    "CompatibleRuntimes": [
        "python3.10",
        "python3.11"
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLayerVersion。

get-policy

以下代码示例演示了如何使用 get-policy。

Amazon CLI

检索函数、版本或别名的基于资源的 IAM 策略

以下 get-policy 示例展示了有关 my-function Lambda 函数的策略信息。

```
aws lambda get-policy \
    --function-name my-function
```

输出：

```
{
    "Policy": {
        "Version":"2012-10-17",
        "Id":"default",
        "Statement":
        [
            {
                "Sid":"iot-events",
                "Effect":"Allow",
                "Principal": {"Service":"iotevents.amazonaws.com"},
                "Action":"lambda:InvokeFunction",
                "Resource":"arn:aws:lambda:us-west-2:123456789012:function:my-
function"
            }
        ]
```

```
    },
    "RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的对 Amazon Lambda 使用基于资源的策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicy。

`get-provisioned-concurrency-config`

以下代码示例演示了如何使用 `get-provisioned-concurrency-config`。

Amazon CLI

查看预置并发配置

以下 `get-provisioned-concurrency-config` 示例展示了指定函数 BLUE 别名的预置并发配置的详细信息。

```
aws lambda get-provisioned-concurrency-config \
    --function-name my-function \
    --qualifier BLUE
```

输出：

```
{
    "RequestedProvisionedConcurrentExecutions": 100,
    "AvailableProvisionedConcurrentExecutions": 100,
    "AllocatedProvisionedConcurrentExecutions": 100,
    "Status": "READY",
    "LastModified": "2019-12-31T20:28:49+0000"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProvisionedConcurrencyConfig。

`invoke`

以下代码示例演示了如何使用 `invoke`。

Amazon CLI

示例 1：同步调用 Lambda 函数

以下 `invoke` 示例同步调用该 `my-function` 函数。如果使用 `cli-binary-format` CLI 版本 2，则 Amazon 选项是必需的。有关更多信息，请参阅《Amazon 命令行界面用户指南》中 Amazon CLI 支持的全局命令行选项。

```
aws lambda invoke \
    --function-name my-function \
    --cli-binary-format raw-in-base64-out \
    --payload '{ "name": "Bob" }' \
    response.json
```

输出：

```
{
    "ExecutedVersion": "$LATEST",
    "StatusCode": 200
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的同步调用 Lambda 函数。

示例 2：异步调用 Lambda 函数

以下 `invoke` 示例异步调用该 `my-function` 函数。如果使用 `cli-binary-format` CLI 版本 2，则 Amazon 选项是必需的。有关更多信息，请参阅《Amazon 命令行界面用户指南》中 Amazon CLI 支持的全局命令行选项。

```
aws lambda invoke \
    --function-name my-function \
    --invocation-type Event \
    --cli-binary-format raw-in-base64-out \
    --payload '{ "name": "Bob" }' \
    response.json
```

输出：

```
{
    "StatusCode": 202
```

```
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的异步调用 Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Invoke。

`list-aliases`

以下代码示例演示了如何使用 `list-aliases`。

Amazon CLI

检索 Lambda 函数的别名列表

以下 `list-aliases` 示例显示 `my-function` Lambda 函数的别名列表。

```
aws lambda list-aliases \
    --function-name my-function
```

输出：

```
{
    "Aliases": [
        {
            "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:BETA",
            "RevisionId": "a410117f-ab16-494e-8035-7e204bb7933b",
            "FunctionVersion": "2",
            "Name": "BETA",
            "Description": "alias for beta version of function"
        },
        {
            "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:LIVE",
            "RevisionId": "21d40116-f8b1-40ba-9360-3ea284da1bb5",
            "FunctionVersion": "1",
            "Name": "LIVE",
            "Description": "alias for live version of function"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAliases。

list-event-source-mappings

以下代码示例演示了如何使用 list-event-source-mappings。

Amazon CLI

列出函数的事件源映射

以下 list-event-source-mappings 示例显示 my-function Lambda 函数的事件源映射列表。

```
aws lambda list-event-source-mappings \
    --function-name my-function
```

输出：

```
{
    "EventSourceMappings": [
        {
            "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "StateTransitionReason": "USER_INITIATED",
            "LastModified": 1569284520.333,
            "BatchSize": 5,
            "State": "Enabled",
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
            "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 事件源映射。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEventSourceMappings。

list-function-event-invoke-configs

以下代码示例演示了如何使用 list-function-event-invoke-configs。

Amazon CLI

查看异步调用配置列表

以下 `list-function-event-invoke-configs` 示例列出指定函数的异步调用配置。

```
aws lambda list-function-event-invoke-configs \
    --function-name my-function
```

输出：

```
{
    "FunctionEventInvokeConfigs": [
        {
            "LastModified": 1577824406.719,
            "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:GREEN",
            "MaximumRetryAttempts": 2,
            "MaximumEventAgeInSeconds": 1800
        },
        {
            "LastModified": 1577824396.653,
            "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:BLUE",
            "MaximumRetryAttempts": 0,
            "MaximumEventAgeInSeconds": 3600
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFunctionEventInvokeConfigs。

list-functions

以下代码示例演示了如何使用 `list-functions`。

Amazon CLI

检索 Lambda 函数列表

以下 `list-functions` 示例显示当前用户所有函数的列表。

```
aws lambda list-functions
```

输出：

```
{
    "Functions": [
        {
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "Version": "$LATEST",
            "CodeSha256": "dBG9m8SGdmlEjw/JYXlhhvCrAv5TxvXsbL/RMr0fT/I=",
            "FunctionName": "helloworld",
            "MemorySize": 128,
            "RevisionId": "1718e831-badf-4253-9518-d0644210af7b",
            "CodeSize": 294,
            "FunctionArn": "arn:aws:lambda:us-
west-2:123456789012:function:helloworld",
            "Handler": "helloworld.handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-
role-zgur6bf4",
            "Timeout": 3,
            "LastModified": "2025-09-23T18:32:33.857+0000",
            "Runtime": "nodejs22.x",
            "Description": ""
        },
        {
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "Version": "$LATEST",
            "CodeSha256": "sU0cJ2/hOZevwV/lTxCuQqK3gDZP3i8gUoqUUVRmY6E=",
            "FunctionName": "my-function",
            "VpcConfig": {
                "SubnetIds": [],
                "VpcId": "",
                "SecurityGroupIds": []
            },
            "MemorySize": 256,
            "RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668",
            "CodeSize": 266,
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
```

```
            "Handler": "index.handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
            "Timeout": 3,
            "LastModified": "2025-10-01T16:47:28.490+0000",
            "Runtime": "nodejs22.x",
            "Description": ""
        },
        {
            "Layers": [
                {
                    "CodeSize": 41784542,
                    "Arn": "arn:aws:lambda:us-west-2:420165488524:layer:AWSLambda-
Python37-SciPy1x:2"
                },
                {
                    "CodeSize": 4121,
                    "Arn": "arn:aws:lambda:us-
west-2:123456789012:layer:pythonLayer:1"
                }
            ],
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "Version": "$LATEST",
            "CodeSha256": "ZQukCqxtkqFgyF2cU41Avj99TKQ/hNihPtDtRcc08mI=",
            "FunctionName": "my-python-function",
            "VpcConfig": {
                "SubnetIds": [],
                "VpcId": "",
                "SecurityGroupIds": []
            },
            "MemorySize": 128,
            "RevisionId": "80b4eabc-acf7-4ea8-919a-e874c213707d",
            "CodeSize": 299,
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
python-function",
            "Handler": "lambda_function.lambda_handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/my-python-function-
role-z5g7dr6n",
            "Timeout": 3,
            "LastModified": "2025-10-01T19:40:41.643+0000",
            "Runtime": "python3.11",
            "Description": ""
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 配置 Lambda 函数内存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFunctions。

`list-layer-versions`

以下代码示例演示了如何使用 `list-layer-versions`。

Amazon CLI

列出 Amazon Lambda 层的版本

以下 `list-layers-versions` 示例显示名为 `my-layer` 的层的版本信息。

```
aws lambda list-layer-versions \
    --layer-name my-layer
```

输出：

```
{
    "Layers": [
        {
            "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-
layer:2",
            "Version": 2,
            "Description": "My layer",
            "CreatedDate": "2023-11-15T00:37:46.592+0000",
            "CompatibleRuntimes": [
                "python3.10",
                "python3.11"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLayerVersions。

list-layers

以下代码示例演示了如何使用 list-layers。

Amazon CLI

列出与您的函数的运行时兼容的层

以下 list-layers 示例显示了与 Python 3.11 运行时兼容的层的信息。

```
aws lambda list-layers \
    --compatible-runtime python3.11
```

输出：

```
{
    "Layers": [
        {
            "LayerName": "my-layer",
            "LayerArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer",
            "LatestMatchingVersion": {
                "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-
layer:2",
                "Version": 2,
                "Description": "My layer",
                "CreatedDate": "2023-11-15T00:37:46.592+0000",
                "CompatibleRuntimes": [
                    "python3.10",
                    "python3.11"
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLayers。

list-provisioned-concurrency-configs

以下代码示例演示了如何使用 list-provisioned-concurrency-configs。

Amazon CLI

获取预置并发配置列表

以下 `list-provisioned-concurrency-configs` 示例列出了指定函数的预置并发配置。

```
aws lambda list-provisioned-concurrency-configs \
    --function-name my-function
```

输出：

```
{
    "ProvisionedConcurrencyConfigs": [
        {
            "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:GREEN",
            "RequestedProvisionedConcurrentExecutions": 100,
            "AvailableProvisionedConcurrentExecutions": 100,
            "AllocatedProvisionedConcurrentExecutions": 100,
            "Status": "READY",
            "LastModified": "2019-12-31T20:29:00+0000"
        },
        {
            "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:BLUE",
            "RequestedProvisionedConcurrentExecutions": 100,
            "AvailableProvisionedConcurrentExecutions": 100,
            "AllocatedProvisionedConcurrentExecutions": 100,
            "Status": "READY",
            "LastModified": "2019-12-31T20:28:49+0000"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProvisionedConcurrencyConfigs。

list-tags

以下代码示例演示了如何使用 `list-tags`。

Amazon CLI

检索 Lambda 函数的标签列表

以下 `list-tags` 示例展示了附加到 `my-function` Lambda 函数的标签。

```
aws lambda list-tags \
    --resource arn:aws:lambda:us-west-2:123456789012:function:my-function
```

输出：

```
{
    "Tags": {
        "Category": "Web Tools",
        "Department": "Sales"
    }
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的标记 Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTags。

`list-versions-by-function`

以下代码示例演示了如何使用 `list-versions-by-function`。

Amazon CLI

检索函数的版本列表

以下 `list-versions-by-function` 示例展示了 `my-function` Lambda 函数的版本列表。

```
aws lambda list-versions-by-function \
    --function-name my-function
```

输出：

```
{
    "Versions": [
        {
            "TracingConfig": {
                "Mode": "PassThrough"
            },
```

```
            "Version": "$LATEST",
            "CodeSha256": "sU0cJ2/hOZevwV/lTxCuQqK3gDZP3i8gUoqUUVRmY6E=",
            "FunctionName": "my-function",
            "VpcConfig": {
                "SubnetIds": [],
                "VpcId": "",
                "SecurityGroupIds": []
            },
            "MemorySize": 256,
            "RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668",
            "CodeSize": 266,
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:$LATEST",
            "Handler": "index.handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
            "Timeout": 3,
            "LastModified": "2019-10-01T16:47:28.490+0000",
            "Runtime": "nodejs10.x",
            "Description": ""
        },
        {
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "Version": "1",
            "CodeSha256": "5tT2qgzYUHoqwR616pZ2dpkn/0J1FrzJmlKidWaaCgk=",
            "FunctionName": "my-function",
            "VpcConfig": {
                "SubnetIds": [],
                "VpcId": "",
                "SecurityGroupIds": []
            },
            "MemorySize": 256,
            "RevisionId": "949c8914-012e-4795-998c-e467121951b1",
            "CodeSize": 304,
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:1",
            "Handler": "index.handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
            "Timeout": 3,
            "LastModified": "2019-09-26T20:28:40.438+0000",
            "Runtime": "nodejs10.x",
```

```
            "Description": "new version"
        },
        {
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "Version": "2",
            "CodeSha256": "sU0cJ2/hOZevwV/lTxCuQqK3gDZP3i8gUoqUUVRmY6E=",
            "FunctionName": "my-function",
            "VpcConfig": {
                "SubnetIds": [],
                "VpcId": "",
                "SecurityGroupIds": []
            },
            "MemorySize": 256,
            "RevisionId": "cd669f21-0f3d-4e1c-9566-948837f2e2ea",
            "CodeSize": 266,
            "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:2",
            "Handler": "index.handler",
            "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
            "Timeout": 3,
            "LastModified": "2019-10-01T16:47:28.490+0000",
            "Runtime": "nodejs10.x",
            "Description": "newer version"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVersionsByFunction。

publish-layer-version

以下代码示例演示了如何使用 publish-layer-version。

Amazon CLI

创建 Lambda 层版本

以下 publish-layer-version 示例创建一个新的 Python 库层版本。该命令检索指定 S3 存储桶中名为 layer.zip 的文件的层内容。

```
aws lambda publish-layer-version \
    --layer-name my-layer \
    --description "My Python layer" \
    --license-info "MIT" \
    --content S3Bucket=lambda-layers-us-west-2-123456789012,S3Key=layer.zip \
    --compatible-runtimes python3.10 python3.11
```

输出：

```
{
    "Content": {
        "Location": "https://awslambda-us-west-2-layers.s3.us-west-2.amazonaws.com/
snapshots/123456789012/my-layer-4aaa2fbb-ff77-4b0a-ad92-5b78a716a96a?
versionId=27iWyA73cCAYqyH...",
        "CodeSha256": "tv9jJO+rPbXUUXuRKi7CwHzKtLDkDRJLB3cC3Z/ouXo=",
        "CodeSize": 169
    },
    "LayerArn": "arn:aws:lambda:us-west-2:123456789012:layer:my-layer",
    "LayerVersionArn": "arn:aws:lambda:us-west-2:123456789012:layer:my-layer:1",
    "Description": "My Python layer",
    "CreatedDate": "2023-11-14T23:03:52.894+0000",
    "Version": 1,
    "LicenseInfo": "MIT",
    "CompatibleRuntimes": [
        "python3.10",
        "python3.11"
    ]
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PublishLayerVersion。

publish-version

以下代码示例演示了如何使用 publish-version。

Amazon CLI

发布函数的新版本

以下 publish-version 示例发布了 my-function Lambda 函数的新版本。

```
aws lambda publish-version \
    --function-name my-function
```

输出：

```
{
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "CodeSha256": "dBG9m8SGdmlEjw/JYXlhhvCrAv5TxvXsbL/RMr0fT/I=",
    "FunctionName": "my-function",
    "CodeSize": 294,
    "RevisionId": "f31d3d39-cc63-4520-97d4-43cd44c94c20",
    "MemorySize": 128,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:3",
    "Version": "2",
    "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-
zgur6bf4",
    "Timeout": 3,
    "LastModified": "2019-09-23T18:32:33.857+0000",
    "Handler": "my-function.handler",
    "Runtime": "nodejs10.x",
    "Description": ""
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PublishVersion。

put-function-concurrency

以下代码示例演示了如何使用 put-function-concurrency。

Amazon CLI

配置函数的预留并发限制

以下 put-function-concurrency 示例为 my-function 函数配置了 100 个预留并发执行。

```
aws lambda put-function-concurrency \
```

```
    --function-name  my-function  \
    --reserved-concurrent-executions 100
```

输出：

```
{
    "ReservedConcurrentExecutions": 100
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的为 Lambda 函数预留并发。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutFunctionConcurrency。

put-function-event-invoke-config

以下代码示例演示了如何使用 put-function-event-invoke-config。

Amazon CLI

配置异步调用的错误处理

以下 put-function-event-invoke-config 示例为指定函数设置了最长一小时的事件持续时间，并且禁用重试。

```
aws lambda put-function-event-invoke-config \
    --function-name my-function \
    --maximum-event-age-in-seconds 3600 \
    --maximum-retry-attempts 0
```

输出：

```
{
    "LastModified": 1573686021.479,
    "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function:
$LATEST",
    "MaximumRetryAttempts": 0,
    "MaximumEventAgeInSeconds": 3600,
    "DestinationConfig": {
        "OnSuccess": {},
        "OnFailure": {}
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutFunctionEventInvokeConfig。

put-provisioned-concurrency-config

以下代码示例演示了如何使用 put-provisioned-concurrency-config。

Amazon CLI

分配预置并发

以下 put-provisioned-concurrency-config 示例为指定函数的 BLUE 别名分配了 100 个预置并发。

```
aws lambda put-provisioned-concurrency-config \
    --function-name my-function \
    --qualifier BLUE \
    --provisioned-concurrent-executions 100
```

输出：

```
{
    "Requested ProvisionedConcurrentExecutions": 100,
    "Allocated ProvisionedConcurrentExecutions": 0,
    "Status": "IN_PROGRESS",
    "LastModified": "2019-11-21T19:32:12+0000"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutProvisionedConcurrencyConfig。

remove-layer-version-permission

以下代码示例演示了如何使用 remove-layer-version-permission。

Amazon CLI

删除层版本权限

以下 remove-layer-version-permission 示例删除账户配置层版本的权限。

```
aws lambda remove-layer-version-permission \
    --layer-name my-layer \
    --statement-id xaccount \
    --version-number 1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveLayerVersionPermission。

remove-permission

以下代码示例演示了如何使用 remove-permission。

Amazon CLI

从现有 Lambda 函数中删除权限

以下 remove-permission 示例删除了调用名为 my-function 的函数的权限。

```
aws lambda remove-permission \
    --function-name my-function \
    --statement-id sns
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的对 Amazon Lambda 使用基于资源的策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemovePermission。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签添加到现有 Lambda 函数

以下 tag-resource 示例向指定的 Lambda 函数添加了键名称为 DEPARTMENT 和值为 Department A 的标签。

```
aws lambda tag-resource \
    --resource arn:aws:lambda:us-west-2:123456789012:function:my-function \
    --tags "DEPARTMENT=Department A"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的标记 Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从现有 Lambda 函数中删除标签

以下 untag-resource 示例从 my-function Lambda 函数中删除了键名称为 DEPARTMENT 的标签。

```
aws lambda untag-resource \
    --resource arn:aws:lambda:us-west-2:123456789012:function:my-function \
    --tag-keys DEPARTMENT
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的标记 Lambda 函数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-alias

以下代码示例演示了如何使用 update-alias。

Amazon CLI

更新函数别名

以下 update-alias 示例会更新名为 LIVE 的别名，该别名指向 my-function Lambda 函数的版本 3。

```
aws lambda update-alias \
    --function-name my-function \
    --function-version 3 \
    --name LIVE
```

输出：

```
{
    "FunctionVersion": "3",
    "Name": "LIVE",
    "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",
    "RevisionId": "594f41fb-b85f-4c20-95c7-6ca5f2a92c93",
    "Description": "alias for live version of function"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAlias。

update-event-source-mapping

以下代码示例演示了如何使用 update-event-source-mapping。

Amazon CLI

更新事件源和 Amazon Lambda 函数之间的映射

以下 update-event-source-mapping 示例将指定映射中的批量大小更新为 8。

```
aws lambda update-event-source-mapping \
    --uuid  "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \
    --batch-size 8
```

输出：

```
{
    "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "StateTransitionReason": "USER_INITIATED",
    "LastModified": 1569284520.333,
    "BatchSize": 8,
    "State": "Updating",
```

```
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的 Amazon Lambda 事件源映射。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEventSourceMapping。

update-function-code

以下代码示例演示了如何使用 update-function-code。

Amazon CLI

更新 Lambda 函数代码

以下 update-function-code 示例将 my-function 函数的未发布（$LATEST）版本的代码替换为指定 zip 文件的内容。

```
aws lambda update-function-code \
    --function-name  my-function \
    --zip-file fileb://my-function.zip
```

输出：

```
{
    "FunctionName": "my-function",
    "LastModified": "2019-09-26T20:28:40.438+0000",
    "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",
    "MemorySize": 256,
    "Version": "$LATEST",
    "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qyq",
    "Timeout": 3,
    "Runtime": "nodejs10.x",
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "CodeSha256": "5tT2qgzYUHaqwR716pZ2dpkn/0J1FrzJmlKidWoaCgk=",
    "Description": "",
    "VpcConfig": {
        "SubnetIds": [],
        "VpcId": "",
```

```
        "SecurityGroupIds": []
    },
    "CodeSize": 304,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "Handler": "index.handler"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数选项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFunctionCode。

update-function-configuration

以下代码示例演示了如何使用 update-function-configuration。

Amazon CLI

修改函数的配置

以下 update-function-configuration 示例将 my-function 函数的未发布（$LATEST）版本的内存大小修改为 256 MB。

```
aws lambda update-function-configuration \
    --function-name  my-function \
    --memory-size 256
```

输出：

```
{
    "FunctionName": "my-function",
    "LastModified": "2019-09-26T20:28:40.438+0000",
    "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",
    "MemorySize": 256,
    "Version": "$LATEST",
    "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qyq",
    "Timeout": 3,
    "Runtime": "nodejs10.x",
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "CodeSha256": "5tT2qgzYUHaqwR716pZ2dpkn/0J1FrzJmlKidWoaCgk=",
    "Description": "",
```

```
    "VpcConfig": {
        "SubnetIds": [],
        "VpcId": "",
        "SecurityGroupIds": []
    },
    "CodeSize": 304,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "Handler": "index.handler"
}
```

有关更多信息，请参阅《Amazon Lambda 开发人员指南》中的配置 Amazon Lambda 函数选项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFunctionConfiguration。

update-function-event-invoke-config

以下代码示例演示了如何使用 update-function-event-invoke-config。

Amazon CLI

更新异步调用配置

以下 update-function-event-invoke-config 示例将失败时目标添加到指定函数的现有异步调用配置中。

```
aws lambda update-function-event-invoke-config \
    --function-name my-function \
    --destination-config '{"OnFailure":{"Destination": "arn:aws:sqs:us-
east-2:123456789012:destination"}}'
```

输出：

```
{
    "LastModified": 1573687896.493,
    "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function:
$LATEST",
    "MaximumRetryAttempts": 0,
    "MaximumEventAgeInSeconds": 3600,
    "DestinationConfig": {
        "OnSuccess": {},
        "OnFailure": {
            "Destination": "arn:aws:sqs:us-east-2:123456789012:destination"
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFunctionEventInvokeConfig。

使用 Amazon CLI 的 License Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 License Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-license-configuration`

以下代码示例演示了如何使用 `create-license-configuration`。

Amazon CLI

示例 1：创建许可证配置

以下 `create-license-configuration` 示例使用 10 个核心的硬限制创建了一个许可证配置。

```
aws license-manager create-license-configuration --name my-license-configuration \
    --license-counting-type Core \
    --license-count 10 \
    --license-count-hard-limit
```

输出：

```
{
  "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba41EXAMPLE1111"
}
```

示例 2：创建许可证配置

以下 create-license-configuration 示例使用 100 个 vCPU 的软限制创建了一个许可证配置。它使用规则来启用 vCPU 优化。

```
aws license-manager create-license-configuration --name my-license-configuration
    --license-counting-type vCPU \
    --license-count 100 \
    --license-rules "#honorVcpuOptimization=true"
```

输出：

```
{
  "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba41EXAMPLE2222"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLicenseConfiguration。

delete-license-configuration

以下代码示例演示了如何使用 delete-license-configuration。

Amazon CLI

删除许可证配置

以下 delete-license-configuration 示例删除指定的许可证配置。

```
aws license-manager delete-license-configuration \
    --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLicenseConfiguration。

get-license-configuration

以下代码示例演示了如何使用 get-license-configuration。

Amazon CLI

获取许可证配置信息

以下 get-license-configuration 示例显示指定许可证配置的详细信息。

```
aws license-manager get-license-configuration \
    --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

输出：

```
{
    "LicenseConfigurationId": "lic-38b658717b87478aaa7c00883EXAMPLE",
    "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE",
    "Name": "my-license-configuration",
    "LicenseCountingType": "vCPU",
    "LicenseRules": [],
    "LicenseCountHardLimit": false,
    "ConsumedLicenses": 0,
    "Status": "AVAILABLE",
    "OwnerAccountId": "123456789012",
    "ConsumedLicenseSummaryList": [
        {
            "ResourceType": "EC2_INSTANCE",
            "ConsumedLicenses": 0
        },
        {
            "ResourceType": "EC2_HOST",
            "ConsumedLicenses": 0
        },
        {
            "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
            "ConsumedLicenses": 0
        }
```

```
    ],
    "ManagedResourceSummaryList": [
        {
            "ResourceType": "EC2_INSTANCE",
            "AssociationCount": 0
        },
        {
            "ResourceType": "EC2_HOST",
            "AssociationCount": 0
        },
        {
            "ResourceType": "EC2_AMI",
            "AssociationCount": 2
        },
        {
            "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
            "AssociationCount": 0
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLicenseConfiguration。

get-service-settings

以下代码示例演示了如何使用 get-service-settings。

Amazon CLI

获取 License Manager 设置

以下 get-service-settings 示例显示当前区域中 License Manager 的服务设置。

```
aws license-manager get-service-settings
```

以下显示当跨账户资源发现为禁用时的输出示例。

```
{
    "OrganizationConfiguration": {
        "EnableIntegration": false
    },
    "EnableCrossAccountsDiscovery": false
```

```
}
```

以下显示当跨账户资源发现为启用时的输出示例。

```
{
    "S3BucketArn": "arn:aws:s3:::aws-license-manager-service-c22d6279-35c4-47c4-bb",
    "OrganizationConfiguration": {
        "EnableIntegration": true
    },
    "EnableCrossAccountsDiscovery": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceSettings。

list-associations-for-license-configuration

以下代码示例演示了如何使用 list-associations-for-license-configuration。

Amazon CLI

获取许可证配置的关联

以下 list-associations-for-license-configuration 示例显示指定许可证配置的关联的详细信息。

```
aws license-manager list-associations-for-license-configuration \
    --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

输出：

```
{
    "LicenseConfigurationAssociations": [
        {
            "ResourceArn": "arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0",
            "ResourceType": "EC2_AMI",
            "ResourceOwnerId": "123456789012",
            "AssociationTime": 1568825118.617
        },
        {
            "ResourceArn": "arn:aws:ec2:us-west-2::image/ami-0abcdef1234567890",
            "ResourceType": "EC2_AMI",
```

```
            "ResourceOwnerId": "123456789012",
            "AssociationTime": 1568825118.946
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssociationsForLicenseConfiguration。

list-license-configurations

以下代码示例演示了如何使用 list-license-configurations。

Amazon CLI

示例 1：列出所有许可证配置

以下 list-license-configurations 示例列出您的所有许可证配置。

```
aws license-manager list-license-configurations
```

输出：

```
{
    "LicenseConfigurations": [
        {
            "LicenseConfigurationId": "lic-6eb6586f508a786a2ba4f56c1EXAMPLE",
            "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE",
            "Name": "my-license-configuration",
            "LicenseCountingType": "Core",
            "LicenseRules": [],
            "LicenseCount": 10,
            "LicenseCountHardLimit": true,
            "ConsumedLicenses": 0,
            "Status": "AVAILABLE",
            "OwnerAccountId": "123456789012",
            "ConsumedLicenseSummaryList": [
                {
                    "ResourceType": "EC2_INSTANCE",
                    "ConsumedLicenses": 0
                },
```

```
                {
                    "ResourceType": "EC2_HOST",
                    "ConsumedLicenses": 0
                },
                {
                    "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
                    "ConsumedLicenses": 0
                }
            ],
            "ManagedResourceSummaryList": [
                {
                    "ResourceType": "EC2_INSTANCE",
                    "AssociationCount": 0
                },
                {
                    "ResourceType": "EC2_HOST",
                    "AssociationCount": 0
                },
                {
                    "ResourceType": "EC2_AMI",
                    "AssociationCount": 0
                },
                {
                    "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
                    "AssociationCount": 0
                }
            ]
        },
        {
            ...
        }
    ]
}
```

示例 2：列出特定许可证配置

以下 `list-license-configurations` 示例仅列出指定许可证配置。

```
aws license-manager list-license-configurations \
    --license-configuration-arns arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLicenseConfigurations。

list-license-specifications-for-resource

以下代码示例演示了如何使用 list-license-specifications-for-resource。

Amazon CLI

列出资源的许可证配置

以下 list-license-specifications-for-resource 示例列出与指定亚马逊机器映像（AMI）关联的许可证配置。

```
aws license-manager list-license-specifications-for-resource \
    --resource-arn arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0
```

输出：

```
{
    "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLicenseSpecificationsForResource。

list-resource-inventory

以下代码示例演示了如何使用 list-resource-inventory。

Amazon CLI

列出资源库存中的资源

以下 list-resource-inventory 示例列出使用 Systems Manager 库存托管的资源。

```
aws license-manager list-resource-inventory
```

输出：

```
{
    "ResourceInventoryList": [
        {
            "Platform": "Red Hat Enterprise Linux Server",
```

```
            "ResourceType": "EC2Instance",
            "PlatformVersion": "7.4",
            "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/
i-05d3cdfb05bd36376",
            "ResourceId": "i-05d3cdfb05bd36376",
            "ResourceOwningAccountId": "1234567890129"
        },
        {
            "Platform": "Amazon Linux",
            "ResourceType": "EC2Instance",
            "PlatformVersion": "2",
            "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/
i-0b1d036cfd4594808",
            "ResourceId": "i-0b1d036cfd4594808",
            "ResourceOwningAccountId": "1234567890129"
        },
        {
            "Platform": "Microsoft Windows Server 2019 Datacenter",
            "ResourceType": "EC2Instance",
            "PlatformVersion": "10.0.17763",
            "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/
i-0cdb3b54a2a8246ad",
            "ResourceId": "i-0cdb3b54a2a8246ad",
            "ResourceOwningAccountId": "1234567890129"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceInventory。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出许可证配置的标签

以下 list-tags-for-resource 示例列出指定许可证配置的标签。

```
aws license-manager list-tags-for-resource \
    --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

输出：

```
{
    "Tags": [
        {
            "Key": "project",
            "Value": "lima"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-usage-for-license-configuration

以下代码示例演示了如何使用 list-usage-for-license-configuration。

Amazon CLI

列出用于许可证配置的许可证

以下 list-usage-for-license-configuration 示例列出将许可证用于指定许可证配置的资源的信息。例如，如果许可证类型为 vCPU，则任何实例将是每个 vCPU 使用一个许可证。

```
aws license-manager list-usage-for-license-configuration \
    --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

输出：

```
{
    "LicenseConfigurationUsageList": [
        {
            "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-04a636d18e83cfacb",
            "ResourceType": "EC2_INSTANCE",
            "ResourceStatus": "running",
            "ResourceOwnerId": "123456789012",
            "AssociationTime": 1570892850.519,
            "ConsumedLicenses": 2
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsageForLicenseConfiguration。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为许可证配置添加标签

以下 tag-resource 示例为指定许可证配置添加指定标签（键名称和值）。

```
aws license-manager tag-resource \
    --tags Key=project,Value=lima \
    --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从许可证配置中移除标签

以下 untag-resource 示例从指定许可证配置中移除指定标签（键名称和值）。

```
aws license-manager untag-resource \
    --tag-keys project \
    --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-license-configuration

以下代码示例演示了如何使用 update-license-configuration。

Amazon CLI

更新许可证配置

以下 update-license-configuration 示例更新了指定的许可证配置以移除硬限制。

```
aws license-manager update-license-configuration \
    --no-license-count-hard-limit \
    --license-configuration-arn arn:aws:license-manager:us-
west-2:880185128111:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

此命令不生成任何输出。

以下 update-license-configuration 示例更新了指定的许可证配置以将其状态更改为 DISABLED。

```
aws license-manager update-license-configuration \
    --license-configuration-status DISABLED
    --license-configuration-arn arn:aws:license-manager:us-
west-2:880185128111:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLicenseConfiguration。

update-license-specifications-for-resource

以下代码示例演示了如何使用 update-license-specifications-for-resource。

Amazon CLI

更新资源的许可证配置

以下 update-license-specifications-for-resource 示例通过移除一个许可证配置并添加另一个来替换与指定亚马逊机器映像（AMI）关联的许可证配置。

```
aws license-manager update-license-specifications-for-resource \
    --resource-arn arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0 \
```

```
    --remove-license-specifications LicenseConfigurationArn=arn:aws:license-
manager:us-west-2:123456789012:license-
configuration:lic-38b658717b87478aaa7c00883EXAMPLE \
    --add-license-specifications LicenseConfigurationArn=arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-42b6deb06e5399a980d555927EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLicenseSpecificationsForResource。

update-service-settings

以下代码示例演示了如何使用 update-service-settings。

Amazon CLI

更新 License Manager 设置

以下 update-service-settings 示例为当前 Amazon 区域的 License Manager 启用跨账户资源发现。Amazon S3 存储桶是 Systems Manager 库存所需的资源数据同步。

```
aws license-manager update-service-settings \
    --organization-configuration EnableIntegration=true \
    --enable-cross-accounts-discovery \
    --s3-bucket-arn arn:aws:s3:::aws-license-manager-service-abcd1234EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServiceSettings。

Lightsail使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Lightsail 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

allocate-static-ip

以下代码示例演示了如何使用 allocate-static-ip。

Amazon CLI

创建静态 IP

以下 allocate-static-ip 示例创建可以附加到实例的指定静态 IP。

```
aws lightsail allocate-static-ip \
    --static-ip-name StaticIp-1
```

输出：

```
{
    "operations": [
        {
            "id": "b5d06d13-2f19-4683-889f-dEXAMPLEed79",
            "resourceName": "StaticIp-1",
            "resourceType": "StaticIp",
            "createdAt": 1571071325.076,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "AllocateStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571071325.274
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateStaticIp。

attach-disk

以下代码示例演示了如何使用 attach-disk。

Amazon CLI

将块存储磁盘附加到实例

以下 attach-disk 示例将磁盘 Disk-1 附加实例 WordPress_Multisite-1，磁盘路径为 /dev/xvdf

```
aws lightsail attach-disk \
    --disk-name Disk-1 \
    --disk-path /dev/xvdf \
    --instance-name WordPress_Multisite-1
```

输出：

```
{
    "operations": [
        {
            "id": "10a08267-19ce-43be-b913-6EXAMPLE7e80",
            "resourceName": "Disk-1",
            "resourceType": "Disk",
            "createdAt": 1571071465.472,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "WordPress_Multisite-1",
            "operationType": "AttachDisk",
            "status": "Started",
            "statusChangedAt": 1571071465.472
        },
        {
            "id": "2912c477-5295-4539-88c9-bEXAMPLEd1f0",
            "resourceName": "WordPress_Multisite-1",
            "resourceType": "Instance",
            "createdAt": 1571071465.474,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
```

```
            },
            "isTerminal": false,
            "operationDetails": "Disk-1",
            "operationType": "AttachDisk",
            "status": "Started",
            "statusChangedAt": 1571071465.474
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachDisk。

attach-instances-to-load-balancer

以下代码示例演示了如何使用 attach-instances-to-load-balancer。

Amazon CLI

将实例附加到负载均衡器

以下 attach-instances-to-load-balancer 示例将实例 MEAN-1、MEAN-2 和 MEAN-3 附加到负载均衡器 LoadBalancer-1。

```
aws lightsail attach-instances-to-load-balancer \
    --instance-names {"MEAN-1","MEAN-2","MEAN-3"} \
    --load-balancer-name LoadBalancer-1
```

输出：

```
{
    "operations": [
        {
            "id": "8055d19d-abb2-40b9-b527-1EXAMPLE3c7b",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571071699.892,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "MEAN-2",
```

```
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.892
        },
        {
            "id": "c35048eb-8538-456a-a118-0EXAMPLEfb73",
            "resourceName": "MEAN-2",
            "resourceType": "Instance",
            "createdAt": 1571071699.887,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "LoadBalancer-1",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.887
        },
        {
            "id": "910d09e0-adc5-4372-bc2e-0EXAMPLEd891",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571071699.882,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "MEAN-3",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.882
        },
        {
            "id": "178b18ac-43e8-478c-9bed-1EXAMPLE4755",
            "resourceName": "MEAN-3",
            "resourceType": "Instance",
            "createdAt": 1571071699.901,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
```

```
            "operationDetails": "LoadBalancer-1",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.901
        },
        {
            "id": "fb62536d-2a98-4190-a6fc-4EXAMPLE7470",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571071699.885,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "MEAN-1",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.885
        },
        {
            "id": "787dac0d-f98d-46c3-8571-3EXAMPLE5a85",
            "resourceName": "MEAN-1",
            "resourceType": "Instance",
            "createdAt": 1571071699.901,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "LoadBalancer-1",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1571071699.901
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachInstancesToLoadBalancer。

attach-load-balancer-tls-certificate

以下代码示例演示了如何使用 attach-load-balancer-tls-certificate。

Amazon CLI

将 TLS 证书附加到负载均衡器

以下 attach-load-balancer-tls-certificate 示例将负载均衡器 TLS 证书 Certificate2 附加到负载均衡器 LoadBalancer-1。

```
aws lightsail attach-load-balancer-tls-certificate \
    --certificate-name Certificate2 \
    --load-balancer-name LoadBalancer-1
```

输出：

```
{
    "operations": [
        {
            "id": "cf1ad6e3-3cbb-4b8a-a7f2-3EXAMPLEa118",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571072255.416,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "Certificate2",
            "operationType": "AttachLoadBalancerTlsCertificate",
            "status": "Succeeded",
            "statusChangedAt": 1571072255.416
        },
        {
            "id": "dae1bcfb-d531-4c06-b4ea-bEXAMPLEc04e",
            "resourceName": "Certificate2",
            "resourceType": "LoadBalancerTlsCertificate",
            "createdAt": 1571072255.416,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "LoadBalancer-1",
            "operationType": "AttachLoadBalancerTlsCertificate",
            "status": "Succeeded",
```

```
            "statusChangedAt": 1571072255.416
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachLoadBalancerTlsCertificate。

attach-static-ip

以下代码示例演示了如何使用 attach-static-ip。

Amazon CLI

将静态 IP 附加到实例

以下 attach-static-ip 示例将静态 IP StaticIp-1 附加到实例 MEAN-1。

```
aws lightsail attach-static-ip \
    --static-ip-name StaticIp-1 \
    --instance-name MEAN-1
```

输出：

```
{
    "operations": [
        {
            "id": "45e6fa13-4808-4b8d-9292-bEXAMPLE20b2",
            "resourceName": "StaticIp-1",
            "resourceType": "StaticIp",
            "createdAt": 1571072569.375,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MEAN-1",
            "operationType": "AttachStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571072569.375
        },
        {
            "id": "9ee09a17-863c-4e51-8a6d-3EXAMPLE5475",
```

```
            "resourceName": "MEAN-1",
            "resourceType": "Instance",
            "createdAt": 1571072569.376,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "StaticIp-1",
            "operationType": "AttachStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571072569.376
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachStaticIp。

close-instance-public-ports

以下代码示例演示了如何使用 close-instance-public-ports。

Amazon CLI

关闭实例的防火墙端口

以下 close-instance-public-ports 示例关闭实例 MEAN-2 上的 TCP 端口 22。

```
aws lightsail close-instance-public-ports \
    --instance-name MEAN-2 \
    --port-info fromPort=22,protocol=TCP,toPort=22
```

输出：

```
{
    "operation": {
        "id": "4f328636-1c96-4649-ae6d-1EXAMPLEf446",
        "resourceName": "MEAN-2",
        "resourceType": "Instance",
        "createdAt": 1571072845.737,
        "location": {
            "availabilityZone": "us-west-2a",
```

```
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationDetails": "22/tcp",
        "operationType": "CloseInstancePublicPorts",
        "status": "Succeeded",
        "statusChangedAt": 1571072845.737
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CloseInstancePublicPorts。

copy-snapshot

以下代码示例演示了如何使用 copy-snapshot。

Amazon CLI

示例 1：在同一 Amazon 区域中复制快照

以下 copy-snapshot 示例在同一 Amazon 区域 us-west-2 中将实例快照 MEAN-1-1571075291 复制为实例快照 MEAN-1-Copy。

```
aws lightsail copy-snapshot \
    --source-snapshot-name MEAN-1-1571075291 \
    --target-snapshot-name MEAN-1-Copy \
    --source-region us-west-2
```

输出：

```
{
    "operations": [
        {
            "id": "ced16fc1-f401-4556-8d82-1EXAMPLEb982",
            "resourceName": "MEAN-1-Copy",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1571075581.498,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
```

```
            "isTerminal": false,
            "operationDetails": "us-west-2:MEAN-1-1571075291",
            "operationType": "CopySnapshot",
            "status": "Started",
            "statusChangedAt": 1571075581.498
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中将快照从一个 Amazon 区域复制到另一个区域。

示例 2：将快照从一个 Amazon 区域复制到另一个区域

以下 copy-snapshot 示例将 Amazon 区域 us-west-2 中的实例快照 MEAN-1-1571075291 复制为 us-east-1 中的实例快照 MEAN-1-1571075291-Copy。

```
aws lightsail copy-snapshot \
    --source-snapshot-name MEAN-1-1571075291 \
    --target-snapshot-name MEAN-1-1571075291-Copy \
    --source-region us-west-2 \
    --region us-east-1
```

输出：

```
{
    "operations": [
        {
            "id": "91116b79-119c-4451-b44a-dEXAMPLEd97b",
            "resourceName": "MEAN-1-1571075291-Copy",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1571075695.069,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-east-1"
            },
            "isTerminal": false,
            "operationDetails": "us-west-2:MEAN-1-1571075291",
            "operationType": "CopySnapshot",
            "status": "Started",
            "statusChangedAt": 1571075695.069
        }
```

```
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中将快照从一个 Amazon 区域复制到另一个区域。

示例 3：在同一个 Amazon 区域内复制自动快照

以下 copy-snapshot 示例将 Amazon 区域 us-west-2 中的实例 WordPress-1 的自动快照 2019-10-14 复制为手动快照 WordPress-1-10142019。

```
aws lightsail copy-snapshot \
    --source-resource-name WordPress-1 \
    --restore-date 2019-10-14 \
    --target-snapshot-name WordPress-1-10142019 \
    --source-region us-west-2
```

输出：

```
{
    "operations": [
        {
            "id": "be3e6754-cd1d-48e6-ad9f-2EXAMPLE1805",
            "resourceName": "WordPress-1-10142019",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1571082412.311,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "us-west-2:WordPress-1",
            "operationType": "CopySnapshot",
            "status": "Started",
            "statusChangedAt": 1571082412.311
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中保留实例或磁盘的自动快照。

示例 4：将自动快照从一个 Amazon 区域复制到另一个区域

以下 copy-snapshot 示例将 Amazon 区域 us-west-2 中的实例 WordPress-1 的自动快照 2019-10-14 复制为 us-east-1 中的手动快照 WordPress-1-10142019。

```
aws lightsail copy-snapshot \
    --source-resource-name WordPress-1 \
    --restore-date 2019-10-14 \
    --target-snapshot-name WordPress-1-10142019 \
    --source-region us-west-2 \
    --region us-east-1
```

输出：

```
{
    "operations": [
        {
            "id": "dffa128b-0b07-476e-b390-bEXAMPLE3775",
            "resourceName": "WordPress-1-10142019",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1571082493.422,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-east-1"
            },
            "isTerminal": false,
            "operationDetails": "us-west-2:WordPress-1",
            "operationType": "CopySnapshot",
            "status": "Started",
            "statusChangedAt": 1571082493.422
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中保留实例或磁盘的自动快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopySnapshot。

create-disk-from-snapshot

以下代码示例演示了如何使用 create-disk-from-snapshot。

Amazon CLI

根据磁盘快照创建磁盘

以下 create-disk-from-snapshot 示例根据指定的块存储磁盘快照创建了一个名为 Disk-2 的块存储磁盘。该磁盘在指定的 Amazon 区域和可用区中创建，存储空间为 32 GB。

```
aws lightsail create-disk-from-snapshot \
    --disk-name Disk-2 \
    --disk-snapshot-name Disk-1-1566839161 \
    --availability-zone us-west-2a \
    --size-in-gb 32
```

输出：

```
{
    "operations": [
        {
            "id": "d42b605d-5ef1-4b4a-8791-7a3e8b66b5e7",
            "resourceName": "Disk-2",
            "resourceType": "Disk",
            "createdAt": 1569624941.471,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateDiskFromSnapshot",
            "status": "Started",
            "statusChangedAt": 1569624941.791
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中根据快照创建块存储磁盘。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDiskFromSnapshot。

create-disk-snapshot

以下代码示例演示了如何使用 create-disk-snapshot。

Amazon CLI

示例 1：创建磁盘快照

以下 create-disk-snapshot 示例创建指定块存储磁盘的名为 DiskSnapshot-1 的快照。

```
aws lightsail create-disk-snapshot \
    --disk-name Disk-1 \
    --disk-snapshot-name DiskSnapshot-1
```

输出：

```
{
    "operations": [
        {
            "id": "fa74c6d2-03a3-4f42-a7c7-792f124d534b",
            "resourceName": "DiskSnapshot-1",
            "resourceType": "DiskSnapshot",
            "createdAt": 1569625129.739,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "Disk-1",
            "operationType": "CreateDiskSnapshot",
            "status": "Started",
            "statusChangedAt": 1569625129.739
        },
        {
            "id": "920a25df-185c-4528-87cd-7b85f5488c06",
            "resourceName": "Disk-1",
            "resourceType": "Disk",
            "createdAt": 1569625129.739,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "DiskSnapshot-1",
            "operationType": "CreateDiskSnapshot",
            "status": "Started",
            "statusChangedAt": 1569625129.739
```

```
        }
    ]
}
```

示例 2：创建实例的系统磁盘的快照

以下 create-disk-snapshot 示例创建指定实例的系统磁盘的快照。

```
aws lightsail create-disk-snapshot \
    --instance-name WordPress-1 \
    --disk-snapshot-name SystemDiskSnapshot-1
```

输出：

```
{
    "operations": [
        {
            "id": "f508cf1c-6597-42a6-a4c3-4aebd75af0d9",
            "resourceName": "SystemDiskSnapshot-1",
            "resourceType": "DiskSnapshot",
            "createdAt": 1569625294.685,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "WordPress-1",
            "operationType": "CreateDiskSnapshot",
            "status": "Started",
            "statusChangedAt": 1569625294.685
        },
        {
            "id": "0bb9f712-da3b-4d99-b508-3bf871d989e5",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1569625294.685,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "SystemDiskSnapshot-1",
            "operationType": "CreateDiskSnapshot",
```

```
            "status": "Started",
            "statusChangedAt": 1569625294.685
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的 Amazon Lightsail 中的快照和在 Amazon Lightsail 中创建实例根卷的快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDiskSnapshot。

create-disk

以下代码示例演示了如何使用 create-disk。

Amazon CLI

创建块存储磁盘

以下 create-disk 示例在指定的 Amazon 区域和可用区中创建块存储磁盘 Disk-1，存储空间为 32 GB。

```
aws lightsail create-disk \
    --disk-name Disk-1 \
    --availability-zone us-west-2a \
    --size-in-gb 32
```

输出：

```
{
    "operations": [
        {
            "id": "1c85e2ec-86ba-4697-b936-77f4d3dc013a",
            "resourceName": "Disk-1",
            "resourceType": "Disk",
            "createdAt": 1569449220.36,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateDisk",
```

```
            "status": "Started",
            "statusChangedAt": 1569449220.588
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDisk。

create-domain-entry

以下代码示例演示了如何使用 create-domain-entry。

Amazon CLI

创建域条目（DNS 记录）

以下 create-domain-entry 示例为指定域的顶点创建指向实例 IP 地址的 DNS 记录（A）。

注意：Lightsail 的域相关 API 操作仅在 us-east-1 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含 --region us-east-1 参数，否则命令将失败。

```
aws lightsail create-domain-entry \
    --region us-east-1 \
    --domain-name example.com \
    --domain-entry name=example.com,type=A,target=192.0.2.0
```

输出：

```
{
    "operation": {
        "id": "5be4494d-56f4-41fc-8730-693dcd0ef9e2",
        "resourceName": "example.com",
        "resourceType": "Domain",
        "createdAt": 1569865296.519,
        "location": {
            "availabilityZone": "all",
            "regionName": "global"
        },
        "isTerminal": true,
        "operationType": "CreateDomainEntry",
        "status": "Succeeded",
```

```
        "statusChangedAt": 1569865296.519
    }
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的 Amazon Lightsail 中的 DNS 和在 Amazon Lightsail 中创建 DNS 区域来管理域的 DNS 记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomainEntry。

`create-domain`

以下代码示例演示了如何使用 `create-domain`。

Amazon CLI

创建域（DNS 区域）

以下 `create-domain` 示例为指定域创建 DNS 区域。

注意：Lightsail 的域相关 API 操作仅在 `us-east-1` 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含 `--region us-east-1` 参数，否则命令将失败。

```
aws lightsail create-domain \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "operation": {
        "id": "64e522c8-9ae1-4c05-9b65-3f237324dc34",
        "resourceName": "example.com",
        "resourceType": "Domain",
        "createdAt": 1569864291.92,
        "location": {
            "availabilityZone": "all",
            "regionName": "global"
        },
        "isTerminal": true,
        "operationType": "CreateDomain",
        "status": "Succeeded",
        "statusChangedAt": 1569864292.109
```

```
    }
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的 Amazon Lightsail 中的 DNS 和在 Amazon Lightsail 中创建 DNS 区域来管理域的 DNS 记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDomain。

`create-instance-snapshot`

以下代码示例演示了如何使用 `create-instance-snapshot`。

Amazon CLI

创建实例的快照

以下 `create-instance-snapshot` 示例根据指定实例创建快照。

```
aws lightsail create-instance-snapshot \
    --instance-name WordPress-1 \
    --instance-snapshot-name WordPress-Snapshot-1
```

输出：

```
{
    "operations": [
        {
            "id": "4c3db559-9dd0-41e7-89c0-2cb88c19786f",
            "resourceName": "WordPress-Snapshot-1",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1569866438.48,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "WordPress-1",
            "operationType": "CreateInstanceSnapshot",
            "status": "Started",
            "statusChangedAt": 1569866438.48
        },
        {
```

```
            "id": "c04fdc45-2981-488c-88b5-d6d2fd759a6a",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1569866438.48,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "WordPress-Snapshot-1",
            "operationType": "CreateInstanceSnapshot",
            "status": "Started",
            "statusChangedAt": 1569866438.48
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstanceSnapshot。

create-instances-from-snapshot

以下代码示例演示了如何使用 create-instances-from-snapshot。

Amazon CLI

根据快照创建实例

以下 create-instances-from-snapshot 示例在指定的 Amazon 区域和可用区中，使用价值 12 美元的捆绑包，根据指定实例快照创建了一个实例。

注意：您指定的捆绑包的规格必须等于或大于用于创建快照的原始源实例的捆绑包。

```
aws lightsail create-instances-from-snapshot \
    --instance-snapshot-name WordPress-1-1569866208 \
    --instance-names WordPress-2 \
    --availability-zone us-west-2a \
    --bundle-id small_3_0
```

输出：

```
{
    "operations": [
```

```
        {
            "id": "003f8271-b711-464d-b9b8-7f3806cb496e",
            "resourceName": "WordPress-2",
            "resourceType": "Instance",
            "createdAt": 1569865914.908,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateInstancesFromSnapshot",
            "status": "Started",
            "statusChangedAt": 1569865914.908
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstancesFromSnapshot。

create-instances

以下代码示例演示了如何使用 create-instances。

Amazon CLI

示例 1：创建单个实例

以下 create-instances 示例在指定的 Amazon 区域和可用区中，使用 WordPress 蓝图和价值 5 美元的捆绑包创建了一个实例。

```
aws lightsail create-instances \
    --instance-names Instance-1 \
    --availability-zone us-west-2a \
    --blueprint-id wordpress \
    --bundle-id nano_3_0
```

输出：

```
{
    "operations": [
        {
            "id": "9a77158f-7be3-4d6d-8054-cf5ae2b720cc",
```

```
            "resourceName": "Instance-1",
            "resourceType": "Instance",
            "createdAt": 1569447986.061,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateInstance",
            "status": "Started",
            "statusChangedAt": 1569447986.061
        }
    ]
}
```

示例 2：一次创建多个实例

以下 `create-instances` 示例在指定的 Amazon 区域和可用区中，使用 WordPress 蓝图和价值 5 美元的捆绑包创建了三个实例。

```
aws lightsail create-instances \
    --instance-names {"Instance1","Instance2","Instance3"} \
    --availability-zone us-west-2a \
    --blueprint-id wordpress \
    --bundle-id nano_3_0
```

输出：

```
{
    "operations": [
        {
            "id": "5492f015-9d2e-48c6-8eea-b516840e6903",
            "resourceName": "Instance1",
            "resourceType": "Instance",
            "createdAt": 1569448780.054,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateInstance",
            "status": "Started",
            "statusChangedAt": 1569448780.054
```

```
        },
        {
            "id": "c58b5f46-2676-44c8-b95c-3ad375898515",
            "resourceName": "Instance2",
            "resourceType": "Instance",
            "createdAt": 1569448780.054,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateInstance",
            "status": "Started",
            "statusChangedAt": 1569448780.054
        },
        {
            "id": "a5ad8006-9bee-4499-9eb7-75e42e6f5882",
            "resourceName": "Instance3",
            "resourceType": "Instance",
            "createdAt": 1569448780.054,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateInstance",
            "status": "Started",
            "statusChangedAt": 1569448780.054
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInstances。

create-key-pair

以下代码示例演示了如何使用 create-key-pair。

Amazon CLI

创建密钥对

以下 create-key-pair 示例创建可用于身份验证进而连接到实例的密钥对。

```
aws lightsail create-key-pair \
    --key-pair-name MyPersonalKeyPair
```

输出提供了私钥 base64 值，您可以使用该值进行身份验证，进而连接使用创建的密钥对的实例。注意：请将私钥 base64 值复制粘贴到一个安全位置，因为以后您无法检索它。

```
{
    "keyPair": {
        "name": "MyPersonalKeyPair",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:KeyPair/55025c71-198f-403b-
b42f-a69433e724fb",
        "supportCode": "621291663362/MyPersonalKeyPair",
        "createdAt": 1569866556.567,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "KeyPair"
    },
    "publicKeyBase64": "ssh-rsa ssh-rsa
 AAAAB3NzaC1yc2EAAAADAQABAAABAQCV0xUEwx96amPERH7K1bVT1tTFl9OmNk6o7m5YVHk9xlOdMbDRbFvhtXvw4jz
+BHUgedGUXno6uF7agqxZNO1kPLJBIVTW26SSYBJ0tE
+y804UyVsjrbUqCaMXDhmfXpWulMPwuXhwcKh7e8hwoTfkiX0E6Ql
+KqF/MiA3w6DCjEqvvdIO7SiEZJFsuGNfYDDN3w60Rel5MUhmn3OJdn4y/
A7NWb3IxL4pPfVE4rgFRKU8n1jp9kwRnlVMVBOWuGXk6n+H6M2f1 ",
    "privateKeyBase64": "-----BEGIN RSA PRIVATE KEY-----
EXAMPLETCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
\nVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6\nb24xFDASBgNVBAsTC0lBTSBD
\nBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
\nMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
\nVQQHEwdTZWF0dGxlMQ8wDQEXAMPLEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
\nb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
\nYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMEXAMPLE4GmWIWJ
\n21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
\nrDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
\nIbb3OhjZnzcvQAaREXAMPLEMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4\nnUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
\nFFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OEXAMPLELvjx79LjSTb
\nNYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=\n-----END RSA PRIVATE KEY-----",
    "operation": {
        "id": "67f984db-9994-45fe-ad38-59bafcaf82ef",
        "resourceName": "MyPersonalKeyPair",
        "resourceType": "KeyPair",
```

```
        "createdAt": 1569866556.567,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationType": "CreateKeyPair",
        "status": "Succeeded",
        "statusChangedAt": 1569866556.704
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKeyPair。

create-load-balancer-tls-certificate

以下代码示例演示了如何使用 create-load-balancer-tls-certificate。

Amazon CLI

为负载均衡器创建 TLS 证书

以下 create-load-balancer-tls-certificate 示例创建一个附加到指定负载均衡器的 TLS 证书。创建的证书适用于指定域。注意：只能为一个负载均衡器创建两个证书。

```
aws lightsail create-load-balancer-tls-certificate \
    --certificate-alternative-names abc.example.com \
    --certificate-domain-name example.com \
    --certificate-name MySecondCertificate \
    --load-balancer-name MyFirstLoadBalancer
```

输出：

```
{
    "operations": [
        {
            "id": "be663aed-cb46-41e2-9b23-e2f747245bd4",
            "resourceName": "MySecondCertificate",
            "resourceType": "LoadBalancerTlsCertificate",
            "createdAt": 1569867364.971,
            "location": {
                "availabilityZone": "all",
```

```
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MyFirstLoadBalancer",
            "operationType": "CreateLoadBalancerTlsCertificate",
            "status": "Succeeded",
            "statusChangedAt": 1569867365.219
        },
        {
            "id": "f3dfa930-969e-41cc-ac7d-337178716f6d",
            "resourceName": "MyFirstLoadBalancer",
            "resourceType": "LoadBalancer",
            "createdAt": 1569867364.971,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MySecondCertificate",
            "operationType": "CreateLoadBalancerTlsCertificate",
            "status": "Succeeded",
            "statusChangedAt": 1569867365.219
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancerTlsCertificate。

create-load-balancer

以下代码示例演示了如何使用 create-load-balancer。

Amazon CLI

创建负载均衡器

以下 create-load-balancer 示例创建具有一个 TLS 证书的负载均衡器。该 TLS 证书适用于指定域，并将流量路由到端口 80 上的实例。

```
aws lightsail create-load-balancer \
    --certificate-alternative-names www.example.com test.example.com \
    --certificate-domain-name example.com \
    --certificate-name Certificate-1 \
```

```
    --instance-port 80 \
    --load-balancer-name LoadBalancer-1
```

输出：

```
{
    "operations": [
        {
            "id": "cc7b920a-83d8-4762-a74e-9174fe1540be",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1569867169.406,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateLoadBalancer",
            "status": "Started",
            "statusChangedAt": 1569867169.406
        },
        {
            "id": "658ed43b-f729-42f3-a8e4-3f8024d3c98d",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancerTlsCertificate",
            "createdAt": 1569867170.193,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "LoadBalancer-1",
            "operationType": "CreateLoadBalancerTlsCertificate",
            "status": "Succeeded",
            "statusChangedAt": 1569867170.54
        },
        {
            "id": "4757a342-5181-4870-b1e0-227eebc35ab5",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1569867170.193,
            "location": {
                "availabilityZone": "all",
```

```
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "Certificate-1",
            "operationType": "CreateLoadBalancerTlsCertificate",
            "status": "Succeeded",
            "statusChangedAt": 1569867170.54
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的 Lightsail 负载均衡器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLoadBalancer。

create-relational-database-from-snapshot

以下代码示例演示了如何使用 create-relational-database-from-snapshot。

Amazon CLI

根据快照创建托管数据库

以下 create-relational-database-from-snapshot 示例在指定的 Amazon 区域和可用区中，使用价值 15 美元的标准数据库捆绑包，根据指定快照创建了一个托管数据库。注意：您指定的捆绑包的规格必须等于或大于用于创建快照的原始源实例的捆绑包。

```
aws lightsail create-relational-database-from-snapshot \
    --relational-database-snapshot-name Database-Oregon-1-1566839359 \
    --relational-database-name Database-1 \
    --availability-zone us-west-2a \
    --relational-database-bundle-id micro_1_0 \
    --no-publicly-accessible
```

输出：

```
{
    "operations": [
        {
            "id": "ad6d9193-9d5c-4ea1-97ae-8fe6de600b4c",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
```

```
            "createdAt": 1569867916.938,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateRelationalDatabaseFromSnapshot",
            "status": "Started",
            "statusChangedAt": 1569867918.643
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRelationalDatabaseFromSnapshot。

create-relational-database-snapshot

以下代码示例演示了如何使用 create-relational-database-snapshot。

Amazon CLI

创建托管数据库的快照

以下 create-relational-database-snapshot 示例创建指定托管数据库的快照。

```
aws lightsail create-relational-database-snapshot \
    --relational-database-name Database1 \
    --relational-database-snapshot-name RelationalDatabaseSnapshot1
```

输出：

```
{
    "operations": [
        {
            "id": "853667fb-ea91-4c02-8d20-8fc5fd43b9eb",
            "resourceName": "RelationalDatabaseSnapshot1",
            "resourceType": "RelationalDatabaseSnapshot",
            "createdAt": 1569868074.645,
            "location": {
                "availabilityZone": "all",
```

```
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "Database1",
            "operationType": "CreateRelationalDatabaseSnapshot",
            "status": "Started",
            "statusChangedAt": 1569868074.645
        },
        {
            "id": "fbafa521-3cac-4be8-9773-1c143780b239",
            "resourceName": "Database1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1569868074.645,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "RelationalDatabaseSnapshot1",
            "operationType": "CreateRelationalDatabaseSnapshot",
            "status": "Started",
            "statusChangedAt": 1569868074.645
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRelationalDatabaseSnapshot。

create-relational-database

以下代码示例演示了如何使用 create-relational-database。

Amazon CLI

创建托管数据库

以下 create-relational-database 示例在指定的 Amazon 区域和可用区内，使用 MySQL 5.6 数据库引擎（mysql_5_6）和价值 15 美元的标准数据库捆绑包（micro_1_0）创建了一个托管数据库。该托管数据库被预先填入了主用户名，因此不能公开访问。

```
aws lightsail create-relational-database \
```

```
    --relational-database-name Database-1 \
    --availability-zone us-west-2a \
    --relational-database-blueprint-id mysql_5_6 \
    --relational-database-bundle-id micro_1_0 \
    --master-database-name dbmaster \
    --master-username user \
    --no-publicly-accessible
```

输出：

```
{
    "operations": [
        {
            "id": "b52bedee-73ed-4798-8d2a-9c12df89adcd",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1569450017.244,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "CreateRelationalDatabase",
            "status": "Started",
            "statusChangedAt": 1569450018.637
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRelationalDatabase。

delete-auto-snapshot

以下代码示例演示了如何使用 delete-auto-snapshot。

Amazon CLI

删除自动快照

以下 delete-auto-snapshot 示例删除实例 WordPress-1 的自动快照 2019-10-10。

```
aws lightsail delete-auto-snapshot \
```

```
    --resource-name WordPress-1 \
    --date 2019-10-10
```

输出：

```
{
    "operations": [
        {
            "id": "31c36e09-3d52-46d5-b6d8-7EXAMPLE534a",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1571088141.501,
            "location": {
                "availabilityZone": "us-west-2",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "DeleteAutoSnapshot-2019-10-10",
            "operationType": "DeleteAutoSnapshot",
            "status": "Succeeded"
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中删除实例或磁盘的自动快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAutoSnapshot。

delete-disk-snapshot

以下代码示例演示了如何使用 delete-disk-snapshot。

Amazon CLI

删除块存储磁盘的快照

以下 delete-disk-snapshot 示例删除块存储磁盘的指定快照。

```
aws lightsail delete-disk-snapshot \
    --disk-snapshot-name DiskSnapshot-1
```

输出：

```
{
    "operations": [
        {
            "id": "d1e5766d-b81e-4595-ad5d-02afbccfcd5d",
            "resourceName": "DiskSnapshot-1",
            "resourceType": "DiskSnapshot",
            "createdAt": 1569873552.79,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteDiskSnapshot",
            "status": "Succeeded",
            "statusChangedAt": 1569873552.79
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDiskSnapshot。

delete-disk

以下代码示例演示了如何使用 delete-disk。

Amazon CLI

删除块存储磁盘

以下 delete-disk 示例删除指定块存储磁盘。

```
aws lightsail delete-disk \
    --disk-name Disk-1
```

输出：

```
{
    "operations": [
        {
```

```
            "id": "6378c70f-4d75-4f7a-ab66-730fca0bb2fc",
            "resourceName": "Disk-1",
            "resourceType": "Disk",
            "createdAt": 1569872887.864,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteDisk",
            "status": "Succeeded",
            "statusChangedAt": 1569872887.864
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDisk。

delete-domain-entry

以下代码示例演示了如何使用 delete-domain-entry。

Amazon CLI

删除域条目（DNS 记录）

以下 delete-domain-entry 示例从现有域中删除指定域条目。

注意：Lightsail 的域相关 API 操作仅在 us-east-1 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含 --region us-east-1 参数，否则命令将失败。

```
aws lightsail delete-domain-entry \
    --region us-east-1 \
    --domain-name example.com \
    --domain-entry name=123.example.com,target=192.0.2.0,type=A
```

输出：

```
{
    "operation": {
        "id": "06eacd01-d785-420e-8daa-823150c7dca1",
```

```
        "resourceName": "example.com ",
        "resourceType": "Domain",
        "createdAt": 1569874157.005,
        "location": {
            "availabilityZone": "all",
            "regionName": "global"
        },
        "isTerminal": true,
        "operationType": "DeleteDomainEntry",
        "status": "Succeeded",
        "statusChangedAt": 1569874157.005
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomainEntry。

delete-domain

以下代码示例演示了如何使用 delete-domain。

Amazon CLI

删除域（DNS 区域）

以下 delete-domain 示例删除指定域以及该域中的所有条目（DNS 记录）。

注意：Lightsail 的域相关 API 操作仅在 us-east-1 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含 --region us-east-1 参数，否则命令将失败。

```
aws lightsail delete-domain \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "operation": {
        "id": "fcef5265-5af1-4a46-a3d7-90b5e18b9b32",
        "resourceName": "example.com",
        "resourceType": "Domain",
        "createdAt": 1569873788.13,
        "location": {
```

```
            "availabilityZone": "all",
            "regionName": "global"
        },
        "isTerminal": true,
        "operationType": "DeleteDomain",
        "status": "Succeeded",
        "statusChangedAt": 1569873788.13
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDomain。

delete-instance-snapshot

以下代码示例演示了如何使用 delete-instance-snapshot。

Amazon CLI

删除实例快照

以下 delete-instance-snapshot 示例删除一个实例的指定快照。

```
aws lightsail delete-instance-snapshot \
    --instance-snapshot-name WordPress-1-Snapshot-1
```

输出：

```
{
    "operations": [
        {
            "id": "14dad182-976a-46c6-bfd4-9480482bf0ea",
            "resourceName": "WordPress-1-Snapshot-1",
            "resourceType": "InstanceSnapshot",
            "createdAt": 1569874524.562,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteInstanceSnapshot",
            "status": "Succeeded",
            "statusChangedAt": 1569874524.562
```

```
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInstanceSnapshot。

delete-instance

以下代码示例演示了如何使用 delete-instance。

Amazon CLI

删除实例

以下 delete-instance 示例删除指定实例。

```
aws lightsail delete-instance \
    --instance-name WordPress-1
```

输出：

```
{
    "operations": [
        {
            "id": "d77345a3-8f80-4d2e-b47d-aaa622718df2",
            "resourceName": "Disk-1",
            "resourceType": "Disk",
            "createdAt": 1569874357.469,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "WordPress-1",
            "operationType": "DetachDisk",
            "status": "Started",
            "statusChangedAt": 1569874357.469
        },
        {
            "id": "708fa606-2bfd-4e48-a2c1-0b856585b5b1",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
```

```
            "createdAt": 1569874357.465,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "Disk-1",
            "operationType": "DetachDisk",
            "status": "Started",
            "statusChangedAt": 1569874357.465
        },
        {
            "id": "3187e823-8acb-405d-b098-fad5ceb17bec",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1569874357.829,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteInstance",
            "status": "Succeeded",
            "statusChangedAt": 1569874357.829
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInstance。

delete-key-pair

以下代码示例演示了如何使用 delete-key-pair。

Amazon CLI

删除密钥对

以下 delete-key-pair 示例删除指定密钥对。

```
aws lightsail delete-key-pair \
    --key-pair-name MyPersonalKeyPair
```

输出：

```
{
    "operation": {
        "id": "81621463-df38-4810-b866-6e801a15abbf",
        "resourceName": "MyPersonalKeyPair",
        "resourceType": "KeyPair",
        "createdAt": 1569874626.466,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationType": "DeleteKeyPair",
        "status": "Succeeded",
        "statusChangedAt": 1569874626.685
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteKeyPair。

delete-known-host-keys

以下代码示例演示了如何使用 delete-known-host-keys。

Amazon CLI

从实例中删除已知的主机密钥

以下 delete-known-host-keys 示例从指定实例中删除已知的主机密钥。

```
aws lightsail delete-known-host-keys \
    --instance-name Instance-1
```

输出：

```
{
    "operations": [
        {
            "id": "c61afe9c-45a4-41e6-a97e-d212364da3f5",
            "resourceName": "Instance-1",
```

```
            "resourceType": "Instance",
            "createdAt": 1569874760.201,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteKnownHostKeys",
            "status": "Succeeded",
            "statusChangedAt": 1569874760.201
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的解决 Amazon Lightsail 基于浏览器的 SSH 或 RDP 客户端的连接问题。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteKnownHostKeys。

delete-load-balancer-tls-certificate

以下代码示例演示了如何使用 delete-load-balancer-tls-certificate。

Amazon CLI

删除负载均衡器的 TLS 证书

以下 delete-load-balancer-tls-certificate 示例从指定负载均衡器中删除指定 TLS 证书。

```
aws lightsail delete-load-balancer-tls-certificate \
    --load-balancer-name MyFirstLoadBalancer \
    --certificate-name MyFirstCertificate
```

输出：

```
{
    "operations": [
        {
            "id": "50bec274-e45e-4caa-8a69-b763ef636583",
            "resourceName": "MyFirstCertificate",
```

```
            "resourceType": "LoadBalancerTlsCertificate",
            "createdAt": 1569874989.48,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "DeleteLoadBalancerTlsCertificate",
            "status": "Started",
            "statusChangedAt": 1569874989.48
        },
        {
            "id": "78c58cdc-a59a-4b27-8213-500638634a8f",
            "resourceName": "MyFirstLoadBalancer",
            "resourceType": "LoadBalancer",
            "createdAt": 1569874989.48,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "DeleteLoadBalancerTlsCertificate",
            "status": "Started",
            "statusChangedAt": 1569874989.48
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancerTlsCertificate。

delete-load-balancer

以下代码示例演示了如何使用 delete-load-balancer。

Amazon CLI

删除负载均衡器

以下 delete-load-balancer 示例删除指定的负载均衡器以及任何关联的 TLS 证书。

```
aws lightsail delete-load-balancer \
    --load-balancer-name MyFirstLoadBalancer
```

输出：


```
{
    "operations": [
        {
            "id": "a8c968c7-72a3-4680-a714-af8f03eea535",
            "resourceName": "MyFirstLoadBalancer",
            "resourceType": "LoadBalancer",
            "createdAt": 1569875092.125,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1569875092.125
        },
        {
            "id": "f91a29fc-8ce3-4e69-a227-ea70ca890bf5",
            "resourceName": "MySecondCertificate",
            "resourceType": "LoadBalancerTlsCertificate",
            "createdAt": 1569875091.938,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "DeleteLoadBalancerTlsCertificate",
            "status": "Started",
            "statusChangedAt": 1569875091.938
        },
        {
            "id": "cf64c060-154b-4eb4-ba57-84e2e41563d6",
            "resourceName": "MyFirstLoadBalancer",
            "resourceType": "LoadBalancer",
            "createdAt": 1569875091.94,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "DeleteLoadBalancerTlsCertificate",
            "status": "Started",
```

```
            "statusChangedAt": 1569875091.94
        }
    ]
}
```

有关更多信息，请参阅《指南》中相应标题下的内容。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoadBalancer。

delete-relational-database-snapshot

以下代码示例演示了如何使用 delete-relational-database-snapshot。

Amazon CLI

删除托管数据库的快照

以下 delete-relational-database-snapshot 示例删除托管数据库的指定快照。

```
aws lightsail delete-relational-database-snapshot \
    --relational-database-snapshot-name Database-Oregon-1-1566839359
```

输出：

```
{
    "operations": [
        {
            "id": "b99acae8-735b-4823-922f-30af580e3729",
            "resourceName": "Database-Oregon-1-1566839359",
            "resourceType": "RelationalDatabaseSnapshot",
            "createdAt": 1569875293.58,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "DeleteRelationalDatabaseSnapshot",
            "status": "Succeeded",
            "statusChangedAt": 1569875293.58
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRelationalDatabaseSnapshot。

delete-relational-database

以下代码示例演示了如何使用 delete-relational-database。

Amazon CLI

删除托管数据库

以下 delete-relational-database 示例删除指定托管数据库。

```
aws lightsail delete-relational-database \
    --relational-database-name Database-1
```

输出：

```
{
    "operations": [
        {
            "id": "3b0c41c1-053d-46f0-92a3-14f76141dc86",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1569875210.999,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "DeleteRelationalDatabase",
            "status": "Started",
            "statusChangedAt": 1569875210.999
        },
        {
            "id": "01ddeae8-a87a-4a4b-a1f3-092c71bf9180",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1569875211.029,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
```

```
            },
            "isTerminal": false,
            "operationDetails": "Database-1-FinalSnapshot-1569875210793",
            "operationType": "CreateRelationalDatabaseSnapshot",
            "status": "Started",
            "statusChangedAt": 1569875211.029
        },
        {
            "id": "74d73681-30e8-4532-974e-1f23cd3f9f73",
            "resourceName": "Database-1-FinalSnapshot-1569875210793",
            "resourceType": "RelationalDatabaseSnapshot",
            "createdAt": 1569875211.029,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "Database-1",
            "operationType": "CreateRelationalDatabaseSnapshot",
            "status": "Started",
            "statusChangedAt": 1569875211.029
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRelationalDatabase。

detach-static-ip

以下代码示例演示了如何使用 detach-static-ip。

Amazon CLI

将静态 IP 与实例分离

以下 detach-static-ip 示例将静态 IP StaticIp-1 与任何关联的实例分离。

```
aws lightsail detach-static-ip \
    --static-ip-name StaticIp-1
```

输出：

```
{
```

```
    "operations": [
        {
            "id": "2a43d8a3-9f2d-4fe7-bdd0-eEXAMPLE3cf3",
            "resourceName": "StaticIp-1",
            "resourceType": "StaticIp",
            "createdAt": 1571088261.999,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MEAN-1",
            "operationType": "DetachStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571088261.999
        },
        {
            "id": "41a7d40c-74e8-4d2e-a837-cEXAMPLEf747",
            "resourceName": "MEAN-1",
            "resourceType": "Instance",
            "createdAt": 1571088262.022,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "StaticIp-1",
            "operationType": "DetachStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571088262.022
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachStaticIp。

get-active-names

以下代码示例演示了如何使用 `get-active-names`。

Amazon CLI

获取活动资源名称

以下 get-active-names 示例返回已配置的 Amazon 区域中的活动资源名称。

```
aws lightsail get-active-names
```

输出：

```
{
    "activeNames": [
        "WordPress-1",
        "StaticIp-1",
        "MEAN-1",
        "Plesk_Hosting_Stack_on_Ubuntu-1"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetActiveNames。

get-auto-snapshots

以下代码示例演示了如何使用 get-auto-snapshots。

Amazon CLI

获取实例的可用自动快照

例如，以下 get-auto-snapshots 示例返回实例 WordPress-1 的可用自动快照。

```
aws lightsail get-auto-snapshots \
    --resource-name WordPress-1
```

输出：

```
{
    "resourceName": "WordPress-1",
    "resourceType": "Instance",
    "autoSnapshots": [
        {
            "date": "2019-10-14",
            "createdAt": 1571033872.0,
            "status": "Success",
            "fromAttachedDisks": []
        },
```

```
        {
            "date": "2019-10-13",
            "createdAt": 1570947473.0,
            "status": "Success",
            "fromAttachedDisks": []
        },
        {
            "date": "2019-10-12",
            "createdAt": 1570861072.0,
            "status": "Success",
            "fromAttachedDisks": []
        },
        {
            "date": "2019-10-11",
            "createdAt": 1570774672.0,
            "status": "Success",
            "fromAttachedDisks": []
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中保留实例或磁盘的自动快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAutoSnapshots。

get-blueprints

以下代码示例演示了如何使用 get-blueprints。

Amazon CLI

获取新实例的蓝图

以下 get-blueprints 示例显示可用于在 Amazon Lightsail 中创建新实例的所有可用蓝图的详细信息。

```
aws lightsail get-blueprints
```

输出：

```
{
    "blueprints": [
```

```
        {
            "blueprintId": "wordpress",
            "name": "WordPress",
            "group": "wordpress",
            "type": "app",
            "description": "Bitnami, the leaders in application packaging, and
 Automattic, the experts behind WordPress, have teamed up to offer this official
 WordPress image. This image is a pre-configured, ready-to-run image for running
 WordPress on Amazon Lightsail. WordPress is the world's most popular content
 management platform. Whether it's for an enterprise or small business website, or
 a personal or corporate blog, content authors can easily create content using its
 new Gutenberg editor, and developers can extend the base platform with additional
 features. Popular plugins like Jetpack, Akismet, All in One SEO Pack, WP Mail,
 Google Analytics for WordPress, and Amazon Polly are all pre-installed in this
 image. Let's Encrypt SSL certificates are supported through an auto-configuration
 script.",
            "isActive": true,
            "minPower": 0,
            "version": "6.5.3-0",
            "versionCode": "1",
            "productUrl": "https://aws.amazon.com/marketplace/pp/B00NN8Y43U",
            "licenseUrl": "https://aws.amazon.com/marketplace/pp/B00NN8Y43U#pdp-
usage",
            "platform": "LINUX_UNIX"
        },
        {
            "blueprintId": "lamp_8_bitnami",
            "name": "LAMP (PHP 8)",
            "group": "lamp_8",
            "type": "app",
            "description": "LAMP with PHP 8.X packaged by Bitnami enables you
 to quickly start building your websites and applications by providing a coding
 framework. As a developer, it provides standalone project directories to store your
 applications. This blueprint is configured for production environments. It includes
 SSL auto-configuration with Let's Encrypt certificates, and the latest releases of
 PHP, Apache, and MariaDB on Linux. This application also includes phpMyAdmin, PHP
 main modules and Composer.",
            "isActive": true,
            "minPower": 0,
            "version": "8.2.18-4",
            "versionCode": "1",
            "productUrl": "https://aws.amazon.com/marketplace/pp/
prodview-6g3gzfcih6dvu",
```

```
            "licenseUrl": "https://aws.amazon.com/marketplace/pp/
prodview-6g3gzfcih6dvu#pdp-usage",
            "platform": "LINUX_UNIX"
        },
        {
            "blueprintId": "nodejs",
            "name": "Node.js",
            "group": "node",
            "type": "app",
            "description": "Node.js packaged by Bitnami is a pre-configured, ready
 to run image for Node.js on Amazon EC2. It includes the latest version of Node.js,
 Apache, Python and Redis. The image supports multiple Node.js applications, each
 with its own virtual host and project directory. It is configured for production
 use and is secure by default, as all ports except HTTP, HTTPS and SSH ports are
 closed. Let's Encrypt SSL certificates are supported through an auto-configuration
 script. Developers benefit from instant access to a secure, update and consistent
 Node.js environment without having to manually install and configure multiple
 components and libraries.",
            "isActive": true,
            "minPower": 0,
            "version": "18.20.2-0",
            "versionCode": "1",
            "productUrl": "https://aws.amazon.com/marketplace/pp/B00NNZUAKO",
            "licenseUrl": "https://aws.amazon.com/marketplace/pp/B00NNZUAKO#pdp-
usage",
            "platform": "LINUX_UNIX"
        },
        ...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBlueprints。

get-bundles

以下代码示例演示了如何使用 get-bundles。

Amazon CLI

获取新实例的捆绑包

以下 get-bundles 示例显示可用于在 Amazon Lightsail 中创建新实例的所有可用捆绑包的详细信息。

```
aws lightsail get-bundles
```

输出：

```
{
    "bundles": [
        {
            "price": 5.0,
            "cpuCount": 2,
            "diskSizeInGb": 20,
            "bundleId": "nano_3_0",
            "instanceType": "nano",
            "isActive": true,
            "name": "Nano",
            "power": 298,
            "ramSizeInGb": 0.5,
            "transferPerMonthInGb": 1024,
            "supportedPlatforms": [
                "LINUX_UNIX"
            ]
        },
        {
            "price": 7.0,
            "cpuCount": 2,
            "diskSizeInGb": 40,
            "bundleId": "micro_3_0",
            "instanceType": "micro",
            "isActive": true,
            "name": "Micro",
            "power": 500,
            "ramSizeInGb": 1.0,
            "transferPerMonthInGb": 2048,
            "supportedPlatforms": [
                "LINUX_UNIX"
            ]
        },
        {
            "price": 12.0,
            "cpuCount": 2,
            "diskSizeInGb": 60,
```

```
            "bundleId": "small_3_0",
            "instanceType": "small",
            "isActive": true,
            "name": "Small",
            "power": 1000,
            "ramSizeInGb": 2.0,
            "transferPerMonthInGb": 3072,
            "supportedPlatforms": [
                "LINUX_UNIX"
            ]
        },
        ...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBundles。

get-cloud-formation-stack-records

以下代码示例演示了如何使用 get-cloud-formation-stack-records。

Amazon CLI

获取 CloudFormation 堆栈记录及其关联堆栈

以下 get-cloud-formation-stack-records 示例显示用于根据导出的 Amazon Lightsail 快照创建 Amazon EC2 资源的 CloudFormation 堆栈记录及其关联堆栈的详细信息。

```
aws lightsail get-cloud-formation-stack-records
```

输出：

```
{
    "cloudFormationStackRecords": [
        {
            "name": "CloudFormationStackRecord-588a4243-
e2d1-490d-8200-3a7513ecebdf",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:CloudFormationStackRecord/28d646ab-27bc-48d9-a422-1EXAMPLE6d37",
            "createdAt": 1565301666.586,
            "location": {
```

```
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "CloudFormationStackRecord",
            "state": "Succeeded",
            "sourceInfo": [
                {
                    "resourceType": "ExportSnapshotRecord",
                    "name": "ExportSnapshotRecord-
e02f23d7-0453-4aa9-9c95-91aa01a141dd",
                    "arn": "arn:aws:lightsail:us-
west-2:111122223333:ExportSnapshotRecord/f12b8792-f3ea-4d6f-b547-2EXAMPLE8796"
                }
            ],
            "destinationInfo": {
                "id": "arn:aws:cloudformation:us-west-2:111122223333:stack/
Lightsail-Stack-588a4243-e2d1-490d-8200-3EXAMPLEebdf/063203b0-
ba28-11e9-838b-0EXAMPLE8b00",
                "service": "Aws::CloudFormation::Stack"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCloudFormationStackRecords。

get-disk-snapshot

以下代码示例演示了如何使用 get-disk-snapshot。

Amazon CLI

获取有关磁盘快照的信息

以下 get-disk-snapshot 示例显示有关磁盘快照 Disk-1-1566839161 的详细信息。

```
aws lightsail get-disk-snapshot \
    --disk-snapshot-name Disk-1-1566839161
```

输出：

```
{
```

```
    "diskSnapshot": {
        "name": "Disk-1-1566839161",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:DiskSnapshot/
e2d0fa53-8ee0-41a0-8e56-0EXAMPLE1051",
        "supportCode": "6EXAMPLE3362/snap-0EXAMPLE06100d09",
        "createdAt": 1566839163.749,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "DiskSnapshot",
        "tags": [],
        "sizeInGb": 8,
        "state": "completed",
        "progress": "100%",
        "fromDiskName": "Disk-1",
        "fromDiskArn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
        "isFromAutoSnapshot": false
    }
}
```

有关更多信息，请参阅《指南》中相应标题下的内容。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDiskSnapshot。

get-disk-snapshots

以下代码示例演示了如何使用 get-disk-snapshots。

Amazon CLI

获取有关所有磁盘快照的信息

以下 get-disk-snapshots 示例显示有关已配置的 Amazon 区域中所有磁盘快照的详细信息。

```
aws lightsail get-disk-snapshots
```

输出：

```
{
    "diskSnapshots": [
```

```
        {
            "name": "Disk-2-1571090588",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:DiskSnapshot/32e889a9-38d4-4687-9f21-eEXAMPLE7839",
            "supportCode": "6EXAMPLE3362/snap-0EXAMPLE1ca192a4",
            "createdAt": 1571090591.226,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "DiskSnapshot",
            "tags": [],
            "sizeInGb": 8,
            "state": "completed",
            "progress": "100%",
            "fromDiskName": "Disk-2",
            "fromDiskArn": "arn:aws:lightsail:us-
west-2:111122223333:Disk/6a343ff8-6341-422d-86e2-bEXAMPLE16c2",
            "isFromAutoSnapshot": false
        },
        {
            "name": "Disk-1-1566839161",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:DiskSnapshot/
e2d0fa53-8ee0-41a0-8e56-0EXAMPLE1051",
            "supportCode": "6EXAMPLE3362/snap-0EXAMPLEe06100d09",
            "createdAt": 1566839163.749,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "DiskSnapshot",
            "tags": [],
            "sizeInGb": 8,
            "state": "completed",
            "progress": "100%",
            "fromDiskName": "Disk-1",
            "fromDiskArn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
            "isFromAutoSnapshot": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDiskSnapshots。

get-disk

以下代码示例演示了如何使用 get-disk。

Amazon CLI

获取有关块存储磁盘的信息

以下 get-disk 示例显示有关磁盘 Disk-1 的详细信息。

```
aws lightsail get-disk \
    --disk-name Disk-1
```

输出：

```
{
    "disk": {
        "name": "Disk-1",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
        "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
        "createdAt": 1566585439.587,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
        "resourceType": "Disk",
        "tags": [],
        "sizeInGb": 8,
        "isSystemDisk": false,
        "iops": 100,
        "path": "/dev/xvdf",
        "state": "in-use",
        "attachedTo": "WordPress_Multisite-1",
        "isAttached": true,
        "attachmentState": "attached"
    }
}
```

有关更多信息，请参阅《指南》中相应标题下的内容。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDisk。

get-disks

以下代码示例演示了如何使用 get-disks。

Amazon CLI

获取有关所有块存储磁盘的信息

以下 get-disks 示例显示有关已配置的 Amazon 区域中所有磁盘的详细信息。

```
aws lightsail get-disks
```

输出：

```
{
    "disks": [
        {
            "name": "Disk-2",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:Disk/6a343ff8-6341-422d-86e2-bEXAMPLE16c2",
            "supportCode": "6EXAMPLE3362/vol-0EXAMPLE929602087",
            "createdAt": 1571090461.634,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "resourceType": "Disk",
            "tags": [],
            "sizeInGb": 8,
            "isSystemDisk": false,
            "iops": 100,
            "state": "available",
            "isAttached": false,
            "attachmentState": "detached"
        },
        {
            "name": "Disk-1",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
            "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
            "createdAt": 1566585439.587,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
```

```
            },
            "resourceType": "Disk",
            "tags": [],
            "sizeInGb": 8,
            "isSystemDisk": false,
            "iops": 100,
            "path": "/dev/xvdf",
            "state": "in-use",
            "attachedTo": "WordPress_Multisite-1",
            "isAttached": true,
            "attachmentState": "attached"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDisks。

get-domain

以下代码示例演示了如何使用 get-domain。

Amazon CLI

获取有关域的信息

以下 get-domain 示例显示有关域 example.com 的详细信息。

注意：Lightsail 的域相关 API 操作仅在 us-east-1 Amazon 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含“ --region us-east-1”参数，否则命令将失败。

```
aws lightsail get-domain \
    --domain-name example.com \
    --region us-east-1
```

输出：

```
{
    "domain": {
        "name": "example.com",
        "arn":
 "arn:aws:lightsail:global:111122223333:Domain/28cda903-3f15-44b2-9baf-3EXAMPLEb304",
        "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEONGSC1",
        "createdAt": 1570728588.6,
```

```
        "location": {
            "availabilityZone": "all",
            "regionName": "global"
        },
        "resourceType": "Domain",
        "tags": [],
        "domainEntries": [
            {
                "id": "-1682899164",
                "name": "example.com",
                "target": "192.0.2.0",
                "isAlias": false,
                "type": "A"
            },
            {
                "id": "1703104243",
                "name": "example.com",
                "target": "ns-137.awsdns-17.com",
                "isAlias": false,
                "type": "NS"
            },
            {
                "id": "-1038331153",
                "name": "example.com",
                "target": "ns-1710.awsdns-21.co.uk",
                "isAlias": false,
                "type": "NS"
            },
            {
                "id": "-2107289565",
                "name": "example.com",
                "target": "ns-692.awsdns-22.net",
                "isAlias": false,
                "type": "NS"
            },
            {
                "id": "1582095705",
                "name": "example.com",
                "target": "ns-1436.awsdns-51.org",
                "isAlias": false,
                "type": "NS"
            },
            {
                "id": "-1769796132",
```

```
                "name": "example.com",
                "target": "ns-1710.awsdns-21.co.uk. awsdns-hostmaster.amazon.com. 1
 7200 900 1209600 86400",
                "isAlias": false,
                "type": "SOA"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomain。

get-domains

以下代码示例演示了如何使用 get-domains。

Amazon CLI

获取有关所有域的信息

以下 get-domains 示例显示有关已配置的 Amazon 区域中所有域的详细信息。

注意：Lightsail 的域相关 API 操作仅在 us-east-1 Amazon 区域可用。如果您的 CLI 配置文件配置为使用其他区域，则必须包含 --region us-east-1 参数，否则命令将失败。

```
aws lightsail get-domains \
    --region us-east-1
```

输出：

```
{
    "domains": [
        {
            "name": "example.com",
            "arn":
 "arn:aws:lightsail:global:111122223333:Domain/28cda903-3f15-44b2-9baf-3EXAMPLEb304",
            "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEONGSC1",
            "createdAt": 1570728588.6,
            "location": {
                "availabilityZone": "all",
                "regionName": "global"
            },
            "resourceType": "Domain",
```

```
            "tags": [],
            "domainEntries": [
                {
                    "id": "-1682899164",
                    "name": "example.com",
                    "target": "192.0.2.0",
                    "isAlias": false,
                    "type": "A"
                },
                {
                    "id": "1703104243",
                    "name": "example.com",
                    "target": "ns-137.awsdns-17.com",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "-1038331153",
                    "name": "example.com",
                    "target": "ns-4567.awsdns-21.co.uk",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "-2107289565",
                    "name": "example.com",
                    "target": "ns-333.awsdns-22.net",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "1582095705",
                    "name": "example.com",
                    "target": "ns-1111.awsdns-51.org",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "-1769796132",
                    "name": "example.com",
                    "target": "ns-1234.awsdns-21.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
                    "isAlias": false,
                    "type": "SOA"
```

```
                },
                {
                    "id": "1029454894",
                    "name": "_dead6a124ede046a0319eb44a4eb3cbc.example.com",
                    "target": "_be133b0a0899fb7b6bf79d9741d1a383.hkvuiqjoua.acm-
validations.aws",
                    "isAlias": false,
                    "type": "CNAME"
                }
            ]
        },
        {
            "name": "example.net",
            "arn": "arn:aws:lightsail:global:111122223333:Domain/9c9f0d70-
c92e-4753-86c2-6EXAMPLE029d",
            "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLE5TPKMV",
            "createdAt": 1556661071.384,
            "location": {
                "availabilityZone": "all",
                "regionName": "global"
            },
            "resourceType": "Domain",
            "tags": [],
            "domainEntries": [
                {
                    "id": "-766320943",
                    "name": "example.net",
                    "target": "192.0.2.2",
                    "isAlias": false,
                    "type": "A"
                },
                {
                    "id": "-453913825",
                    "name": "example.net",
                    "target": "ns-123.awsdns-10.net",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "1553601564",
                    "name": "example.net",
                    "target": "ns-4444.awsdns-47.co.uk",
                    "isAlias": false,
                    "type": "NS"
```

```
                },
                {
                    "id": "1653797661",
                    "name": "example.net",
                    "target": "ns-7890.awsdns-61.org",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "706414698",
                    "name": "example.net",
                    "target": "ns-123.awsdns-44.com",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "337271745",
                    "name": "example.net",
                    "target": "ns-4444.awsdns-47.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
                    "isAlias": false,
                    "type": "SOA"
                },
                {
                    "id": "-1785431096",
                    "name": "www.example.net",
                    "target": "192.0.2.2",
                    "isAlias": false,
                    "type": "A"
                }
            ]
        },
        {
            "name": "example.org",
            "arn": "arn:aws:lightsail:global:111122223333:Domain/
f0f13ba3-3df0-4fdc-8ebb-1EXAMPLEf26e",
            "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEAFO38",
            "createdAt": 1556661199.106,
            "location": {
                "availabilityZone": "all",
                "regionName": "global"
            },
            "resourceType": "Domain",
            "tags": [],
```

```
            "domainEntries": [
                {
                    "id": "2065301345",
                    "name": "example.org",
                    "target": "192.0.2.4",
                    "isAlias": false,
                    "type": "A"
                },
                {
                    "id": "-447198516",
                    "name": "example.org",
                    "target": "ns-123.awsdns-45.com",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "136463022",
                    "name": "example.org",
                    "target": "ns-9999.awsdns-15.co.uk",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "1395941679",
                    "name": "example.org",
                    "target": "ns-555.awsdns-01.net",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "872052569",
                    "name": "example.org",
                    "target": "ns-6543.awsdns-38.org",
                    "isAlias": false,
                    "type": "NS"
                },
                {
                    "id": "1001949377",
                    "name": "example.org",
                    "target": "ns-1234.awsdns-15.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
                    "isAlias": false,
                    "type": "SOA"
                },
```

```
                {
                    "id": "1046191192",
                    "name": "www.example.org",
                    "target": "192.0.2.4",
                    "isAlias": false,
                    "type": "A"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomains。

get-export-snapshot-record

以下代码示例演示了如何使用 get-export-snapshot-record。

Amazon CLI

获取导出到 Amazon EC2 的快照记录

以下 get-export-snapshot-record 示例显示有关导出到 Amazon EC2 的 Amazon Lightsail 实例或磁盘快照的详细信息。

```
aws lightsail get-export-snapshot-records
```

输出：

```
{
    "exportSnapshotRecords": [
        {
            "name": "ExportSnapshotRecord-d2da10ce-0b3c-4ae1-ab3a-2EXAMPLEa586",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:ExportSnapshotRecord/076c7060-b0cc-4162-98f0-2EXAMPLEe28e",
            "createdAt": 1543534665.678,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "ExportSnapshotRecord",
            "state": "Succeeded",
```

```
            "sourceInfo": {
                "resourceType": "InstanceSnapshot",
                "createdAt": 1540339310.706,
                "name": "WordPress-512MB-Oregon-1-1540339219",
                "arn": "arn:aws:lightsail:us-
west-2:111122223333:InstanceSnapshot/5446f534-ed60-4c17-b4a5-bEXAMPLEf8b7",
                "fromResourceName": "WordPress-512MB-Oregon-1",
                "fromResourceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/4b8f1f24-e4d1-4cf3-88ff-cEXAMPLEa397",
                "instanceSnapshotInfo": {
                    "fromBundleId": "nano_2_0",
                    "fromBlueprintId": "wordpress_4_9_8",
                    "fromDiskInfo": [
                        {
                            "path": "/dev/sda1",
                            "sizeInGb": 20,
                            "isSystemDisk": true
                        }
                    ]
                }
            },
            "destinationInfo": {
                "id": "ami-0EXAMPLEc0d65058e",
                "service": "Aws::EC2::Image"
            }
        },
        {
            "name": "ExportSnapshotRecord-1c94e884-40ff-4fe1-9302-0EXAMPLE14c2",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:ExportSnapshotRecord/
fb392ce8-6567-4013-9bfd-3EXAMPLE5b4c",
            "createdAt": 1543432110.2,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "ExportSnapshotRecord",
            "state": "Succeeded",
            "sourceInfo": {
                "resourceType": "InstanceSnapshot",
                "createdAt": 1540833603.545,
                "name": "LAMP_PHP_5-512MB-Oregon-1-1540833565",
                "arn": "arn:aws:lightsail:us-
west-2:111122223333:InstanceSnapshot/82334399-b5f2-49ec-8382-0EXAMPLEe45f",
                "fromResourceName": "LAMP_PHP_5-512MB-Oregon-1",
```

```
                "fromResourceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/863b9f35-ab1e-4418-bdd2-1EXAMPLEbab2",
                "instanceSnapshotInfo": {
                    "fromBundleId": "nano_2_0",
                    "fromBlueprintId": "lamp_5_6_37_2",
                    "fromDiskInfo": [
                        {
                            "path": "/dev/sda1",
                            "sizeInGb": 20,
                            "isSystemDisk": true
                        }
                    ]
                }
            },
            "destinationInfo": {
                "id": "ami-0EXAMPLE7c5ec84e2",
                "service": "Aws::EC2::Image"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetExportSnapshotRecord。

get-instance-access-details

以下代码示例演示了如何使用 get-instance-access-details。

Amazon CLI

获取实例的主机密钥信息

以下 get-instance-access-details 示例显示实例 WordPress_Multisite-1 的主机密钥信息。

```
aws lightsail get-instance-access-details \
    --instance-name WordPress_Multisite-1
```

输出：

```
{
    "accessDetails": {
```

```
        "certKey": "ssh-rsa-cert-v01@openssh.com
 AEXAMPLEaC1yc2EtY2VydC12MDFAb3BlbnNzaC5jb20AAAAgNf076Dt3ppmPd0fPxZVMmS491aEAYYH9cHqAJ3fNML8
vEXAMPLE2eBWJyQvn7ol/
i0+s966h5sx8qUD79lPB7q5UESd5VZGFtytrykfQJnjiwqe7EV5agzvjblLj26Fb37EKda9HVfCOu8pWbvky7Tyn9w29
+xMfQM9xVz0rXZmqx8uJidJpRgLCMTviofwQJU/
K1EXAMPLEAAAAAAAABAAAALS00MzMzMDU4MzA4ODg1MTY2NjM4Onp6UWlndHk4UElRSG9STitOTG5QSEE9PQAAAAsAAA
+LiB+ozNbUA0cdNL9Y67x7qPv/R7XhTc21+2A+8+GuVpK/Kz9dqDMKNAEXAMPLE+YYN
+tiXm7Y8OgziK+7iDB7xUuQ4vghmn4+qgz9mKwYgWvVe2+0XLuV7cnWPB7iUlHQg
+E3LUKrV4ZFw9pj7X2dFdNKfMxwWgI1ISWKimEXAMPLEeHjrf1Rqc/
QH6TpWCvPfcx8uvwVqdwTfkE/SfA5BCzbGGI1UmIUadh8nHcb5FamQ1hK7kECy47K/x9FMn/
KwmM7pCwJbSLDMO7n9bnbvck6m8ZoB2N2YLMG5dW7BerEXAMPLEobqfdtyYJHHel1EyyEJs1fWNU3D5JIGlgzcPAV
+ZlbQyUCZXf0oslSa+HE85fO/
FRq9SVSBSHrmbeb0frlPhgMzgSmqLeyhlbr6wwWIDbREXAMPLEJZ49H7RdQxdKyYrZPWvRgcr0qI2EL0tAajnpQQ8UZq
Aqter0xN5PhFL0J49OWTacwCGRAjLhibAx7K1t/1ZXWo6c+ijq8clll327EXAMPLE/
e89GC89KcmKCxfGQniDAUgF8UqofIbq3ZOUgiAAYCVXclI4L68NhVXyoWuQXPBRQSEXAMPLEWm74tDL9tFN3c7tSe/
Oz0cTR+4sAAAIPAAAAB3NzaC1yc2EAAAIAQnG/
L0DqiSnLrWhEox4aHqMgd0m0oLLAYx6OQH9F0TM9EXAMPLE961rzSCMon7ZgsWNnL0OwZQgDG
+rtJ4N0B7HOVwns4ynUFbzNQ3qFGGeE3lKwX1L41vV1iSy7sDk8aI0LmrKJi1LE1Qc1l8uboRlwoXOYEXAMPLEaUCeX
+10+WEXAMPLEg6Y4U4ZvE2B3xyRdpvysb5TGFNtk5qPslacnVkoLOGsZZXMpLGJnG4OBpQLLtpj9sNMxAgZPCAUjhkqk
+nxO9O4NUZ2pTWbVSUaV1gm6pug9xbwNO1Im21t34JeLlKTqxcJ6zzS8W0c0KKpAm5c4hWkseMbyutS2jav/4hiS
+BhrYgptzfwe5qRXEXAMPLEHZQr3YfGzYoBJ/
lLK3NHhxOihhsfAYwMei0BFZT1F/7CT3IH4iitEkIgodi06/
Mw6UDqMPozyQCK1lEA6LFhYCOZG9drWcoRa74lM4kY9TP028Za8gDMh1WpkXLq9Gixon5OHP8aM/
sEXAMPLEr2+fnkw+1BtoO5L6+VKoPlXaGqZ/fBYEXAMPLEAMQHjnLM1JYNvtEEPhp+TNzXHzuixWf/
Ht04m0AVpXrzIDXaS1O2tXY=",
        "ipAddress": "192.0.2.0",
        "privateKey": "-----BEGIN RSA PRIVATE KEY-----
\nEXAMPLEBAAKCAQEA+AD3qeU2toBy5O5v7wnRLVo/tngVickL5+6Jf4tPrPeuoebM
\nfKlA+/ZTwe6uVBEneVWRhbcra8pH0CZ44sKnuxFeWoM7425S49uhW9+xCnWvR1Xw
\njrvKVm75Mu08p/cNvfWugrBuaPB65DspgxNnOfZWMVxpIpSqOSPWmSwQHV597d6C
\nrEXAMPLEo8hJmqz2KFQ09X7fB2lBruGgr9aXiNPmWmovYKqwFmrnFvR7odFmDecq
\n5EXAMPLE9dyU1ZsrWhGby77eYrVaFl0GNGQ8qy1HGUiscquZ9NDIL49n4mXbfsTH
\n0EXAMPLE12ZqsfLiYnSaUYCwjE74qH8ECVPytQIDAQABAoIBAHeZV9Z58JHAjifz
\nCEXAMPLEEqC3doOVDgXSlkKI92qNo4z2VcUEho878paCuVVXVHcCGgSnGeyIh2tN
\nMEXAMPLESohR427BhH3YLA+3Z5SIvnejbTgYPfLC37B8khTaYqkqMvdZiFVZK5qn
\nIEXAMPLEM93oF9eSZCjcLKB/jGHsfb0eCDMP8BshHE2beuqzVMoK1DxOnvoP3+Fp
\nAEXAMPLESq6pDpCo9YVUX8g1u3Ro9cPl2LXHDy+oVEY5KhbZQJ7VU1I72WOvppWW
\nOEXAMPLEkgYlq7p6qYtYcSgTEjz14gDiMfQ7SyHB3alkIoNONQ9ZPaWHyJvymeud
\noQTNuz0CgYEA/LFWNTEZrzdzdR1kJmyNRmAermU0B6utyNENChAlHGSHkB+1lVSh
\nbEXAMPLEQo9ooUeW5UxO3YwacZLoDT1mwxw1Ptc1+PNycZoLe1fE9UdARrdmGTob
\n8l7CPLSXp3xuR8VqSp2fnIc7hfiQs/NrPX9gm/EOrB0we0RKyDSzWScCgYEA+z/r
\niob+nJZq0YbnOSuP6oMULP4vnWniWj8MIhUJU53LwSAM8DeJdONKDdkuiOd52aAL
\nVgn7nLo88rVWKhJwVc4tu/rNgZLcR3bP4+kL6zand0KQnMLyOzNA2Ys26aa5udH1\nqWl0WTt9WEm/
h10ndC1knOMectrvsG17b38y5sMCgYEA54NiRGGz8oCPW6GN/FZA
```

```
\nKEXAMPLE5tw34GEH3Uxlc9n3CejDaQmczOATwX4nIwRZDEqWyYZcS0btg1jhGiBD\nYEXAMPLEkc8Z71L/
agZEAaVCEog9FqfSqwB
+XTfoKh8qur74X1yCu9p6gof1q6k9\neEXAMPLEchJcNNOg4ETIfMkCgYBdVORRhE4mqvWpOdzA7v66FdEz2YSkjAXKk
\naEXAMPLE8Z/8yBSmuBv1Qv03XA12my462uB92uzzGAuW
+1yBc2Kn1sXqYTy0y1z0\ngEXAMPLEBogjw4MqHKL1bPKMHyQU8/
q24PaYgzHPzy13wlH6pTYf1XqlHdE2D6Vv\nyEXAMPLEgQC3i/
kVVhky/2XRwRVlC7JO2Bg3QGTx38hpmDa5IuofKANjA+Wa3/zy\nbEXAMPLE6ytQgD9GN/YtBq+uhO
+2ZkvXPL+CWRi0ZRXpPwYDBBFU9Cw0AuWWGlL8\nwEXAMPLExMlcysRgcWB9RNgf3AuOpFd2i6XT/
riNsvvkpmJ+VooU8g==\n-----END RSA PRIVATE KEY-----\n",
        "protocol": "ssh",
        "instanceName": "WordPress_Multisite-1",
        "username": "bitnami",
        "hostKeys": [
            {
                "algorithm": "ssh-rsa",
                "publicKey":
 "AEXAMPLEaC1yc2EAAAADAQABAAABAQCoeR9ieZTjQ3pXCHczuAYZFjlF7t
+uBkXuqeGMRex78pCvmS+DiEXAMPLEuJ1Q8dcKhrQL4HpXbD9dosVCTaJnJwb4MQqsuSVFdHFzy3guP
+BKclWqtxJEXAMPLEsBGqZZlrIv6a9bTA0TCplZ8AD+hSRTaSXXqg6FT
+Qf16IktH0XlMs7xIEXAMPLEmNtjCpzZiGXDHzytoMvUgwa8uHPp44Og36EUu4VqQxoUHPJKoXvcQizyk3K8ym0hP0Tp
Ot6y9HwvykEXAMPLEAfbKjbR42+u6+OSlkr4d339q2U1sTDytJhhs8HUel1wTfGRfp",
                "witnessedAt": 1570744377.699,
                "fingerprintSHA1": "SHA1:GEXAMPLEMoYgUg0ucadqU9Bt3Lk",
                "fingerprintSHA256": "SHA256:IEXAMPLEcB5vgxnAUoJawbdZ
+MwELhIp6FUxuwq/LIU"
            },
            {
                "algorithm": "ssh-ed25519",
                "publicKey":
 "AEXAMPLEaC1lZDI1NTE5AAAAIC1gwGPDfGaONxEXAMPLEJX3UNap781QxHQmn8nzlrUv",
                "witnessedAt": 1570744377.697,
                "fingerprintSHA1": "SHA1:VEXAMPLE5ReqSmTgv03sSUw9toU",
                "fingerprintSHA256": "SHA256:0EXAMPLEdE6tI95k3TJpG
+qhJbAoknB0yz9nAEaDt3A"
            },
            {
                "algorithm": "ecdsa-sha2-nistp256",
                "publicKey":
 "AEXAMPLEZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABEXAMPLE9B4mZy8YSsZW7cixCDq5yHSAAxjJkDo5
+EnKlDCsYtUkxxEXAMPLE6VOWL2z63RTKa2AUPgd8irjxWI=",
                "witnessedAt": 1570744377.707,
                "fingerprintSHA1": "SHA1:UEXAMPLEOYCfXsCf2G6tDg+7YG0",
                "fingerprintSHA256": "SHA256:wEXAMPLEQ9a/
iEXAMPLEhRufm6U9vFU4cpkMPHnBsNA"
```

```
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceAccessDetails。

get-instance-metric-data

以下代码示例演示了如何使用 get-instance-metric-data。

Amazon CLI

获取实例的指标数据

以下 get-instance-metric-data 示例返回实例 MEAN-1 在 1571342400 和 1571428800 期间 CPUUtilization 每 7200 秒（2 小时）的平均百分率。

我们建议您使用 unix 时间转换器来识别开始和结束时间。

```
aws lightsail get-instance-metric-data \
    --instance-name MEAN-1 \
    --metric-name CPUUtilization \
    --period 7200 \
    --start-time 1571342400 \
    --end-time 1571428800 \
    --unit Percent \
    --statistics Average
```

输出：

```
{
    "metricName": "CPUUtilization",
    "metricData": [
        {
            "average": 0.26113718770120725,
            "timestamp": 1571342400.0,
            "unit": "Percent"
        },
        {
            "average": 0.26861268928111953,
            "timestamp": 1571392800.0,
            "unit": "Percent"
```

```
        },
        {
            "average": 0.28187475104748777,
            "timestamp": 1571378400.0,
            "unit": "Percent"
        },
        {
            "average": 0.2651936960458352,
            "timestamp": 1571421600.0,
            "unit": "Percent"
        },
        {
            "average": 0.2561856213712188,
            "timestamp": 1571371200.0,
            "unit": "Percent"
        },
        {
            "average": 0.3021383254607764,
            "timestamp": 1571356800.0,
            "unit": "Percent"
        },
        {
            "average": 0.2618381649223539,
            "timestamp": 1571407200.0,
            "unit": "Percent"
        },
        {
            "average": 0.26331929394825787,
            "timestamp": 1571400000.0,
            "unit": "Percent"
        },
        {
            "average": 0.2576348407007818,
            "timestamp": 1571385600.0,
            "unit": "Percent"
        },
        {
            "average": 0.2513008454658378,
            "timestamp": 1571364000.0,
            "unit": "Percent"
        },
        {
            "average": 0.26329974562758346,
            "timestamp": 1571414400.0,
```

```
            "unit": "Percent"
        },
        {
            "average": 0.2667092536656445,
            "timestamp": 1571349600.0,
            "unit": "Percent"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceMetricData。

get-instance-port-states

以下代码示例演示了如何使用 get-instance-port-states。

Amazon CLI

获取实例的主机密钥信息

以下 get-instance-port-states 示例返回为实例 MEAN-1 配置的防火墙端口。

```
aws lightsail get-instance-port-states \
    --instance-name MEAN-1
```

输出：

```
{
    "portStates": [
        {
            "fromPort": 80,
            "toPort": 80,
            "protocol": "tcp",
            "state": "open"
        },
        {
            "fromPort": 22,
            "toPort": 22,
            "protocol": "tcp",
            "state": "open"
        },
        {
            "fromPort": 443,
```

```
            "toPort": 443,
            "protocol": "tcp",
            "state": "open"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstancePortStates。

get-instance-snapshot

以下代码示例演示了如何使用 get-instance-snapshot。

Amazon CLI

获取有关指定实例快照的信息

以下 get-instance-snapshot 示例显示有关指定实例快照的详细信息。

```
aws lightsail get-instance-snapshot \
    --instance-snapshot-name MEAN-1-1571419854
```

输出：

```
{
    "instanceSnapshot": {
        "name": "MEAN-1-1571419854",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/
ac54700c-48a8-40fd-b065-2EXAMPLEac8f",
        "supportCode": "6EXAMPLE3362/ami-0EXAMPLE67a73020d",
        "createdAt": 1571419891.927,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "InstanceSnapshot",
        "tags": [],
        "state": "available",
        "fromAttachedDisks": [],
        "fromInstanceName": "MEAN-1",
        "fromInstanceArn": "arn:aws:lightsail:us-west-2:111122223333:Instance/
bd470fc5-a68b-44c5-8dbc-8EXAMPLEbada",
        "fromBlueprintId": "mean",
```

```
        "fromBundleId": "medium_3_0",
        "isFromAutoSnapshot": false,
        "sizeInGb": 80
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceSnapshot。

get-instance-snapshots

以下代码示例演示了如何使用 get-instance-snapshots。

Amazon CLI

获取有关您的所有实例快照的信息

以下 get-instance-snapshots 示例显示有关已配置的 Amazon 区域中所有实例快照的详细信息。

```
aws lightsail get-instance-snapshots
```

输出：

```
{
    "instanceSnapshots": [
        {
            "name": "MEAN-1-1571421498",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/
a20e6ebe-b0ee-4ae4-a750-3EXAMPLEcb0c",
            "supportCode": "6EXAMPLE3362/ami-0EXAMPLEe33cabfa1",
            "createdAt": 1571421527.755,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "InstanceSnapshot",
            "tags": [
                {
                    "key": "no_delete"
                }
            ],
            "state": "available",
```

```
            "fromAttachedDisks": [],
            "fromInstanceName": "MEAN-1",
            "fromInstanceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/1761aa0a-6038-4f25-8b94-2EXAMPLE19fd",
            "fromBlueprintId": "wordpress",
            "fromBundleId": "micro_3_0",
            "isFromAutoSnapshot": false,
            "sizeInGb": 40
        },
        {
            "name": "MEAN-1-1571419854",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/
ac54700c-48a8-40fd-b065-2EXAMPLEac8f",
            "supportCode": "6EXAMPLE3362/ami-0EXAMPLE67a73020d",
            "createdAt": 1571419891.927,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "InstanceSnapshot",
            "tags": [],
            "state": "available",
            "fromAttachedDisks": [],
            "fromInstanceName": "MEAN-1",
            "fromInstanceArn": "arn:aws:lightsail:us-west-2:111122223333:Instance/
bd470fc5-a68b-44c5-8dbc-8EXAMPLEbada",
            "fromBlueprintId": "mean",
            "fromBundleId": "medium_3_0",
            "isFromAutoSnapshot": false,
            "sizeInGb": 80
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceSnapshots。

get-instance-state

以下代码示例演示了如何使用 get-instance-state。

Amazon CLI

获取有关实例状态的信息

以下 get-instance-state 示例返回指定实例的状态。

```
aws lightsail get-instance-state \
    --instance-name MEAN-1
```

输出：

```
{
    "state": {
        "code": 16,
        "name": "running"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstanceState。

get-instance

以下代码示例演示了如何使用 get-instance。

Amazon CLI

获取有关实例的信息

以下 get-instance 示例显示有关实例 MEAN-1 的详细信息。

```
aws lightsail get-instance \
    --instance-name MEAN-1
```

输出：

```
{
    "instance": {
        "name": "MEAN-1",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:Instance/bd470fc5-
a68b-44c5-8dbc-EXAMPLE4bada",
        "supportCode": "6EXAMPLE3362/i-05EXAMPLE407c97d3",
        "createdAt": 1570635023.124,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
```

```
        "resourceType": "Instance",
        "tags": [],
        "blueprintId": "mean",
        "blueprintName": "MEAN",
        "bundleId": "medium_3_0",
        "isStaticIp": false,
        "privateIpAddress": "192.0.2.0",
        "publicIpAddress": "192.0.2.0",
        "hardware": {
            "cpuCount": 2,
            "disks": [
                {
                    "createdAt": 1570635023.124,
                    "sizeInGb": 80,
                    "isSystemDisk": true,
                    "iops": 240,
                    "path": "/dev/xvda",
                    "attachedTo": "MEAN-1",
                    "attachmentState": "attached"
                }
            ],
            "ramSizeInGb": 4.0
        },
        "networking": {
            "monthlyTransfer": {
                "gbPerMonthAllocated": 4096
            },
            "ports": [
                {
                    "fromPort": 80,
                    "toPort": 80,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 22,
                    "toPort": 22,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
```

```
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 443,
                    "toPort": 443,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                }
            ]
        },
        "state": {
            "code": 16,
            "name": "running"
        },
        "username": "bitnami",
        "sshKeyName": "MyKey"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstance。

get-instances

以下代码示例演示了如何使用 get-instances。

Amazon CLI

获取有关所有实例的信息

以下 get-instances 示例显示有关已配置的 Amazon 区域中所有实例的详细信息。

```
aws lightsail get-instances
```

输出：

```
{
    "instances": [
        {
            "name": "Windows_Server_2022-1",
```

```
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/0f44fbb9-8f55-4e47-a25e-EXAMPLE04763",
            "supportCode": "62EXAMPLE362/i-0bEXAMPLE71a686b9",
            "createdAt": 1571332358.665,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "resourceType": "Instance",
            "tags": [],
            "blueprintId": "windows_server_2022",
            "blueprintName": "Windows Server 2022",
            "bundleId": "large_win_3_0",
            "isStaticIp": false,
            "privateIpAddress": "192.0.2.0",
            "publicIpAddress": "192.0.2.0",
            "hardware": {
                "cpuCount": 1,
                "disks": [
                    {
                        "createdAt": 1571332358.665,
                        "sizeInGb": 160,
                        "isSystemDisk": true,
                        "iops": 180,
                        "path": "/dev/sda1",
                        "attachedTo": "Windows_Server_2022-1",
                        "attachmentState": "attached"
                    },
                    {
                        "name": "my-disk-for-windows-server",
                        "arn": "arn:aws:lightsail:us-
west-2:111122223333:Disk/4123a81c-484c-49ea-afea-5EXAMPLEda87",
                        "supportCode": "6EXAMPLE3362/vol-0EXAMPLEb2b99ca3d",
                        "createdAt": 1571355063.494,
                        "location": {
                            "availabilityZone": "us-west-2a",
                            "regionName": "us-west-2"
                        },
                        "resourceType": "Disk",
                        "tags": [],
                        "sizeInGb": 128,
                        "isSystemDisk": false,
                        "iops": 384,
                        "path": "/dev/xvdf",
```

```
                    "state": "in-use",
                    "attachedTo": "Windows_Server_2022-1",
                    "isAttached": true,
                    "attachmentState": "attached"
                }
            ],
            "ramSizeInGb": 8.0
        },
        "networking": {
            "monthlyTransfer": {
                "gbPerMonthAllocated": 3072
            },
            "ports": [
                {
                    "fromPort": 80,
                    "toPort": 80,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 22,
                    "toPort": 22,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 3389,
                    "toPort": 3389,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                }
            ]
        },
        "state": {
            "code": 16,
```

```
                "name": "running"
            },
            "username": "Administrator",
            "sshKeyName": "LightsailDefaultKeyPair"
        },
        {
            "name": "MEAN-1",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:Instance/bd470fc5-
a68b-44c5-8dbc-8EXAMPLEbada",
            "supportCode": "6EXAMPLE3362/i-0EXAMPLEa407c97d3",
            "createdAt": 1570635023.124,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "resourceType": "Instance",
            "tags": [],
            "blueprintId": "mean",
            "blueprintName": "MEAN",
            "bundleId": "medium_3_0",
            "isStaticIp": false,
            "privateIpAddress": "192.0.2.0",
            "publicIpAddress": "192.0.2.0",
            "hardware": {
                "cpuCount": 2,
                "disks": [
                    {
                        "name": "Disk-1",
                        "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
                        "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
                        "createdAt": 1566585439.587,
                        "location": {
                            "availabilityZone": "us-west-2a",
                            "regionName": "us-west-2"
                        },
                        "resourceType": "Disk",
                        "tags": [
                            {
                                "key": "test"
                            }
                        ],
                        "sizeInGb": 8,
                        "isSystemDisk": false,
```

```
                    "iops": 240,
                    "path": "/dev/xvdf",
                    "state": "in-use",
                    "attachedTo": "MEAN-1",
                    "isAttached": true,
                    "attachmentState": "attached"
                },
                {
                    "createdAt": 1570635023.124,
                    "sizeInGb": 80,
                    "isSystemDisk": true,
                    "iops": 240,
                    "path": "/dev/sda1",
                    "attachedTo": "MEAN-1",
                    "attachmentState": "attached"
                }
            ],
            "ramSizeInGb": 4.0
        },
        "networking": {
            "monthlyTransfer": {
                "gbPerMonthAllocated": 4096
            },
            "ports": [
                {
                    "fromPort": 80,
                    "toPort": 80,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 22,
                    "toPort": 22,
                    "protocol": "tcp",
                    "accessFrom": "Anywhere (0.0.0.0/0)",
                    "accessType": "public",
                    "commonName": "",
                    "accessDirection": "inbound"
                },
                {
                    "fromPort": 443,
```

```
                        "toPort": 443,
                        "protocol": "tcp",
                        "accessFrom": "Anywhere (0.0.0.0/0)",
                        "accessType": "public",
                        "commonName": "",
                        "accessDirection": "inbound"
                    }
                ]
            },
            "state": {
                "code": 16,
                "name": "running"
            },
            "username": "bitnami",
            "sshKeyName": "MyTestKey"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInstances。

get-key-pair

以下代码示例演示了如何使用 get-key-pair。

Amazon CLI

获取有关密钥对的信息

以下 get-key-pair 示例显示有关指定密钥对的详细信息。

```
aws lightsail get-key-pair \
    --key-pair-name MyKey1
```

输出：

```
{
    "keyPair": {
        "name": "MyKey1",
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:KeyPair/19a4efdf-3054-43d6-91fd-eEXAMPLE21bf",
        "supportCode": "6EXAMPLE3362/MyKey1",
        "createdAt": 1571255026.975,
```

```
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "KeyPair",
        "tags": [],
        "fingerprint": "00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff:gg:hh:ii:jj"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetKeyPair。

get-key-pairs

以下代码示例演示了如何使用 get-key-pairs。

Amazon CLI

获取有关所有密钥对的信息

以下 get-key-pairs 示例显示有关已配置的 Amazon 区域中所有密钥对的详细信息。

```
aws lightsail get-key-pairs
```

输出：

```
{
    "keyPairs": [
        {
            "name": "MyKey1",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:KeyPair/19a4efdf-3054-43d6-91fd-eEXAMPLE21bf",
            "supportCode": "6EXAMPLE3362/MyKey1",
            "createdAt": 1571255026.975,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "KeyPair",
            "tags": [],
            "fingerprint":
 "00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff:gg:hh:ii:jj"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetKeyPairs。

get-load-balancer-tls-certificates

以下代码示例演示了如何使用 get-load-balancer-tls-certificates。

Amazon CLI

获取有关负载均衡器的 TLS 证书的信息

以下 get-load-balancer-tls-certificates 示例显示指定负载均衡器的 TLS 证书的详细信息。

```
aws lightsail get-load-balancer-tls-certificates \
    --load-balancer-name LoadBalancer-1
```

输出：

```
{
    "tlsCertificates": [
        {
            "name": "example-com",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:LoadBalancerTlsCertificate/d7bf4643-6a02-4cd4-b3c4-
fEXAMPLE9b4d",
            "supportCode": "6EXAMPLE3362/arn:aws:acm:us-
west-2:333322221111:certificate/9af8e32c-a54e-4a67-8c63-cEXAMPLEb314",
            "createdAt": 1571678025.3,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "LoadBalancerTlsCertificate",
            "loadBalancerName": "LoadBalancer-1",
            "isAttached": false,
            "status": "ISSUED",
            "domainName": "example.com",
            "domainValidationRecords": [
                {
```

```
                    "name": "_dEXAMPLE4ede046a0319eb44a4eb3cbc.example.com.",
                    "type": "CNAME",
                    "value": "_bEXAMPLE0899fb7b6bf79d9741d1a383.hkvuiqjoua.acm-
validations.aws.",
                    "validationStatus": "SUCCESS",
                    "domainName": "example.com"
                }
            ],
            "issuedAt": 1571678070.0,
            "issuer": "Amazon",
            "keyAlgorithm": "RSA-2048",
            "notAfter": 1605960000.0,
            "notBefore": 1571616000.0,
            "serial": "00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff",
            "signatureAlgorithm": "SHA256WITHRSA",
            "subject": "CN=example.com",
            "subjectAlternativeNames": [
                "example.com"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoadBalancerTlsCertificates。

get-load-balancer

以下代码示例演示了如何使用 get-load-balancer。

Amazon CLI

获取负载均衡器的信息

以下 get-load-balancer 示例显示指定负载均衡器的详细信息。

```
aws lightsail get-load-balancer \
    --load-balancer-name LoadBalancer-1
```

输出：

```
{
    "loadBalancer": {
        "name": "LoadBalancer-1",
```

```
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:LoadBalancer/40486b2b-1ad0-4152-83e4-cEXAMPLE6f4b",
        "supportCode": "6EXAMPLE3362/arn:aws:elasticloadbalancing:us-
west-2:333322221111:loadbalancer/app/
bEXAMPLE128cb59d86f946a9395dd304/1EXAMPLE8dd9d77e",
        "createdAt": 1571677906.723,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "LoadBalancer",
        "tags": [],
        "dnsName": "bEXAMPLE128cb59d86f946a9395dd304-1486911371.us-
west-2.elb.amazonaws.com",
        "state": "active",
        "protocol": "HTTP",
        "publicPorts": [
            80
        ],
        "healthCheckPath": "/",
        "instancePort": 80,
        "instanceHealthSummary": [
            {
                "instanceName": "MEAN-3",
                "instanceHealth": "healthy"
            },
            {
                "instanceName": "MEAN-1",
                "instanceHealth": "healthy"
            },
            {
                "instanceName": "MEAN-2",
                "instanceHealth": "healthy"
            }
        ],
        "tlsCertificateSummaries": [
            {
                "name": "example-com",
                "isAttached": false
            }
        ],
        "configurationOptions": {
            "SessionStickinessEnabled": "false",
            "SessionStickiness_LB_CookieDurationSeconds": "86400"
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoadBalancer。

get-load-balancers

以下代码示例演示了如何使用 get-load-balancers。

Amazon CLI

获取所有负载均衡器的信息

以下 get-load-balancers 示例显示有关已配置的 Amazon 区域中所有负载均衡器的详细信息。

```
aws lightsail get-load-balancers
```

输出：

```
{
    "loadBalancers": [
        {
            "name": "LoadBalancer-1",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:LoadBalancer/40486b2b-1ad0-4152-83e4-cEXAMPLE6f4b",
            "supportCode": "6EXAMPLE3362/arn:aws:elasticloadbalancing:us-
west-2:333322221111:loadbalancer/app/
bEXAMPLE128cb59d86f946a9395dd304/1EXAMPLE8dd9d77e",
            "createdAt": 1571677906.723,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "LoadBalancer",
            "tags": [],
            "dnsName": "bEXAMPLE128cb59d86f946a9395dd304-1486911371.us-
west-2.elb.amazonaws.com",
            "state": "active",
            "protocol": "HTTP",
            "publicPorts": [
```

```
                80
            ],
            "healthCheckPath": "/",
            "instancePort": 80,
            "instanceHealthSummary": [
                {
                    "instanceName": "MEAN-3",
                    "instanceHealth": "healthy"
                },
                {
                    "instanceName": "MEAN-1",
                    "instanceHealth": "healthy"
                },
                {
                    "instanceName": "MEAN-2",
                    "instanceHealth": "healthy"
                }
            ],
            "tlsCertificateSummaries": [
                {
                    "name": "example-com",
                    "isAttached": false
                }
            ],
            "configurationOptions": {
                "SessionStickinessEnabled": "false",
                "SessionStickiness_LB_CookieDurationSeconds": "86400"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoadBalancers。

get-operation

以下代码示例演示了如何使用 get-operation。

Amazon CLI

获取有关单个操作的信息

以下 get-operation 示例显示指定操作的详细信息。

```
aws lightsail get-operation \
    --operation-id e5700e8a-daf2-4b49-bc01-3EXAMPLE910a
```

输出：

```
{
    "operation": {
        "id": "e5700e8a-daf2-4b49-bc01-3EXAMPLE910a",
        "resourceName": "Instance-1",
        "resourceType": "Instance",
        "createdAt": 1571679872.404,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationType": "CreateInstance",
        "status": "Succeeded",
        "statusChangedAt": 1571679890.304
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOperation。

get-operations-for-resource

以下代码示例演示了如何使用 get-operations-for-resource。

Amazon CLI

获取资源的所有操作

以下 get-operations-for-resource 示例显示指定资源的所有操作的详细信息。

```
aws lightsail get-operations-for-resource \
    --resource-name LoadBalancer-1
```

输出：

```
{
    "operations": [
        {
```

```
            "id": "e2973046-43f8-4252-a4b4-9EXAMPLE69ce",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571678786.071,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MEAN-1",
            "operationType": "DetachInstancesFromLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1571679087.57
        },
        {
            "id": "2d742a18-0e7f-48c8-9705-3EXAMPLEf98a",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571678782.784,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MEAN-1",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1571678798.465
        },
        {
            "id": "6c700fcc-4246-40ab-952b-1EXAMPLEdac2",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571678775.297,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "MEAN-3",
            "operationType": "AttachInstancesToLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1571678842.806
        },
```

```
        ...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOperationsForResource。

get-operations

以下代码示例演示了如何使用 get-operations。

Amazon CLI

获取有关所有操作的信息

以下 get-operations 示例显示有关已配置的 Amazon 区域中所有操作的详细信息。

```
aws lightsail get-operations
```

输出：

```
{
    "operations": [
        {
            "id": "e5700e8a-daf2-4b49-bc01-3EXAMPLE910a",
            "resourceName": "Instance-1",
            "resourceType": "Instance",
            "createdAt": 1571679872.404,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "CreateInstance",
            "status": "Succeeded",
            "statusChangedAt": 1571679890.304
        },
        {
            "id": "701a3339-930e-4914-a9f9-7EXAMPLE68d7",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1571678786.072,
            "location": {
```

```
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "LoadBalancer-1",
            "operationType": "DetachInstancesFromLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1571679086.399
        },
        {
            "id": "e2973046-43f8-4252-a4b4-9EXAMPLE69ce",
            "resourceName": "LoadBalancer-1",
            "resourceType": "LoadBalancer",
            "createdAt": 1571678786.071,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "WordPress-1",
            "operationType": "DetachInstancesFromLoadBalancer",
            "status": "Succeeded",
            "statusChangedAt": 1571679087.57
        },
        ...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOperations。

get-regions

以下代码示例演示了如何使用 `get-regions`。

Amazon CLI

获取 Amazon Lightsail 的所有 Amazon 区域

以下 `get-regions` 示例显示有关 Amazon Lightsail 的所有 Amazon 区域的详细信息。

```
aws lightsail get-regions
```

输出：

```
{
    "regions": [
        {
            "continentCode": "NA",
            "description": "This region is recommended to serve users in the eastern
 United States",
            "displayName": "Virginia",
            "name": "us-east-1",
            "availabilityZones": [],
            "relationalDatabaseAvailabilityZones": []
        },
        {
            "continentCode": "NA",
            "description": "This region is recommended to serve users in the eastern
 United States",
            "displayName": "Ohio",
            "name": "us-east-2",
            "availabilityZones": [],
            "relationalDatabaseAvailabilityZones": []
        },
        {
            "continentCode": "NA",
            "description": "This region is recommended to serve users in the
 northwestern United States, Alaska, and western Canada",
            "displayName": "Oregon",
            "name": "us-west-2",
            "availabilityZones": [],
            "relationalDatabaseAvailabilityZones": []
        },
        ...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRegions。

get-relational-database-blueprints

以下代码示例演示了如何使用 get-relational-database-blueprints。

Amazon CLI

获取新的关系数据库的蓝图

以下 `get-relational-database-blueprints` 示例显示可用于在 Amazon Lightsail 中创建新的关系数据库的所有可用关系数据库蓝图的详细信息。

```
aws lightsail get-relational-database-blueprints
```

输出：

```
{
    "blueprints": [
        {
            "blueprintId": "mysql_5_6",
            "engine": "mysql",
            "engineVersion": "5.6.44",
            "engineDescription": "MySQL Community Edition",
            "engineVersionDescription": "MySQL 5.6.44",
            "isEngineDefault": false
        },
        {
            "blueprintId": "mysql_5_7",
            "engine": "mysql",
            "engineVersion": "5.7.26",
            "engineDescription": "MySQL Community Edition",
            "engineVersionDescription": "MySQL 5.7.26",
            "isEngineDefault": true
        },
        {
            "blueprintId": "mysql_8_0",
            "engine": "mysql",
            "engineVersion": "8.0.16",
            "engineDescription": "MySQL Community Edition",
            "engineVersionDescription": "MySQL 8.0.16",
            "isEngineDefault": false
        },
        {
            "blueprintId": "postgres_9_6",
            "engine": "postgres",
            "engineVersion": "9.6.15",
            "engineDescription": "PostgreSQL",
            "engineVersionDescription": "PostgreSQL 9.6.15-R1",
```

```
            "isEngineDefault": false
        },
        {
            "blueprintId": "postgres_10",
            "engine": "postgres",
            "engineVersion": "10.10",
            "engineDescription": "PostgreSQL",
            "engineVersionDescription": "PostgreSQL 10.10-R1",
            "isEngineDefault": false
        },
        {
            "blueprintId": "postgres_11",
            "engine": "postgres",
            "engineVersion": "11.5",
            "engineDescription": "PostgreSQL",
            "engineVersionDescription": "PostgreSQL 11.5-R1",
            "isEngineDefault": true
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseBlueprints。

get-relational-database-bundles

以下代码示例演示了如何使用 get-relational-database-bundles。

Amazon CLI

获取新的关系数据库的捆绑包

以下 get-relational-database-bundles 示例显示可用于在 Amazon Lightsail 中创建新的关系数据库的所有可用关系数据库捆绑包的详细信息。请注意，响应中不包括非活动捆绑包，因为命令中未指定 --include-inactive 标志。您不能使用非活动捆绑包来创建新的关系数据库。

```
aws lightsail get-relational-database-bundles
```

输出：

```
{
    "bundles": [
        {
```

```
            "bundleId": "micro_2_0",
            "name": "Micro",
            "price": 15.0,
            "ramSizeInGb": 1.0,
            "diskSizeInGb": 40,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "micro_ha_2_0",
            "name": "Micro with High Availability",
            "price": 30.0,
            "ramSizeInGb": 1.0,
            "diskSizeInGb": 40,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "small_2_0",
            "name": "Small",
            "price": 30.0,
            "ramSizeInGb": 2.0,
            "diskSizeInGb": 80,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "small_ha_2_0",
            "name": "Small with High Availability",
            "price": 60.0,
            "ramSizeInGb": 2.0,
            "diskSizeInGb": 80,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
```

```
            "bundleId": "medium_2_0",
            "name": "Medium",
            "price": 60.0,
            "ramSizeInGb": 4.0,
            "diskSizeInGb": 120,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "medium_ha_2_0",
            "name": "Medium with High Availability",
            "price": 120.0,
            "ramSizeInGb": 4.0,
            "diskSizeInGb": 120,
            "transferPerMonthInGb": 100,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "large_2_0",
            "name": "Large",
            "price": 115.0,
            "ramSizeInGb": 8.0,
            "diskSizeInGb": 240,
            "transferPerMonthInGb": 200,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        },
        {
            "bundleId": "large_ha_2_0",
            "name": "Large with High Availability",
            "price": 230.0,
            "ramSizeInGb": 8.0,
            "diskSizeInGb": 240,
            "transferPerMonthInGb": 200,
            "cpuCount": 2,
            "isEncrypted": true,
            "isActive": true
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Lightsail 开发人员指南》中的在 Amazon Lightsail 中创建数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseBundles。

get-relational-database-events

以下代码示例演示了如何使用 get-relational-database-events。

Amazon CLI

获取关系数据库的事件

以下 get-relational-database-events 示例显示指定关系数据库在过去 17 小时（1020 分钟）内发生的事件的详细信息。

```
aws lightsail get-relational-database-events \
    --relational-database-name Database-1 \
    --duration-in-minutes 1020
```

输出：

```
{
    "relationalDatabaseEvents": [
        {
            "resource": "Database-1",
            "createdAt": 1571654146.553,
            "message": "Backing up Relational Database",
            "eventCategories": [
                "backup"
            ]
        },
        {
            "resource": "Database-1",
            "createdAt": 1571654249.98,
            "message": "Finished Relational Database backup",
            "eventCategories": [
                "backup"
            ]
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseEvents。

get-relational-database-log-events

以下代码示例演示了如何使用 get-relational-database-log-events。

Amazon CLI

获取关系数据库的日志事件

以下 get-relational-database-log-events 示例显示关系数据库 Database1 在 1570733176 和 1571597176 期间指定日志的详细信息。返回的信息配置为从 head 开始。

我们建议您使用 unix 时间转换器来识别开始和结束时间。

```
aws lightsail get-relational-database-log-events \
    --relational-database-name Database1 \
    --log-stream-name error \
    --start-from-head \
    --start-time 1570733176 \
    --end-time 1571597176
```

输出：

```
{
    "resourceLogEvents": [
        {
            "createdAt": 1570820267.0,
            "message": "2019-10-11 18:57:47 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Name or service not known"
        },
        {
            "createdAt": 1570860974.0,
            "message": "2019-10-12 06:16:14 20969 [Warning] IP address '8192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        {
            "createdAt": 1570860977.0,
            "message": "2019-10-12 06:16:17 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
```

```
        {
            "createdAt": 1570860979.0,
            "message": "2019-10-12 06:16:19 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        {
            "createdAt": 1570860981.0,
            "message": "2019-10-12 06:16:21 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        {
            "createdAt": 1570860982.0,
            "message": "2019-10-12 06:16:22 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        {
            "createdAt": 1570860984.0,
            "message": "2019-10-12 06:16:24 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        {
            "createdAt": 1570860986.0,
            "message": "2019-10-12 06:16:26 20969 [Warning] IP address '192.0.2.0'
 could not be resolved: Temporary failure in name resolution"
        },
        ...
        }
    ],
    "nextBackwardToken":
 "eEXAMPLEZXJUZXh0IjoiZnRWb3F3cUpRSlQ5NndMYThxelRUZlFhR3J6c2dKWEEvM2kvajZMZzVVVWpqRDN0YjFXTj
    "nextForwardToken":
 "eEXAMPLEZXJUZXh0IjoiT09Lb0Z6ZFRJbHhaNEQ5N2tPbkkwRmwwNUxPZjFTbFFwUklQbzlSaWgvMWVXbEk4aG56VH
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseLogEvents。

get-relational-database-log-streams

以下代码示例演示了如何使用 get-relational-database-log-streams。

Amazon CLI

获取关系数据库的日志流

以下 get-relational-database-log-streams 示例返回指定关系数据库的所有可用日志流。

```
aws lightsail get-relational-database-log-streams \
--relational-database-name Database1
```

输出：

```
{
    "logStreams": [
        "audit",
        "error",
        "general",
        "slowquery"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseLogStreams。

get-relational-database-master-user-password

以下代码示例演示了如何使用 get-relational-database-master-user-password。

Amazon CLI

获取关系数据库的主用户密码

以下 get-relational-database-master-user-password 示例返回有关指定关系数据库的主用户密码的信息。

```
aws lightsail get-relational-database-master-user-password \
    --relational-database-name Database-1
```

输出：

```
{
    "masterUserPassword": "VEXAMPLEec.9qvx,_t<)Wkf)kwboM,>2",
    "createdAt": 1571259453.959
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseMasterUserPassword。

get-relational-database-metric-data

以下代码示例演示了如何使用 get-relational-database-metric-data。

Amazon CLI

获取关系数据库的指标数据

以下 get-relational-database-metric-data 示例返回关系数据库 Database1 在 1570733176 和 1571597176 期间 24 小时（86400 秒）内指标 DatabaseConnections 的总数。

我们建议您使用 unix 时间转换器来识别开始和结束时间。

```
aws lightsail get-relational-database-metric-data \
    --relational-database-name Database1 \
    --metric-name DatabaseConnections \
    --period 86400 \
    --start-time 1570733176 \
    --end-time 1571597176 \
    --unit Count \
    --statistics Sum
```

输出：

```
{
    "metricName": "DatabaseConnections",
    "metricData": [
        {
            "sum": 1.0,
            "timestamp": 1571510760.0,
            "unit": "Count"
        },
        {
            "sum": 1.0,
            "timestamp": 1570733160.0,
            "unit": "Count"
        },
```

```
        {
            "sum": 1.0,
            "timestamp": 1570992360.0,
            "unit": "Count"
        },
        {
            "sum": 0.0,
            "timestamp": 1571251560.0,
            "unit": "Count"
        },
        {
            "sum": 721.0,
            "timestamp": 1570819560.0,
            "unit": "Count"
        },
        {
            "sum": 1.0,
            "timestamp": 1571078760.0,
            "unit": "Count"
        },
        {
            "sum": 2.0,
            "timestamp": 1571337960.0,
            "unit": "Count"
        },
        {
            "sum": 684.0,
            "timestamp": 1570905960.0,
            "unit": "Count"
        },
        {
            "sum": 0.0,
            "timestamp": 1571165160.0,
            "unit": "Count"
        },
        {
            "sum": 1.0,
            "timestamp": 1571424360.0,
            "unit": "Count"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseMetricData。

get-relational-database-parameters

以下代码示例演示了如何使用 get-relational-database-parameters。

Amazon CLI

获取关系数据库的参数

以下 get-relational-database-parameters 示例返回指定关系数据库的所有可用参数的信息。

```
aws lightsail get-relational-database-parameters \
    --relational-database-name Database-1
```

输出：

```
{
    "parameters": [
        {
            "allowedValues": "0,1",
            "applyMethod": "pending-reboot",
            "applyType": "dynamic",
            "dataType": "boolean",
            "description": "Automatically set all granted roles as active after the
 user has authenticated successfully.",
            "isModifiable": true,
            "parameterName": "activate_all_roles_on_login",
            "parameterValue": "0"
        },
        {
            "allowedValues": "0,1",
            "applyMethod": "pending-reboot",
            "applyType": "static",
            "dataType": "boolean",
            "description": "Controls whether user-defined functions that have only
 an xxx symbol for the main function can be loaded",
            "isModifiable": false,
            "parameterName": "allow-suspicious-udfs"
        },
        {
            "allowedValues": "0,1",
            "applyMethod": "pending-reboot",
            "applyType": "dynamic",
```

```
            "dataType": "boolean",
            "description": "Sets the autocommit mode",
            "isModifiable": true,
            "parameterName": "autocommit"
        },
        {
            "allowedValues": "0,1",
            "applyMethod": "pending-reboot",
            "applyType": "static",
            "dataType": "boolean",
            "description": "Controls whether the server autogenerates SSL key and
 certificate files in the data directory, if they do not already exist.",
            "isModifiable": false,
            "parameterName": "auto_generate_certs"
        },
        ...
        }
    ]
}
```

有关更多信息，请参阅《Lightsail 开发人员指南》中的在 Amazon Lightsail 中更新数据库参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseParameters。

get-relational-database-snapshot

以下代码示例演示了如何使用 get-relational-database-snapshot。

Amazon CLI

获取有关关系数据库快照的信息

以下 get-relational-database-snapshot 示例显示指定关系数据库快照的详细信息。

```
aws lightsail get-relational-database-snapshot \
    --relational-database-snapshot-name Database-1-1571350042
```

输出：

```
{
    "relationalDatabaseSnapshot": {
        "name": "Database-1-1571350042",
```

```
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabaseSnapshot/0389bbad-4b85-4c3d-9EXAMPLEaee3643d2",
        "supportCode": "6EXAMPLE3362/ls-8EXAMPLE2ba7ad041451946fafc2ad19cfbd9eb2",
        "createdAt": 1571350046.238,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "RelationalDatabaseSnapshot",
        "tags": [],
        "engine": "mysql",
        "engineVersion": "8.0.16",
        "sizeInGb": 40,
        "state": "available",
        "fromRelationalDatabaseName": "Database-1",
        "fromRelationalDatabaseArn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",
        "fromRelationalDatabaseBundleId": "micro_1_0",
        "fromRelationalDatabaseBlueprintId": "mysql_8_0"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseSnapshot。

get-relational-database-snapshots

以下代码示例演示了如何使用 get-relational-database-snapshots。

Amazon CLI

获取有关所有关系数据库快照的信息

以下 get-relational-database-snapshots 示例显示有关已配置的 Amazon 区域中所有关系数据库快照的详细信息。

```
aws lightsail get-relational-database-snapshots
```

输出：

```
{
    "relationalDatabaseSnapshots": [
        {
```

```
            "name": "Database-1-1571350042",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabaseSnapshot/0389bbad-4b85-4c3d-9861-6EXAMPLE43d2",
            "supportCode": "6EXAMPLE3362/
ls-8EXAMPLE2ba7ad041451946fafc2ad19cfbd9eb2",
            "createdAt": 1571350046.238,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "RelationalDatabaseSnapshot",
            "tags": [],
            "engine": "mysql",
            "engineVersion": "8.0.16",
            "sizeInGb": 40,
            "state": "available",
            "fromRelationalDatabaseName": "Database-1",
            "fromRelationalDatabaseArn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",
            "fromRelationalDatabaseBundleId": "micro_1_0",
            "fromRelationalDatabaseBlueprintId": "mysql_8_0"
        },
        {
            "name": "Database1-Console",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabaseSnapshot/8b94136e-06ec-4b1a-
a3fb-5EXAMPLEe1e9",
            "supportCode": "6EXAMPLE3362/
ls-9EXAMPLE14b000d34c8d1c432734e137612d5b5c",
            "createdAt": 1571249981.025,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "RelationalDatabaseSnapshot",
            "tags": [
                {
                    "key": "test"
                }
            ],
            "engine": "mysql",
            "engineVersion": "5.6.44",
            "sizeInGb": 40,
            "state": "available",
```

```
            "fromRelationalDatabaseName": "Database1",
            "fromRelationalDatabaseArn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/a6161cb7-4535-4f16-9dcf-8EXAMPLE3d4e",
            "fromRelationalDatabaseBundleId": "micro_1_0",
            "fromRelationalDatabaseBlueprintId": "mysql_5_6"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabaseSnapshots。

get-relational-database

以下代码示例演示了如何使用 get-relational-database。

Amazon CLI

获取有关关系数据库的信息

以下 get-relational-database 示例显示指定关系数据库的详细信息。

```
aws lightsail get-relational-database \
    --relational-database-name Database-1
```

输出：

```
{
    "relationalDatabase": {
        "name": "Database-1",
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",
        "supportCode": "6EXAMPLE3362/ls-9EXAMPLE8ad863723b62cc8901a8aa6e794ae0d2",
        "createdAt": 1571259453.795,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
        "resourceType": "RelationalDatabase",
        "tags": [],
        "relationalDatabaseBlueprintId": "mysql_8_0",
        "relationalDatabaseBundleId": "micro_1_0",
        "masterDatabaseName": "dbmaster",
        "hardware": {
```

```
            "cpuCount": 1,
            "diskSizeInGb": 40,
            "ramSizeInGb": 1.0
        },
        "state": "available",
        "backupRetentionEnabled": false,
        "pendingModifiedValues": {},
        "engine": "mysql",
        "engineVersion": "8.0.16",
        "masterUsername": "dbmasteruser",
        "parameterApplyStatus": "in-sync",
        "preferredBackupWindow": "10:01-10:31",
        "preferredMaintenanceWindow": "sat:11:14-sat:11:44",
        "publiclyAccessible": true,
        "masterEndpoint": {
            "port": 3306,
            "address": "ls-9EXAMPLE8ad863723b62ccEXAMPLEa6e794ae0d2.czowadgeezqi.us-
west-2.rds.amazonaws.com"
        },
        "pendingMaintenanceActions": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabase。

get-relational-databases

以下代码示例演示了如何使用 get-relational-databases。

Amazon CLI

获取有关所有关系数据库的信息

以下 get-relational-databases 示例显示有关已配置的 Amazon 区域中所有关系数据库的详细信息。

```
aws lightsail get-relational-databases
```

输出：

```
{
    "relationalDatabases": [
```

```
        {
            "name": "MySQL",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/8529020c-3ab9-4d51-92af-5EXAMPLE8979",
            "supportCode": "6EXAMPLE3362/
ls-3EXAMPLEa995d8c3b06b4501356e5f2f28e1aeba",
            "createdAt": 1554306019.155,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "resourceType": "RelationalDatabase",
            "tags": [],
            "relationalDatabaseBlueprintId": "mysql_8_0",
            "relationalDatabaseBundleId": "micro_1_0",
            "masterDatabaseName": "dbmaster",
            "hardware": {
                "cpuCount": 1,
                "diskSizeInGb": 40,
                "ramSizeInGb": 1.0
            },
            "state": "available",
            "backupRetentionEnabled": true,
            "pendingModifiedValues": {},
            "engine": "mysql",
            "engineVersion": "8.0.15",
            "latestRestorableTime": 1571686200.0,
            "masterUsername": "dbmasteruser",
            "parameterApplyStatus": "in-sync",
            "preferredBackupWindow": "07:51-08:21",
            "preferredMaintenanceWindow": "tue:12:18-tue:12:48",
            "publiclyAccessible": true,
            "masterEndpoint": {
                "port": 3306,
                "address":
 "ls-3EXAMPLEa995d8c3b06b4501356e5f2fEXAMPLEa.czowadgeezqi.us-
west-2.rds.amazonaws.com"
            },
            "pendingMaintenanceActions": []
        },
        {
            "name": "Postgres",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:RelationalDatabase/
e9780b6b-d0ab-4af2-85f1-1EXAMPLEac68",
```

```
            "supportCode": "6EXAMPLE3362/
ls-3EXAMPLEb4fffb5cec056220c734713e14bd5fcd",
            "createdAt": 1554306000.814,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "resourceType": "RelationalDatabase",
            "tags": [],
            "relationalDatabaseBlueprintId": "postgres_11",
            "relationalDatabaseBundleId": "micro_1_0",
            "masterDatabaseName": "dbmaster",
            "hardware": {
                "cpuCount": 1,
                "diskSizeInGb": 40,
                "ramSizeInGb": 1.0
            },
            "state": "available",
            "backupRetentionEnabled": true,
            "pendingModifiedValues": {},
            "engine": "postgres",
            "engineVersion": "11.1",
            "latestRestorableTime": 1571686339.0,
            "masterUsername": "dbmasteruser",
            "parameterApplyStatus": "in-sync",
            "preferredBackupWindow": "06:19-06:49",
            "preferredMaintenanceWindow": "sun:10:19-sun:10:49",
            "publiclyAccessible": false,
            "masterEndpoint": {
                "port": 5432,
                "address":
 "ls-3EXAMPLEb4fffb5cec056220c734713eEXAMPLEd.czowadgeezqi.us-
west-2.rds.amazonaws.com"
            },
            "pendingMaintenanceActions": []
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRelationalDatabases。

get-static-ip

以下代码示例演示了如何使用 get-static-ip。

Amazon CLI

获取有关静态 IP 的信息

以下 get-static-ip 示例显示有关指定静态 IP 的详细信息。

```
aws lightsail get-static-ip \
    --static-ip-name StaticIp-1
```

输出：

```
{
    "staticIp": {
        "name": "StaticIp-1",
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:StaticIp/2257cd76-1f0e-4ac0-82e2-2EXAMPLE23ad",
        "supportCode": "6EXAMPLE3362/192.0.2.0",
        "createdAt": 1571071325.076,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "StaticIp",
        "ipAddress": "192.0.2.0",
        "isAttached": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStaticIp。

get-static-ips

以下代码示例演示了如何使用 get-static-ips。

Amazon CLI

获取有关所有静态 IP 的信息

以下 get-static-ips 示例显示有关已配置的 Amazon 区域中所有静态 IP 的详细信息。

```
aws lightsail get-static-ips
```

输出：

```
{
    "staticIps": [
        {
            "name": "StaticIp-1",
            "arn": "arn:aws:lightsail:us-
west-2:111122223333:StaticIp/2257cd76-1f0e-4ac0-8EXAMPLE16f9423ad",
            "supportCode": "6EXAMPLE3362/192.0.2.0",
            "createdAt": 1571071325.076,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "StaticIp",
            "ipAddress": "192.0.2.0",
            "isAttached": false
        },
        {
            "name": "StaticIP-2",
            "arn": "arn:aws:lightsail:us-west-2:111122223333:StaticIp/c61edb40-
e5f0-4fd6-ae7c-8EXAMPLE19f8",
            "supportCode": "6EXAMPLE3362/192.0.2.2",
            "createdAt": 1568305385.681,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "resourceType": "StaticIp",
            "ipAddress": "192.0.2.2",
            "attachedTo": "WordPress-1",
            "isAttached": true
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetStaticIps。

is-vpc-peered

以下代码示例演示了如何使用 is-vpc-peered。

Amazon CLI

识别您的 Amazon Lightsail 虚拟私有云是否对等连接

以下 is-vpc-peered 示例返回指定 Amazon 区域的 Amazon Lightsail 虚拟私有云（VPC）的对等连接状态。

```
aws lightsail is-vpc-peered \
    --region us-west-2
```

输出：

```
{
    "isPeered": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IsVpcPeered。

open-instance-public-ports

以下代码示例演示了如何使用 open-instance-public-ports。

Amazon CLI

为实例打开防火墙端口

以下 open-instance-public-ports 示例在指定实例上打开 TCP 端口 22。

```
aws lightsail open-instance-public-ports \
    --instance-name MEAN-2 \
    --port-info fromPort=22,protocol=TCP,toPort=22
```

输出：

```
{
```

```
    "operation": {
        "id": "719744f0-a022-46f2-9f11-6EXAMPLE4642",
        "resourceName": "MEAN-2",
        "resourceType": "Instance",
        "createdAt": 1571072906.849,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationDetails": "22/tcp",
        "operationType": "OpenInstancePublicPorts",
        "status": "Succeeded",
        "statusChangedAt": 1571072906.849
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 OpenInstancePublicPorts。

peer-vpc

以下代码示例演示了如何使用 peer-vpc。

Amazon CLI

对等连接 Amazon Lightsail 虚拟私有云

以下 peer-vpc 示例为指定 Amazon 区域对等连接 Amazon Lightsail 虚拟私有云（VPC）。

```
aws lightsail peer-vpc \
    --region us-west-2
```

输出：

```
{
    "operation": {
        "id": "787e846a-54ac-497f-bce2-9EXAMPLE5d91",
        "resourceName": "vpc-0EXAMPLEa5261efb3",
        "resourceType": "PeeredVpc",
        "createdAt": 1571694233.104,
        "location": {
            "availabilityZone": "all",
```

```
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationDetails": "vpc-e2b3eb9b",
        "operationType": "PeeredVpc",
        "status": "Succeeded",
        "statusChangedAt": 1571694233.104
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PeerVpc。

reboot-instance

以下代码示例演示了如何使用 reboot-instance。

Amazon CLI

重启实例

以下 reboot-instance 示例重启指定实例。

```
aws lightsail reboot-instance \
    --instance-name MEAN-1
```

输出：

```
{
    "operations": [
        {
            "id": "2b679f1c-8b71-4bb4-8e97-8EXAMPLEed93",
            "resourceName": "MEAN-1",
            "resourceType": "Instance",
            "createdAt": 1571694445.49,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationDetails": "",
            "operationType": "RebootInstance",
            "status": "Succeeded",
```

```
            "statusChangedAt": 1571694445.49
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootInstance。

reboot-relational-database

以下代码示例演示了如何使用 reboot-relational-database。

Amazon CLI

重启关系数据库

以下 reboot-relational-database 示例重启指定的关系数据库。

```
aws lightsail reboot-relational-database \
    --relational-database-name Database-1
```

输出：

```
{
    "operations": [
        {
            "id": "e4c980c0-3137-496c-9c91-1EXAMPLEdec2",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1571694532.91,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationDetails": "",
            "operationType": "RebootRelationalDatabase",
            "status": "Started",
            "statusChangedAt": 1571694532.91
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootRelationalDatabase。

release-static-ip

以下代码示例演示了如何使用 release-static-ip。

Amazon CLI

删除静态 IP

以下 release-static-ip 示例删除指定的静态 IP。

```
aws lightsail release-static-ip \
    --static-ip-name StaticIp-1
```

输出：

```
{
    "operations": [
        {
            "id": "e374c002-dc6d-4c7f-919f-2EXAMPLE13ce",
            "resourceName": "StaticIp-1",
            "resourceType": "StaticIp",
            "createdAt": 1571694962.003,
            "location": {
                "availabilityZone": "all",
                "regionName": "us-west-2"
            },
            "isTerminal": true,
            "operationType": "ReleaseStaticIp",
            "status": "Succeeded",
            "statusChangedAt": 1571694962.003
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReleaseStaticIp。

start-instance

以下代码示例演示了如何使用 start-instance。

Amazon CLI

启动实例

以下 start-instance 示例启动指定的实例。

```
aws lightsail start-instance \
    --instance-name WordPress-1
```

输出：

```
{
    "operations": [
        {
            "id": "f88d2a93-7cea-4165-afce-2d688cb18f23",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1571695583.463,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "StartInstance",
            "status": "Started",
            "statusChangedAt": 1571695583.463
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartInstance。

start-relational-database

以下代码示例演示了如何使用 start-relational-database。

Amazon CLI

启动关系数据库

以下 start-relational-database 示例启动指定的关系数据库。

```
aws lightsail start-relational-database \
    --relational-database-name Database-1
```

输出：

```
{
    "operations": [
        {
            "id": "4d5294ec-a38a-4fda-9e37-aEXAMPLE0d24",
            "resourceName": "Database-1",
            "resourceType": "RelationalDatabase",
            "createdAt": 1571695998.822,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "StartRelationalDatabase",
            "status": "Started",
            "statusChangedAt": 1571695998.822
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartRelationalDatabase。

stop-instance

以下代码示例演示了如何使用 stop-instance。

Amazon CLI

停止实例

以下 stop-instance 示例停止指定的实例。

```
aws lightsail stop-instance \
--instance-name WordPress-1
```

输出：

```
{
    "operations": [
        {
            "id": "265357e2-2943-4d51-888a-1EXAMPLE7585",
            "resourceName": "WordPress-1",
            "resourceType": "Instance",
            "createdAt": 1571695471.134,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "StopInstance",
            "status": "Started",
            "statusChangedAt": 1571695471.134
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopInstance。

stop-relational-database

以下代码示例演示了如何使用 stop-relational-database。

Amazon CLI

停止关系数据库

以下 stop-relational-database 示例停止指定的关系数据库。

```
aws lightsail stop-relational-database \
    --relational-database-name Database-1
```

输出：

```
{
    "operations": [
        {
            "id": "cc559c19-4adb-41e4-b75b-5EXAMPLE4e61",
            "resourceName": "Database-1",
```

```
            "resourceType": "RelationalDatabase",
            "createdAt": 1571695526.29,
            "location": {
                "availabilityZone": "us-west-2a",
                "regionName": "us-west-2"
            },
            "isTerminal": false,
            "operationType": "StopRelationalDatabase",
            "status": "Started",
            "statusChangedAt": 1571695526.29
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopRelationalDatabase。

unpeer-vpc

以下代码示例演示了如何使用 unpeer-vpc。

Amazon CLI

取消对等连接 Amazon Lightsail 虚拟私有云

以下 unpeer-vpc 示例为指定 Amazon 区域取消对等连接 Amazon Lightsail 虚拟私有云（VPC）。

```
aws lightsail unpeer-vpc \
    --region us-west-2
```

输出：

```
{
    "operation": {
        "id": "531aca64-7157-47ab-84c6-eEXAMPLEd898",
        "resourceName": "vpc-0EXAMPLEa5261efb3",
        "resourceType": "PeeredVpc",
        "createdAt": 1571694109.945,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
```

```
        "isTerminal": true,
        "operationDetails": "vpc-e2b3eb9b",
        "operationType": "UnpeeredVpc",
        "status": "Succeeded",
        "statusChangedAt": 1571694109.945
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnpeerVpc。

使用 Amazon CLI 的 Macie 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Macie 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-buckets`

以下代码示例演示了如何使用 `describe-buckets`。

Amazon CLI

查询 Amazon Macie 为您的账户监控和分析的一个或多个 S3 存储桶的数据

以下 `describe-buckets` 示例查询名称以 amzn-s3-demo-bucket 开头且位于当前 Amazon 区域内的所有 S3 存储桶的元数据。

```
aws macie2 describe-buckets \
    --criteria '{"bucketName":{"prefix":"amzn-s3-demo-bucket"}}'
```

输出：


```
{
    "buckets": [
        {
            "accountId": "123456789012",
            "allowsUnencryptedObjectUploads": "FALSE",
            "automatedDiscoveryMonitoringStatus": "MONITORED",
            "bucketArn": "arn:aws:s3:::amzn-s3-demo-bucket1",
            "bucketCreatedAt": "2020-05-18T19:54:00+00:00",
            "bucketName": "amzn-s3-demo-bucket1",
            "classifiableObjectCount": 13,
            "classifiableSizeInBytes": 1592088,
            "jobDetails": {
                "isDefinedInJob": "TRUE",
                "isMonitoredByJob": "TRUE",
                "lastJobId": "08c81dc4a2f3377fae45c9ddaEXAMPLE",
                "lastJobRunTime": "2024-08-19T14:55:30.270000+00:00"
            },
            "lastAutomatedDiscoveryTime": "2024-10-22T19:11:25.364000+00:00",
            "lastUpdated": "2024-10-25T07:33:06.337000+00:00",
            "objectCount": 13,
            "objectCountByEncryptionType": {
                "customerManaged": 0,
                "kmsManaged": 2,
                "s3Managed": 7,
                "unencrypted": 4,
                "unknown": 0
            },
            "publicAccess": {
                "effectivePermission": "NOT_PUBLIC",
                "permissionConfiguration": {
                    "accountLevelPermissions": {
                        "blockPublicAccess": {
                            "blockPublicAcls": true,
                            "blockPublicPolicy": true,
                            "ignorePublicAcls": true,
                            "restrictPublicBuckets": true
                        }
                    },
                    "bucketLevelPermissions": {
                        "accessControlList": {
                            "allowsPublicReadAccess": false,
                            "allowsPublicWriteAccess": false
```

```
                    },
                    "blockPublicAccess": {
                        "blockPublicAcls": true,
                        "blockPublicPolicy": true,
                        "ignorePublicAcls": true,
                        "restrictPublicBuckets": true
                    },
                    "bucketPolicy": {
                        "allowsPublicReadAccess": false,
                        "allowsPublicWriteAccess": false
                    }
                }
            }
        },
        "region": "us-west-2",
        "replicationDetails": {
            "replicated": false,
            "replicatedExternally": false,
            "replicationAccounts": []
        },
        "sensitivityScore": 78,
        "serverSideEncryption": {
            "kmsMasterKeyId": null,
            "type": "NONE"
        },
        "sharedAccess": "NOT_SHARED",
        "sizeInBytes": 4549746,
        "sizeInBytesCompressed": 0,
        "tags": [
            {
                "key": "Division",
                "value": "HR"
            },
            {
                "key": "Team",
                "value": "Recruiting"
            }
        ],
        "unclassifiableObjectCount": {
            "fileType": 0,
            "storageClass": 0,
            "total": 0
        },
        "unclassifiableObjectSizeInBytes": {
```

```
                "fileType": 0,
                "storageClass": 0,
                "total": 0
            },
            "versioning": true
        },
        {
            "accountId": "123456789012",
            "allowsUnencryptedObjectUploads": "TRUE",
            "automatedDiscoveryMonitoringStatus": "MONITORED",
            "bucketArn": "arn:aws:s3:::amzn-s3-demo-bucket2",
            "bucketCreatedAt": "2020-11-25T18:24:38+00:00",
            "bucketName": "amzn-s3-demo-bucket2",
            "classifiableObjectCount": 8,
            "classifiableSizeInBytes": 133810,
            "jobDetails": {
                "isDefinedInJob": "TRUE",
                "isMonitoredByJob": "FALSE",
                "lastJobId": "188d4f6044d621771ef7d65f2EXAMPLE",
                "lastJobRunTime": "2024-07-09T19:37:11.511000+00:00"
            },
            "lastAutomatedDiscoveryTime": "2024-10-24T19:11:25.364000+00:00",
            "lastUpdated": "2024-10-25T07:33:06.337000+00:00",
            "objectCount": 8,
            "objectCountByEncryptionType": {
                "customerManaged": 0,
                "kmsManaged": 0,
                "s3Managed": 8,
                "unencrypted": 0,
                "unknown": 0
            },
            "publicAccess": {
                "effectivePermission": "NOT_PUBLIC",
                "permissionConfiguration": {
                    "accountLevelPermissions": {
                        "blockPublicAccess": {
                            "blockPublicAcls": true,
                            "blockPublicPolicy": true,
                            "ignorePublicAcls": true,
                            "restrictPublicBuckets": true
                        }
                    },
                    "bucketLevelPermissions": {
                        "accessControlList": {
```

```
                        "allowsPublicReadAccess": false,
                        "allowsPublicWriteAccess": false
                    },
                    "blockPublicAccess": {
                        "blockPublicAcls": true,
                        "blockPublicPolicy": true,
                        "ignorePublicAcls": true,
                        "restrictPublicBuckets": true
                    },
                    "bucketPolicy": {
                        "allowsPublicReadAccess": false,
                        "allowsPublicWriteAccess": false
                    }
                }
            }
        },
        "region": "us-west-2",
        "replicationDetails": {
            "replicated": false,
            "replicatedExternally": false,
            "replicationAccounts": []
        },
        "sensitivityScore": 95,
        "serverSideEncryption": {
            "kmsMasterKeyId": null,
            "type": "AES256"
        },
        "sharedAccess": "EXTERNAL",
        "sizeInBytes": 175978,
        "sizeInBytesCompressed": 0,
        "tags": [
            {
                "key": "Division",
                "value": "HR"
            },
            {
                "key": "Team",
                "value": "Recruiting"
            }
        ],
        "unclassifiableObjectCount": {
            "fileType": 3,
            "storageClass": 0,
            "total": 3
```

```
            },
            "unclassifiableObjectSizeInBytes": {
                "fileType": 2999826,
                "storageClass": 0,
                "total": 2999826
            },
            "versioning": true
        }
    ]
}
```

有关更多信息，请参阅《Amazon Macie 用户指南》中的筛选您的 S3 存储桶库存。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBuckets。

使用 Amazon CLI 的 Amazon Managed Grafana 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Managed Grafana 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`list-workspaces`

以下代码示例演示了如何使用 `list-workspaces`。

Amazon CLI

在用户凭证指定的区域中列出账户的工作区

以下 `list-workspaces` 示例列出了账户所在区域的 Grafana 工作区。

```
aws grafana list-workspaces
```

输出：

```
{
    "workspaces": [
        {
            "authentication": {
                "providers": [
                    "AWS_SSO"
                ]
            },
            "created": "2022-04-04T16:20:21.796000-07:00",
            "description": "to test tags",
            "endpoint": "g-949e7b44df.grafana-workspace.us-east-1.amazonaws.com",
            "grafanaVersion": "8.2",
            "id": "g-949e7b44df",
            "modified": "2022-04-04T16:20:21.796000-07:00",
            "name": "testtag2",
            "notificationDestinations": [
                "SNS"
            ],
            "status": "ACTIVE"
        },
        {
            "authentication": {
                "providers": [
                    "AWS_SSO"
                ]
            },
            "created": "2022-04-20T10:22:15.115000-07:00",
            "description": "ww",
            "endpoint": "g-bffa51ed1b.grafana-workspace.us-east-1.amazonaws.com",
            "grafanaVersion": "8.2",
            "id": "g-bffa51ed1b",
            "modified": "2022-04-20T10:22:15.115000-07:00",
            "name": "ww",
            "notificationDestinations": [
                "SNS"
            ],
            "status": "ACTIVE"
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWorkspaces。

使用 Amazon CLI 的 MediaConnect 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaConnect 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-flow-outputs`

以下代码示例演示了如何使用 `add-flow-outputs`。

Amazon CLI

向流程中添加输出

以下 `add-flow-outputs` 示例向指定流程中添加输出。

```
aws mediaconnect add-flow-outputs \
--flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
--outputs Description='NYC
 stream',Destination=192.0.2.12,Name=NYC,Port=3333,Protocol=rtp-
fec,SmoothingLatency=100 Description='LA
 stream',Destination=203.0.113.9,Name=LA,Port=4444,Protocol=rtp-
fec,SmoothingLatency=100
```

输出：

```
{
    "Outputs": [
        {
            "Port": 3333,
            "OutputArn": "arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",
            "Name": "NYC",
            "Description": "NYC stream",
            "Destination": "192.0.2.12",
            "Transport": {
                "Protocol": "rtp-fec",
                "SmoothingLatency": 100
            }
        },
        {
            "Port": 4444,
            "OutputArn": "arn:aws:mediaconnect:us-
east-1:111122223333:output:2-987655dEF67hiJ89-c34de5fG678h:LA",
            "Name": "LA",
            "Description": "LA stream",
            "Destination": "203.0.113.9",
            "Transport": {
                "Protocol": "rtp-fec",
                "SmoothingLatency": 100
            }
        }
    ],
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的向流程中添加输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddFlowOutputs。

`create-flow`

以下代码示例演示了如何使用 `create-flow`。

Amazon CLI

创建流程

以下 create-flow 示例使用指定配置创建了一个流程。

```
aws mediaconnect create-flow \
    --availability-zone us-west-2c \
    --name ExampleFlow \
    --source Description='Example source,
 backup',IngestPort=1055,Name=BackupSource,Protocol=rtp,WhitelistCidr=10.24.34.0/23
```

输出：

```
{
    "Flow": {
        "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:ExampleFlow",
        "AvailabilityZone": "us-west-2c",
        "EgressIp": "54.245.71.21",
        "Source": {
            "IngestPort": 1055,
            "SourceArn": "arn:aws:mediaconnect:us-
east-1:123456789012:source:2-3aBC45dEF67hiJ89-c34de5fG678h:BackupSource",
            "Transport": {
                "Protocol": "rtp",
                "MaxBitrate": 80000000
            },
            "Description": "Example source, backup",
            "IngestIp": "54.245.71.21",
            "WhitelistCidr": "10.24.34.0/23",
            "Name": "mySource"
        },
        "Entitlements": [],
        "Name": "ExampleFlow",
        "Outputs": [],
        "Status": "STANDBY",
        "Description": "Example source, backup"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的创建流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFlow。

delete-flow

以下代码示例演示了如何使用 delete-flow。

Amazon CLI

删除流程

以下 delete-flow 示例删除指定流程。

```
aws mediaconnect delete-flow \
    --flow-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
    "Status": "DELETING"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的删除流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFlow。

describe-flow

以下代码示例演示了如何使用 describe-flow。

Amazon CLI

查看流程的详细信息

以下 describe-flow 示例显示指定流程的详细信息，如 ARN、可用区、状态、源、授权和输出。

```
aws mediaconnect describe-flow \
    --flow-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

输出：

```
{
    "Flow": {
        "EgressIp": "54.201.4.39",
        "AvailabilityZone": "us-west-2c",
        "Status": "ACTIVE",
        "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
        "Entitlements": [
            {
                "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:123456789012:entitlement:1-AaBb11CcDd22EeFf-34DE5fG12AbC:MyEntitlement",
                "Description": "Assign to this account",
                "Name": "MyEntitlement",
                "Subscribers": [
                    "444455556666"
                ]
            }
        ],
        "Description": "NYC awards show",
        "Name": "AwardsShow",
        "Outputs": [
            {
                "Port": 2355,
                "Name": "NYC",
                "Transport": {
                    "SmoothingLatency": 0,
                    "Protocol": "rtp-fec"
                },
                "OutputArn": "arn:aws:mediaconnect:us-
east-1:123456789012:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",
                "Destination": "192.0.2.0"
            },
            {
                "Port": 3025,
                "Name": "LA",
                "Transport": {
                    "SmoothingLatency": 0,
                    "Protocol": "rtp-fec"
                },
                "OutputArn": "arn:aws:mediaconnect:us-
east-1:123456789012:output:2-987655dEF67hiJ89-c34de5fG678h:LA",
                "Destination": "192.0.2.0"
```

```
            }
        ],
        "Source": {
            "IngestIp": "54.201.4.39",
            "SourceArn": "arn:aws:mediaconnect:us-
east-1:123456789012:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource",
            "Transport": {
                "MaxBitrate": 80000000,
                "Protocol": "rtp"
            },
            "IngestPort": 1069,
            "Description": "Saturday night show",
            "Name": "ShowSource",
            "WhitelistCidr": "10.24.34.0/23"
        }
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的查看流程的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFlow。

grant-flow-entitlements

以下代码示例演示了如何使用 grant-flow-entitlements。

Amazon CLI

授予对于流程的权利

以下 grant-flow-entitlements 示例授予对于指定现有流程的权利，以便您可以将您的内容与其他 Amazon 账户分享。

```
aws mediaconnect grant-flow-entitlements \
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
    --entitlements Description='For
 AnyCompany',Encryption={"Algorithm=aes128,KeyType=static-
key,RoleArn=arn:aws:iam::111122223333:role/MediaConnect-
ASM,SecretArn=arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1"},Name=AnyCompany_Entitlement,Subscribers=444455556666
 Description='For Example Corp',Name=ExampleCorp,Subscribers=777788889999
```

输出：

```
{
    "Entitlements": [
        {
            "Name": "AnyCompany_Entitlement",
            "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement",
            "Subscribers": [
                "444455556666"
            ],
            "Description": "For AnyCompany",
            "Encryption": {
                "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1",
                "Algorithm": "aes128",
                "RoleArn": "arn:aws:iam::111122223333:role/MediaConnect-ASM",
                "KeyType": "static-key"
            }
        },
        {
            "Name": "ExampleCorp",
            "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-3333cccc4444dddd-1111aaaa2222:ExampleCorp",
            "Subscribers": [
                "777788889999"
            ],
            "Description": "For Example Corp"
        }
    ],
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的授予对于流程的权利。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GrantFlowEntitlements。

`list-entitlements`

以下代码示例演示了如何使用 `list-entitlements`。

Amazon CLI

查看权利列表

以下 `list-entitlements` 示例显示授予账户的所有权利的列表。

```
aws mediaconnect list-entitlements
```

输出：

```
{
    "Entitlements": [
        {
            "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:MyEntitlement",
            "EntitlementName": "MyEntitlement"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect API 参考》中的 ListEntitlements。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEntitlements。

`list-flows`

以下代码示例演示了如何使用 `list-flows`。

Amazon CLI

查看流程列表

以下 `list-flows` 示例显示流程的列表。

```
aws mediaconnect list-flows
```

输出：

```
{
    "Flows": [
        {
            "Status": "STANDBY",
            "SourceType": "OWNED",
```

```
            "AvailabilityZone": "us-west-2a",
            "Description": "NYC awards show",
            "Name": "AwardsShow",
            "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow"
        },
        {
            "Status": "STANDBY",
            "SourceType": "OWNED",
            "AvailabilityZone": "us-west-2c",
            "Description": "LA basketball game",
            "Name": "BasketballGame",
            "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的查看流程列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFlows。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 MediaConnect 资源的标签

以下 `list-tags-for-resource` 示例显示与指定 MediaConnect 资源关联的标签键和值。

```
aws mediaconnect list-tags-for-resource \
    --resource-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame
```

输出：

```
{
    "Tags": {
        "region": "west",
        "stage": "prod"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect API 参考》中的 ListTagsForResource，TagResource，UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

remove-flow-output

以下代码示例演示了如何使用 remove-flow-output。

Amazon CLI

从流程中移除输出

以下 remove-flow-output 示例从指定流程中移除输出。

```
aws mediaconnect remove-flow-output \
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
    --output-arn arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
    "OutputArn": "arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的从流程中移除输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveFlowOutput。

revoke-flow-entitlement

以下代码示例演示了如何使用 revoke-flow-entitlement。

Amazon CLI

撤销权利

以下 revoke-flow-entitlement 示例撤销对于指定流程的权利。

```
aws mediaconnect revoke-flow-entitlement \
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
    --entitlement-arn arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
    "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的撤销权利。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeFlowEntitlement。

start-flow

以下代码示例演示了如何使用 start-flow。

Amazon CLI

启动流程

以下 start-flow 示例启动指定流程。

```
aws mediaconnect start-flow \
    --flow-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

此命令不生成任何输出。输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
    "Status": "STARTING"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的启动流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartFlow。

stop-flow

以下代码示例演示了如何使用 stop-flow。

Amazon CLI

停止流程

以下 stop-flow 示例停止指定流程。

```
aws mediaconnect stop-flow \
    --flow-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

输出：

```
{
    "Status": "STOPPING",
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow"
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的停止流程。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopFlow。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为 MediaConnect 资源添加标签

以下 tag-resource 示例为指定的 MediaConnect 资源添加一个带有键名称和值的标签。

```
aws mediaconnect tag-resource \
    --resource-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame
```

```
    --tags region=west
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaConnect API 参考》中的 ListTagsForResource，TagResource，UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从 MediaConnect 资源移除标签

以下 untag-resource 示例从 MediaConnect 资源移除具有指定键名和值的标签及其关联值。

```
aws mediaconnect untag-resource \
    --resource-arn arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame \
    --tag-keys region
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaConnect API 参考》中的 ListTagsForResource，TagResource，UntagResource。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-flow-entitlement

以下代码示例演示了如何使用 update-flow-entitlement。

Amazon CLI

更新权利

以下 update-flow-entitlement 示例使用新的描述和订阅用户更新了指定的权利。

```
aws mediaconnect update-flow-entitlement \
```

```
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
    --entitlement-arn arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement
 \
    --description 'For AnyCompany Affiliate' \
    --subscribers 777788889999
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
    "Entitlement": {
        "Name": "AnyCompany_Entitlement",
        "Description": "For AnyCompany Affiliate",
        "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement",
        "Encryption": {
            "KeyType": "static-key",
            "Algorithm": "aes128",
            "RoleArn": "arn:aws:iam::111122223333:role/MediaConnect-ASM",
            "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1"
        },
        "Subscribers": [
            "777788889999"
        ]
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的更新权利。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFlowEntitlement。

update-flow-output

以下代码示例演示了如何使用 update-flow-output。

Amazon CLI

更新流程上的输出

以下 update-flow-output 示例更新指定流程上的输出。

```
aws mediaconnect update-flow-output \
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
    --output-arn arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC \
    --port 3331
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
    "Output": {
        "Name": "NYC",
        "Port": 3331,
        "Description": "NYC stream",
        "Transport": {
            "Protocol": "rtp-fec",
            "SmoothingLatency": 100
        },
        "OutputArn": "arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",
        "Destination": "192.0.2.12"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的更新流程上的输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFlowOutput。

update-flow-source

以下代码示例演示了如何使用 update-flow-source。

Amazon CLI

更新现有流程的来源

以下 update-flow-source 示例更新现有流程的来源。

```
aws mediaconnect update-flow-source \
```

```
    --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow \
    --source-arn arn:aws:mediaconnect:us-
east-1:111122223333:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource \
    --description 'Friday night show' \
    --ingest-port 3344 \
    --protocol rtp-fec \
    --whitelist-cidr 10.24.34.0/23
```

输出：

```
{
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
    "Source": {
        "IngestIp": "34.210.136.56",
        "WhitelistCidr": "10.24.34.0/23",
        "Transport": {
            "Protocol": "rtp-fec"
        },
        "IngestPort": 3344,
        "Name": "ShowSource",
        "Description": "Friday night show",
        "SourceArn": "arn:aws:mediaconnect:us-
east-1:111122223333:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConnect 用户指南》中的更新流程的来源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFlowSource。

使用 Amazon CLI 的 MediaConvert 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaConvert 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

cancel-job

以下代码示例演示了如何使用 cancel-job。

Amazon CLI

取消队列中的一个作业

以下 cancel-job 示例取消 ID 为 1234567891234-abc123 的作业。如果服务已经开始处理该作业，则无法取消它。

```
aws mediaconvert cancel-job \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1 \
    --id 1234567891234-abc123
```

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelJob。

create-job-template

以下代码示例演示了如何使用 create-job-template。

Amazon CLI

创建作业模板

以下 create-job-template 示例使用您的系统上的文件 job-template.json 中指定的转码设置创建一个作业模板。

```
aws mediaconvert create-job-template \
```

```
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1 \
    --name JobTemplate1 \
    --cli-input-json file://~/job-template.json
```

如果您通过使用 get-job-template 然后修改文件的方法来创建作业模板 JSON 文件，则移除 JobTemplate 对象，但保留其中的 Settings 子对象。另外，请务必移除以下键-值对：LastUpdated、Arn、Type 和 CreatedAt。您可以在 JSON 文件或命令行中指定类别、描述、名称和队列。

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

如果您的请求成功，则服务会为您创建的作业模板返回 JSON 规范。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJobTemplate。

create-job

以下代码示例演示了如何使用 create-job。

Amazon CLI

创建作业

以下 create-job 示例使用在位于您发送命令的系统上的文件 job.json 中指定的设置创建转码作业。此 JSON 作业规范可以单独指定每个设置，引用作业模板或引用输出预设。

```
aws mediaconvert create-job \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1 \
    --cli-input-json file://~/job.json
```

您可以使用 Amazon Elemental MediaConvert 控制台生成 JSON 作业规范，方法是：选择您的作业设置，然后选择作业 部分底部的显示作业 JSON。

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

如果您的请求成功，则该服务将返回您在请求中发送的 JSON 作业规范。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJob。

create-preset

以下代码示例演示了如何使用 create-preset。

Amazon CLI

创建自定义输出预设

以下 create-preset 示例根据文件 preset.json 中指定的输出设置创建自定义输出预设。您可以在 JSON 文件或命令行中指定类别、描述和名称。

```
aws mediaconvert create-preset \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com
    --region region-name-1 \
    --cli-input-json file://~/preset.json
```

如果您通过使用 get-preset 然后修改输出文件的方法来创建预设 JSON 文件，请务必删除以下键-值对：LastUpdated、Arn、Type 和 CreatedAt。

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePreset。

create-queue

以下代码示例演示了如何使用 create-queue。

Amazon CLI

创建自定义队列

以下 create-queue 示例创建一个自定义转码队列。

```
aws mediaconvert create-queue \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1 \
    --name Queue1 \
    --description "Keep this queue empty unless job is urgent."
```

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

输出：

```
{
    "Queue": {
        "Status": "ACTIVE",
        "Name": "Queue1",
        "LastUpdated": 1518034928,
        "Arn": "arn:aws:mediaconvert:region-name-1:012345678998:queues/Queue1",
        "Type": "CUSTOM",
        "CreatedAt": 1518034928,
        "Description": "Keep this queue empty unless job is urgent."
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateQueue。

delete-job-template

以下代码示例演示了如何使用 delete-job-template。

Amazon CLI

删除作业模板

以下 delete-job-template 示例删除指定的自定义作业模板。

```
aws mediaconvert delete-job-template \
```

```
    --name "DASH Streaming" \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

此命令不生成任何输出。运行 aws mediaconvert list-job-templates 来确认您的模板是否被删除。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteJobTemplate。

delete-preset

以下代码示例演示了如何使用 delete-preset。

Amazon CLI

删除自定义按需队列

以下 delete-preset 示例删除指定自定义预设。

```
aws mediaconvert delete-preset \
    --name SimpleMP4 \
    --endpoint-url  https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

此命令不生成任何输出。运行 aws mediaconvert list-presets 来确认您的预设是否被删除。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePreset。

delete-queue

以下代码示例演示了如何使用 delete-queue。

Amazon CLI

删除自定义按需队列

以下 delete-queue 示例删除指定的自定义按需队列。

您无法删除您的默认队列。您无法删除具有活动定价计划或包含未处理作业的预留队列。

```
aws mediaconvert delete-queue \
    --name Customer1 \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

此命令不生成任何输出。运行 aws mediaconvert list-queues 来确认您的队列是否被删除。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteQueue。

describe-endpoints

以下代码示例演示了如何使用 describe-endpoints。

Amazon CLI

获取特定账户端点

以下 describe-endpoints 示例检索向服务发送任何其他请求所需的端点。

```
aws mediaconvert describe-endpoints
```

输出：

```
{
    "Endpoints": [
        {
            "Url": "https://abcd1234.mediaconvert.region-name-1.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert API 参考》中的使用 API 的 MediaConvert 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEndpoints。

get-job-template

以下代码示例演示了如何使用 get-job-template。

Amazon CLI

获取作业模板的详细信息

以下 get-job-template 示例显示指定的自定义作业模板的 JSON 定义。

```
aws mediaconvert get-job-template \
    --name "DASH Streaming" \
    --endpoint-url https://abcd1234.mediaconvert.us-east-1.amazonaws.com
```

输出：

```
{
    "JobTemplate": {
        "StatusUpdateInterval": "SECONDS_60",
        "LastUpdated": 1568652998,
        "Description": "Create a DASH streaming ABR stack",
        "CreatedAt": 1568652998,
        "Priority": 0,
        "Name": "DASH Streaming",
        "Settings": {
            ...<truncatedforbrevity>...
        },
        "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/DASH
 Streaming",
        "Type": "CUSTOM"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJobTemplate。

get-job

以下代码示例演示了如何使用 get-job。

Amazon CLI

获取特定作业的详细信息

以下示例请求 ID 为 1234567890987-1ab2c3 的作业的信息，在本示例中，该作业以错误结尾。

```
aws mediaconvert get-job \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1 \
    --id 1234567890987-1ab2c3
```

要获取账户特定的端点，请使用 describe-endpoints，或发送不带端点的命令。该服务会返回错误和您的端点。

如果您的请求成功，该服务将返回一个含作业信息的 JSON 文件，包括作业设置、任何返回的错误以及其他作业数据，如下所示：

```
{
    "Job": {
        "Status": "ERROR",
        "Queue": "arn:aws:mediaconvert:region-name-1:012345678998:queues/Queue1",
        "Settings": {
            ...<truncated for brevity>...
        },
        "ErrorMessage": "Unable to open input file [s3://my-input-bucket/file-
name.mp4]: [Failed probe/open: [Failed to read data: AssumeRole failed]]",
        "ErrorCode": 1434,
        "Role": "arn:aws:iam::012345678998:role/MediaConvertServiceRole",
        "Arn": "arn:aws:mediaconvert:us-
west-1:012345678998:jobs/1234567890987-1ab2c3",
        "UserMetadata": {},
        "Timing": {
            "FinishTime": 1517442131,
            "SubmitTime": 1517442103,
            "StartTime": 1517442104
        },
        "Id": "1234567890987-1ab2c3",
        "CreatedAt": 1517442103
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetJob。

get-preset

以下代码示例演示了如何使用 get-preset。

Amazon CLI

获取特定预设的详细信息

以下 get-preset 示例请求指定的自定义预设的 JSON 定义。

```
aws mediaconvert get-preset \
    --name SimpleMP4 \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Preset": {
        "Description": "Creates basic MP4 file. No filtering or preproccessing.",
        "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4",
        "LastUpdated": 1568843141,
        "Name": "SimpleMP4",
        "Settings": {
            "ContainerSettings": {
                "Mp4Settings": {
                    "FreeSpaceBox": "EXCLUDE",
                    "CslgAtom": "INCLUDE",
                    "MoovPlacement": "PROGRESSIVE_DOWNLOAD"
                },
                "Container": "MP4"
            },
            "AudioDescriptions": [
                {
                    "LanguageCodeControl": "FOLLOW_INPUT",
                    "AudioTypeControl": "FOLLOW_INPUT",
                    "CodecSettings": {
                        "AacSettings": {
                            "RawFormat": "NONE",
                            "CodecProfile": "LC",
                            "AudioDescriptionBroadcasterMix": "NORMAL",
                            "SampleRate": 48000,
```

```
                        "Bitrate": 96000,
                        "RateControlMode": "CBR",
                        "Specification": "MPEG4",
                        "CodingMode": "CODING_MODE_2_0"
                    },
                    "Codec": "AAC"
                }
            }
        ],
        "VideoDescription": {
            "RespondToAfd": "NONE",
            "TimecodeInsertion": "DISABLED",
            "Sharpness": 50,
            "ColorMetadata": "INSERT",
            "CodecSettings": {
                "H264Settings": {
                    "FramerateControl": "INITIALIZE_FROM_SOURCE",
                    "SpatialAdaptiveQuantization": "ENABLED",
                    "Softness": 0,
                    "Telecine": "NONE",
                    "CodecLevel": "AUTO",
                    "QualityTuningLevel": "SINGLE_PASS",
                    "UnregisteredSeiTimecode": "DISABLED",
                    "Slices": 1,
                    "Syntax": "DEFAULT",
                    "GopClosedCadence": 1,
                    "AdaptiveQuantization": "HIGH",
                    "EntropyEncoding": "CABAC",
                    "InterlaceMode": "PROGRESSIVE",
                    "ParControl": "INITIALIZE_FROM_SOURCE",
                    "NumberBFramesBetweenReferenceFrames": 2,
                    "GopSizeUnits": "FRAMES",
                    "RepeatPps": "DISABLED",
                    "CodecProfile": "MAIN",
                    "FieldEncoding": "PAFF",
                    "GopSize": 90.0,
                    "SlowPal": "DISABLED",
                    "SceneChangeDetect": "ENABLED",
                    "GopBReference": "DISABLED",
                    "RateControlMode": "CBR",
                    "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                    "FlickerAdaptiveQuantization": "DISABLED",
                    "DynamicSubGop": "STATIC",
                    "MinIInterval": 0,
```

```
                        "TemporalAdaptiveQuantization": "ENABLED",
                        "Bitrate": 400000,
                        "NumberReferenceFrames": 3
                    },
                    "Codec": "H_264"
                },
                "AfdSignaling": "NONE",
                "AntiAlias": "ENABLED",
                "ScalingBehavior": "DEFAULT",
                "DropFrameTimecode": "ENABLED"
            }
        },
        "Type": "CUSTOM",
        "CreatedAt": 1568841521
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPreset。

get-queue

以下代码示例演示了如何使用 get-queue。

Amazon CLI

获取队列的详细信息

以下 get-queue 示例检索指定自定义队列的详细信息。

```
aws mediaconvert get-queue \
    --name Customer1 \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Queue": {
        "LastUpdated": 1526428502,
        "Type": "CUSTOM",
        "SubmittedJobsCount": 0,
```

```
        "Status": "ACTIVE",
        "PricingPlan": "ON_DEMAND",
        "CreatedAt": 1526428502,
        "ProgressingJobsCount": 0,
        "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Customer1",
        "Name": "Customer1"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueue。

list-job-templates

以下代码示例演示了如何使用 list-job-templates。

Amazon CLI

示例 1：列出您的自定义作业模板

以下 list-job-templates 示例列出当前区域中的所有自定义作业模板。要列出系统作业模板，请参阅下一个示例。

```
aws mediaconvert list-job-templates \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "JobTemplates": [
        {
            "Description": "Create a DASH streaming ABR stack",
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/DASH
 Streaming",
            "Name": "DASH Streaming",
            "LastUpdated": 1568653007,
            "Priority": 0,
            "Settings": {
                ...<truncatedforbrevity>...
            },
            "Type": "CUSTOM",
```

```
            "StatusUpdateInterval": "SECONDS_60",
            "CreatedAt": 1568653007
        },
        {
            "Description": "Create a high-res file",
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/File",
            "Name": "File",
            "LastUpdated": 1568653007,
            "Priority": 0,
            "Settings": {
                ...<truncatedforbrevity>...
            },
            "Type": "CUSTOM",
            "StatusUpdateInterval": "SECONDS_60",
            "CreatedAt": 1568653023
        }
    ]
}
```

示例 2：列出 MediaConvert 系统作业模板

以下 `list-job-templates` 示例列出所有系统作业模板。

```
aws mediaconvert list-job-templates \
    --endpoint-url https://abcd1234.mediaconvert.us-east-1.amazonaws.com \
    --list-by SYSTEM
```

输出：

```
{
    "JobTemplates": [
        {
            "CreatedAt": 1568321779,
            "Arn": "arn:aws:mediaconvert:us-east-1:123456789012:jobTemplates/System-
Generic_Mp4_Hev1_Avc_Aac_Sdr_Qvbr",
            "Name": "System-Generic_Mp4_Hev1_Avc_Aac_Sdr_Qvbr",
            "Description": "GENERIC, MP4, AVC + HEV1(HEVC,SDR), AAC, SDR, QVBR",
            "Category": "GENERIC",
            "Settings": {
                "AdAvailOffset": 0,
                "OutputGroups": [
                    {
                        "Outputs": [
```

```
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1280x720p_30Hz_5Mbps_Qvbr_Vq9",
                                "NameModifier":
 "_Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1280x720p_30Hz_5000Kbps_Qvbr_Vq9"
                            },
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1920x1080p_30Hz_10Mbps_Qvbr_Vq9",
                                "NameModifier":
 "_Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1920x1080p_30Hz_10000Kbps_Qvbr_Vq9"
                            },
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Sd_Mp4_Avc_Aac_16x9_Sdr_640x360p_30Hz_0.8Mbps_Qvbr_Vq7",
                                "NameModifier":
 "_Generic_Sd_Mp4_Avc_Aac_16x9_Sdr_640x360p_30Hz_800Kbps_Qvbr_Vq7"
                            },
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1280x720p_30Hz_4Mbps_Qvbr_Vq9",
                                "NameModifier":
 "_Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1280x720p_30Hz_4000Kbps_Qvbr_Vq9"
                            },
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1920x1080p_30Hz_8Mbps_Qvbr_Vq9",
                                "NameModifier":
 "_Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1920x1080p_30Hz_8000Kbps_Qvbr_Vq9"
                            },
                            {
                                "Extension": "mp4",
                                "Preset": "System-
Generic_Uhd_Mp4_Hev1_Aac_16x9_Sdr_3840x2160p_30Hz_12Mbps_Qvbr_Vq9",
                                "NameModifier":
 "_Generic_Uhd_Mp4_Hev1_Aac_16x9_Sdr_3840x2160p_30Hz_12000Kbps_Qvbr_Vq9"
                            }
                        ],
                        "OutputGroupSettings": {
```

```
                            "FileGroupSettings": {

                            },
                            "Type": "FILE_GROUP_SETTINGS"
                        },
                        "Name": "File Group"
                    }
                ]
            },
            "Type": "SYSTEM",
            "LastUpdated": 1568321779
        },
        ...<truncatedforbrevity>...
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobTemplates。

`list-jobs`

以下代码示例演示了如何使用 `list-jobs`。

Amazon CLI

获取某个区域中所有作业的详细信息

以下示例请求您在指定区域所有作业的信息。

```
aws mediaconvert list-jobs \
    --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
    --region region-name-1
```

要获取账户特定的端点，请使用 `describe-endpoints`，或发送不带端点的命令。该服务会返回错误和您的端点。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

list-presets

以下代码示例演示了如何使用 list-presets。

Amazon CLI

示例 1：列出您的自定义输出预设

以下 list-presets 示例列出您的自定义输出预设。要列出系统预设，请参阅下一个示例。

```
aws mediaconvert list-presets \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Presets": [
        {
            "Name": "SimpleMP4",
            "CreatedAt": 1568841521,
            "Settings": {
                ......
            },
            "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleMP4",
            "Type": "CUSTOM",
            "LastUpdated": 1568843141,
            "Description": "Creates basic MP4 file. No filtering or preproccessing."
        },
        {
            "Name": "SimpleTS",
            "CreatedAt": 1568843113,
            "Settings": {
                ... truncated for brevity ...
            },
            "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleTS",
            "Type": "CUSTOM",
            "LastUpdated": 1568843113,
            "Description": "Create a basic transport stream."
        }
    ]
}
```

示例 2：列出系统输出预设

以下 `list-presets` 示例列出可用的 MediaConvert 系统预设。要列出您的自定义预设，请参阅前面的示例。

```
aws mediaconvert list-presets \
    --list-by SYSTEM \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Presets": [
        {
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/System-
Avc_16x9_1080p_29_97fps_8500kbps",
            "Name": "System-Avc_16x9_1080p_29_97fps_8500kbps",
            "CreatedAt": 1568321789,
            "Description": "Wifi, 1920x1080, 16:9, 29.97fps, 8500kbps",
            "LastUpdated": 1568321789,
            "Type": "SYSTEM",
            "Category": "HLS",
            "Settings": {
            ...<output settings removed for brevity>...
            }
        },

        ...<list of presets shortened for brevity>...

        {
            "Arn": "arn:aws:mediaconvert:us-east-1:123456789012:presets/System-
Xdcam_HD_1080i_29_97fps_35mpbs",
            "Name": "System-Xdcam_HD_1080i_29_97fps_35mpbs",
            "CreatedAt": 1568321790,
            "Description": "XDCAM MPEG HD, 1920x1080i, 29.97fps, 35mbps",
            "LastUpdated": 1568321790,
            "Type": "SYSTEM",
            "Category": "MXF",
            "Settings": {
            ...<output settings removed for brevity>...
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPresets。

`list-queues`

以下代码示例演示了如何使用 `list-queues`。

Amazon CLI

列出队列

以下 `list-queues` 示例列出您的所有 MediaConvert 队列。

```
aws mediaconvert list-queues \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Queues": [
        {
            "PricingPlan": "ON_DEMAND",
            "Type": "SYSTEM",
            "Status": "ACTIVE",
            "CreatedAt": 1503451595,
            "Name": "Default",
            "SubmittedJobsCount": 0,
            "ProgressingJobsCount": 0,
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Default",
            "LastUpdated": 1534549158
        },
        {
            "PricingPlan": "ON_DEMAND",
            "Type": "CUSTOM",
            "Status": "ACTIVE",
            "CreatedAt": 1537460025,
            "Name": "Customer1",
            "SubmittedJobsCount": 0,
            "Description": "Jobs we run for our cusotmer.",
            "ProgressingJobsCount": 0,
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Customer1",
```

```
            "LastUpdated": 1537460025
        },
        {
            "ProgressingJobsCount": 0,
            "Status": "ACTIVE",
            "Name": "transcode-library",
            "SubmittedJobsCount": 0,
            "LastUpdated": 1564066204,
            "ReservationPlan": {
                "Status": "ACTIVE",
                "ReservedSlots": 1,
                "PurchasedAt": 1564066203,
                "Commitment": "ONE_YEAR",
                "ExpiresAt": 1595688603,
                "RenewalType": "EXPIRE"
            },
            "PricingPlan": "RESERVED",
            "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/transcode-
library",
            "Type": "CUSTOM",
            "CreatedAt": 1564066204
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueues。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出 MediaConvert 队列、作业模板或输出预设上的标签

以下 list-tags-for-resource 示例列出指定输出预设上的标签。

```
aws mediaconvert list-tags-for-resource \
    --arn arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4 \
    --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "ResourceTags": {
        "Tags": {
            "customer": "zippyVideo"
        },
        "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的标记 Amazon Elemental MediaConvert 队列、作业模板和输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

update-job-template

以下代码示例演示了如何使用 update-job-template。

Amazon CLI

更改作业模板

以下 update-job-template 示例将指定自定义作业模板的 JSON 定义替换为提供的文件中的 JSON 定义。

aws mediaconvert update-job-template --name File1 --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com --cli-input-json file://~/job-template-update.json

的内容job-template-update.json：

```
{
    "Description": "A simple job template that generates a single file output.",
    "Queue": "arn:aws:mediaconvert:us-east-1:012345678998:queues/Default",
    "Name": "SimpleFile",
    "Settings": {
      "OutputGroups": [
        {
          "Name": "File Group",
          "Outputs": [
            {
```

```
            "ContainerSettings": {
              "Container": "MP4",
              "Mp4Settings": {
                "CslgAtom": "INCLUDE",
                "FreeSpaceBox": "EXCLUDE",
                "MoovPlacement": "PROGRESSIVE_DOWNLOAD"
              }
            },
            "VideoDescription": {
              "ScalingBehavior": "DEFAULT",
              "TimecodeInsertion": "DISABLED",
              "AntiAlias": "ENABLED",
              "Sharpness": 50,
              "CodecSettings": {
                "Codec": "H_264",
                "H264Settings": {
                  "InterlaceMode": "PROGRESSIVE",
                  "NumberReferenceFrames": 3,
                  "Syntax": "DEFAULT",
                  "Softness": 0,
                  "GopClosedCadence": 1,
                  "GopSize": 90,
                  "Slices": 1,
                  "GopBReference": "DISABLED",
                  "SlowPal": "DISABLED",
                  "SpatialAdaptiveQuantization": "ENABLED",
                  "TemporalAdaptiveQuantization": "ENABLED",
                  "FlickerAdaptiveQuantization": "DISABLED",
                  "EntropyEncoding": "CABAC",
                  "Bitrate": 400000,
                  "FramerateControl": "INITIALIZE_FROM_SOURCE",
                  "RateControlMode": "CBR",
                  "CodecProfile": "MAIN",
                  "Telecine": "NONE",
                  "MinIInterval": 0,
                  "AdaptiveQuantization": "HIGH",
                  "CodecLevel": "AUTO",
                  "FieldEncoding": "PAFF",
                  "SceneChangeDetect": "ENABLED",
                  "QualityTuningLevel": "SINGLE_PASS",
                  "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                  "UnregisteredSeiTimecode": "DISABLED",
                  "GopSizeUnits": "FRAMES",
                  "ParControl": "INITIALIZE_FROM_SOURCE",
```

```
                    "NumberBFramesBetweenReferenceFrames": 2,
                    "RepeatPps": "DISABLED",
                    "DynamicSubGop": "STATIC"
                  }
                },
                "AfdSignaling": "NONE",
                "DropFrameTimecode": "ENABLED",
                "RespondToAfd": "NONE",
                "ColorMetadata": "INSERT"
              },
              "AudioDescriptions": [
                {
                  "AudioTypeControl": "FOLLOW_INPUT",
                  "CodecSettings": {
                    "Codec": "AAC",
                    "AacSettings": {
                      "AudioDescriptionBroadcasterMix": "NORMAL",
                      "Bitrate": 96000,
                      "RateControlMode": "CBR",
                      "CodecProfile": "LC",
                      "CodingMode": "CODING_MODE_2_0",
                      "RawFormat": "NONE",
                      "SampleRate": 48000,
                      "Specification": "MPEG4"
                    }
                  },
                  "LanguageCodeControl": "FOLLOW_INPUT"
                }
              ]
            }
          ],
          "OutputGroupSettings": {
            "Type": "FILE_GROUP_SETTINGS",
            "FileGroupSettings": {}
          }
        }
      ],
      "AdAvailOffset": 0
    },
    "StatusUpdateInterval": "SECONDS_60",
    "Priority": 0
}
```

即使请求导致错误，系统也会返回随您的请求发送的 JSON 有效载荷。因此，返回的 JSON 不一定是作业模板的新定义。

因为 JSON 有效载荷较长，所以您可能需要向上滚动才能看到错误消息。

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 作业模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJobTemplate。

update-preset

以下代码示例演示了如何使用 update-preset。

Amazon CLI

更改预设

以下 update-preset 示例替换了指定预设的描述。

```
aws mediaconvert update-preset \
--name Customer1 \
--description "New description text."
--endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

此命令不生成任何输出。输出：

```
{
    "Preset": {
        "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleMP4",
        "Settings": {
        ...<output settings removed for brevity>...
        },
        "Type": "CUSTOM",
        "LastUpdated": 1568938411,
        "Description": "New description text.",
        "Name": "SimpleMP4",
        "CreatedAt": 1568938240
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 输出预设。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePreset。

update-queue

以下代码示例演示了如何使用 update-queue。

Amazon CLI

更改队列

以下 update-queue 示例通过将指定队列的状态改为 PAUSED 来暂停它。

```
aws mediaconvert update-queue \
--name Customer1 \
--status PAUSED
--endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

输出：

```
{
    "Queue": {
        "LastUpdated": 1568839845,
        "Status": "PAUSED",
        "ProgressingJobsCount": 0,
        "CreatedAt": 1526428516,
        "Arn": "arn:aws:mediaconvert:us-west-1:123456789012:queues/Customer1",
        "Name": "Customer1",
        "SubmittedJobsCount": 0,
        "PricingPlan": "ON_DEMAND",
        "Type": "CUSTOM"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaConvert 用户指南》中的使用 Amazon Elemental MediaConvert 队列。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateQueue。

使用 Amazon CLI 的 MediaLive 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaLive 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-channel`

以下代码示例演示了如何使用 `create-channel`。

Amazon CLI

创建通道

以下 `create-channel` 示例通过传入一个 JSON 文件来创建通道，该 JSON 文件中包含要指定的参数。

此示例中的通道采集一个 HLS PULL 输入，该输入连接到包含视频、音频和嵌入式字幕的源。该通道创建一个以 Akamai 服务器为目标的 HLS 输出组。输出组包含两个输出：一个用于 H.265 视频和 AAC 音频，另一个用于 Web-VTT 字幕，仅提供英语版本。

此通道示例的 JSON 包含一个使用 HLS PULL 输入并生成以 Akamai 为目标的 HLS 输出组的通道的最低要求参数。JSON 包含以下主要部分：

`InputAttachments`，用于为音频指定一个来源，以及为字幕指定一个来源。它不指定视频选择器，这意味着 MediaLive 会提取在来源中找到的第一个视频。`Destinations`，其中包含此通道中单个输出组的两个 IP 地址（URL）。这些地址需要密码。`EncoderSettings`，其中包含分部。`AudioDescriptions`，用于指定通道包含一个音频输出资产，该资产使用来自 InputAttachments 的来源，并产生 AAC 格式的音频。`CaptionDescriptions`，用于指定通道包含一个字幕输出资产，该资产使用来自 InputAttachments 的来源，并以 Web-VTT 格式生成字幕。`VideoDescriptions`，用于指定通道包含一个视频输出资产，该资产具有指定分辨

率。OutputGroups，用于指定输出组。在此示例中，有一个名为 Akamai 的组。使用 HLS PUT 进行连接。输出组包含两个输出。一个输出用于视频资产（名为 Video_high），一个用于音频资产（名为 Audio_EN）。一个输出用于字幕资产（名为 WebVTT_EN）。

在此示例中，某些参数不包含任何值或包含嵌套的空参数。例如，Video_and_audio 输出的 OutputSettings 包含几个以空参数 M3u8Settings 结尾的嵌套参数。虽然必须包含此参数，但您可以省略其一个、多个或所有子项，这意味着子项将采用其默认值或为空。

适用于此通道示例但未在此文件中指定的所有参数要么采用默认值，要么设置为 null，要么采用由 MediaLive 生成的唯一值。

```
aws medialive create-channel \
    --cli-input-json file://channel-in-hls-out-hls-akamai.json
```

的内容channel-in-hls-out-hls-akamai.json：

```
{
    "Name": "News_West",
    "RoleArn": "arn:aws:iam::111122223333:role/MediaLiveAccessRole",
    "InputAttachments": [
        {
            "InputAttachmentName": "local_news",
            "InputId": "1234567",
            "InputSettings": {
                "AudioSelectors": [
                    {
                        "Name": "English-Audio",
                        "SelectorSettings": {
                            "AudioLanguageSelection": {
                                "LanguageCode": "EN"
                            }
                        }
                    }
                ],
                "CaptionSelectors": [
                    {
                        "LanguageCode": "ENE",
                        "Name": "English_embedded"
                    }
                ]
            }
        }
```

```
    ],
    "Destinations": [
        {
            "Id": "akamai-server-west",
            "Settings": [
                {
                    "PasswordParam": "/medialive/examplecorp1",
                    "Url": "http://203.0.113.55/news/news_west",
                    "Username": "examplecorp"
                },
                {
                    "PasswordParam": "/medialive/examplecorp2",
                    "Url": "http://203.0.113.82/news/news_west",
                    "Username": "examplecorp"
                }
            ]
        }
    ],
    "EncoderSettings": {
        "AudioDescriptions": [
            {
                "AudioSelectorName": "English-Audio",
                "CodecSettings": {
                    "AacSettings": {}
                },
                "Name": "Audio_EN"
            }
        ],
        "CaptionDescriptions": [
            {
                "CaptionSelectorName": "English_embedded",
                "DestinationSettings": {
                    "WebvttDestinationSettings": {}
                },
                "Name": "WebVTT_EN"
            }
        ],
        "VideoDescriptions": [
            {
                "Height": 720,
                "Name": "Video_high",
                "Width": 1280
            }
        ],
```

```
        "OutputGroups": [
            {
                "Name": "Akamai",
                "OutputGroupSettings": {
                    "HlsGroupSettings": {
                        "Destination": {
                            "DestinationRefId": "akamai-server-west"
                        },
                        "HlsCdnSettings": {
                            "HlsBasicPutSettings": {}
                        }
                    }
                },
                "Outputs": [
                    {
                        "AudioDescriptionNames": [
                            "Audio_EN"
                        ],
                        "OutputName": "Video_and_audio",
                        "OutputSettings": {
                            "HlsOutputSettings": {
                                "HlsSettings": {
                                    "StandardHlsSettings": {
                                        "M3u8Settings": {}
                                    }
                                },
                                "NameModifier": "_1"
                            }
                        },
                        "VideoDescriptionName": "Video_high"
                    },
                    {
                        "CaptionDescriptionNames": [
                            "WebVTT_EN"
                        ],
                        "OutputName": "Captions-WebVTT",
                        "OutputSettings": {
                            "HlsOutputSettings": {
                                "HlsSettings": {
                                    "StandardHlsSettings": {
                                        "M3u8Settings": {}
                                    }
                                },
                                "NameModifier": "_2"
```

```
                            }
                        }
                    }
                ]
            }
        ],
        "TimecodeConfig": {
            "Source": "EMBEDDED"
        }
    }
}
```

输出：

输出重复 JSON 文件的内容，再加上以下值。所有参数均按字母顺序排序。

通道的 ARN。ARN 的最后一部分是唯一的通道 ID。EgressEndpoints 在此通道示例中为空，因为它仅用于 PUSH 输入。在应用时，它会显示内容被推送到的 MediaLive 上的地址。OutputGroups、Outputs。它们显示输出组和输出的所有参数，包括您未包含但与此通道相关的参数。参数可能为空（可能表示此通道配置中禁用该参数或功能），也可能显示适用的默认值。LogLevel 设置为默认值（DISABLED）。Tags 设置为默认值（null）。PipelinesRunningCount 和 State 显示通道的当前状态。

有关更多信息，请参阅《Amazon Elemental MediaLive 用户指南》中的从头开始创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChannel。

create-input

以下代码示例演示了如何使用 create-input。

Amazon CLI

创建输入

以下 create-input 示例通过传入一个 JSON 文件来创建 HLS PULL 输入，该文件中包含适用于此类输入的参数。此输入示例的 JSON 为输入指定了两个来源（地址），以支持采集中的冗余。这些地址需要密码。

```
aws medialive create-input \
    --cli-input-json file://input-hls-pull-news.json
```

的内容input-hls-pull-news.json：

```
{
    "Name": "local_news",
    "RequestId": "cli000059",
    "Sources": [
            {
                    "Url": "https://203.0.113.13/newschannel/anytownusa.m3u8",
                    "Username": "examplecorp",
                    "PasswordParam": "/medialive/examplecorp1"
            },
       {
                    "Url": "https://198.51.100.54/fillervideos/oceanwaves.mp4",
                    "Username": "examplecorp",
                    "PasswordParam": "examplecorp2"
            }
    ],
    "Type": "URL_PULL"
}
```

输出：

输出重复 JSON 文件的内容，再加上以下值。所有参数均按字母顺序排序。

输入的 Arn。ARN 的最后一部分是唯一的输入 ID。Attached Channels，对于新创建的输入始终为空。Destinations，在本示例中为空，因为它仅用于 PUSH 输入。输入的 Id，与 ARN 中的 ID 一样。MediaConnectFlows，在本示例中为空，因为它仅用于 MediaConnect 类型的输入。SecurityGroups，在本示例中为空，因为它仅用于 PUSH 输入。此输入的 State。Tags，为空（此参数的默认值）。

有关更多信息，请参阅《Amazon Elemental MediaLive 用户指南》中的创建输入。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInput。

使用 Amazon CLI 的 MediaPackage 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaPackage 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-channel`

以下代码示例演示了如何使用 `create-channel`。

Amazon CLI

创建通道

以下 `create-channel` 命令在当前账户中创建一个名为 `sportschannel` 的通道。

```
aws mediapackage create-channel --id sportschannel
```

输出：

```
{
   "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0",
   "HlsIngest": {
        "IngestEndpoints": [
            {
                "Id": "6d345804ec3f46c9b454a91d4a80d0e0",
                "Password": "generatedwebdavpassword1",
                "Url": "https://f31c86aed53b815a.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/6d345804ec3f46c9b454a91d4a80d0e0/channel",
                "Username": "generatedwebdavusername1"
            },
            {
                "Id": "2daa32878af24803b24183727211b8ff",
                "Password": "generatedwebdavpassword2",
                "Url": "https://6ebbe7e04c4b0afa.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/2daa32878af24803b24183727211b8ff/channel",
                "Username": "generatedwebdavusername2"
```

```
            }
        ]
    },
    "Id": "sportschannel",
    "Tags": {
        "region": "west"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的创建通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateChannel。

create-origin-endpoint

以下代码示例演示了如何使用 create-origin-endpoint。

Amazon CLI

创建源端点

以下 create-origin-endpoint 命令使用 JSON 文件中提供的包设置和指定的端点设置创建一个名为 cmafsports 的源端点。

```
aws mediapackage create-origin-endpoint \
    --channel-id sportschannel \
    --id cmafsports \
    --cmaf-package file://file/path/cmafpkg.json --description "cmaf output of
 sports" \
    --id cmaf_sports \
    --manifest-name sports_channel \
    --startover-window-seconds 300 \
    --tags region=west,media=sports \
    --time-delay-seconds 10
```

输出：

```
{
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",
    "ChannelId": "sportschannel",
    "CmafPackage": {
```

```
        "HlsManifests": [
            {
                "AdMarkers": "PASSTHROUGH",
                "Id": "cmaf_sports_endpoint",
                "IncludeIframeOnlyStream": true,
                "ManifestName": "index",
                "PlaylistType": "EVENT",
                "PlaylistWindowSeconds": 300,
                "ProgramDateTimeIntervalSeconds": 300,
                "Url": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmaf_sports_endpoint/
index.m3u8"
            }
        ],
        "SegmentDurationSeconds": 2,
        "SegmentPrefix": "sportschannel"
    },
    "Description": "cmaf output of sports",
    "Id": "cmaf_sports",
    "ManifestName": "sports_channel",
    "StartoverWindowSeconds": 300,
    "Tags": {
        "region": "west",
        "media": "sports"
    },
    "TimeDelaySeconds": 10,
    "Url": "",
    "Whitelist": []
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的创建端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOriginEndpoint。

delete-channel

以下代码示例演示了如何使用 delete-channel。

Amazon CLI

删除通道

以下 delete-channel 命令删除名为 test 的通道。

```
aws mediapackage delete-channel \
    --id test
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的删除通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteChannel。

delete-origin-endpoint

以下代码示例演示了如何使用 delete-origin-endpoint。

Amazon CLI

删除源端点

以下 delete-origin-endpoint 命令删除名为 tester2 的源端点。

```
aws mediapackage delete-origin-endpoint \
    --id tester2
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的删除端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOriginEndpoint。

describe-channel

以下代码示例演示了如何使用 describe-channel。

Amazon CLI

描述通道

以下 describe-channel 命令显示名为 test 的通道的所有详细信息。

```
aws mediapackage describe-channel \
    --id test
```

输出：

```
{
```

```
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/584797f1740548c389a273585dd22a63",
    "HlsIngest": {
        "IngestEndpoints": [
            {
                "Id": "584797f1740548c389a273585dd22a63",
                "Password": "webdavgeneratedpassword1",
                "Url": "https://9be9c4405c474882.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
                "Username": "webdavgeneratedusername1"
            },
            {
                "Id": "7d187c8616fd455f88aaa5a9fcf74442",
                "Password": "webdavgeneratedpassword2",
                "Url": "https://7bf454c57220328d.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",
                "Username": "webdavgeneratedusername2"
            }
        ]
    },
    "Id": "test",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的“查看通道详细信息”<https://docs.aws.amazon.com/mediapackage/latest/ug/channels-view.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeChannel。

`describe-origin-endpoint`

以下代码示例演示了如何使用 `describe-origin-endpoint`。

Amazon CLI

描述源端点

以下 `describe-origin-endpoint` 命令显示名为 `cmaf_sports` 的源端点的所有详细信息。

```
aws mediapackage describe-origin-endpoint \
    --id cmaf_sports
```

输出：

```
{
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",
    "ChannelId": "sportschannel",
    "CmafPackage": {
        "HlsManifests": [
            {
                "AdMarkers": "NONE",
                "Id": "cmaf_sports_endpoint",
                "IncludeIframeOnlyStream": false,
                "PlaylistType": "EVENT",
                "PlaylistWindowSeconds": 60,
                "ProgramDateTimeIntervalSeconds": 0,
                "Url": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmaf_sports_endpoint/
index.m3u8"
            }
        ],
        "SegmentDurationSeconds": 2,
        "SegmentPrefix": "sportschannel"
    },
    "Id": "cmaf_sports",
    "ManifestName": "index",
    "StartoverWindowSeconds": 0,
    "Tags": {
        "region": "west",
        "media": "sports"
    },
    "TimeDelaySeconds": 0,
    "Url": "",
    "Whitelist": []
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看单个端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOriginEndpoint。

`list-channels`

以下代码示例演示了如何使用 `list-channels`。

Amazon CLI

列出所有通道

以下 `list-channels` 命令列出了当前 Amazon 账户中配置的所有通道。

```
aws mediapackage list-channels
```

输出：

```
{
    "Channels": [
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/584797f1740548c389a273585dd22a63",
            "HlsIngest": {
                "IngestEndpoints": [
                    {
                        "Id": "584797f1740548c389a273585dd22a63",
                        "Password": "webdavgeneratedpassword1",
                        "Url": "https://9be9c4405c474882.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
                        "Username": "webdavgeneratedusername1"
                    },
                    {
                        "Id": "7d187c8616fd455f88aaa5a9fcf74442",
                        "Password": "webdavgeneratedpassword2",
                        "Url": "https://7bf454c57220328d.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",
                        "Username": "webdavgeneratedusername2"
                    }
                ]
            },
            "Id": "test",
            "Tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看通道详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChannels。

list-origin-endpoints

以下代码示例演示了如何使用 list-origin-endpoints。

Amazon CLI

列出通道上的所有源端点

以下 list-origin-endpoints 命令列出了名为 test 的通道中配置的所有源端点。

```
aws mediapackage list-origin-endpoints \
    --channel-id test
```

输出：

```
{
    "OriginEndpoints": [
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/247cff871f2845d3805129be22f2c0a2",
            "ChannelId": "test",
            "DashPackage": {
                "ManifestLayout": "FULL",
                "ManifestWindowSeconds": 60,
                "MinBufferTimeSeconds": 30,
                "MinUpdatePeriodSeconds": 15,
                "PeriodTriggers": [],
                "Profile": "NONE",
                "SegmentDurationSeconds": 2,
                "SegmentTemplateFormat": "NUMBER_WITH_TIMELINE",
                "StreamSelection": {
                    "MaxVideoBitsPerSecond": 2147483647,
                    "MinVideoBitsPerSecond": 0,
                    "StreamOrder": "ORIGINAL"
                },
                "SuggestedPresentationDelaySeconds": 25
            },
            "Id": "tester2",
            "ManifestName": "index",
            "StartoverWindowSeconds": 0,
            "Tags": {},
```

```
            "TimeDelaySeconds": 0,
            "Url": "https://8343f7014c0ea438.mediapackage.us-west-2.amazonaws.com/
out/v1/247cff871f2845d3805129be22f2c0a2/index.mpd",
            "Whitelist": []
        },
        {
            "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/869e237f851549e9bcf10e3bc2830839",
            "ChannelId": "test",
            "HlsPackage": {
                "AdMarkers": "NONE",
                "IncludeIframeOnlyStream": false,
                "PlaylistType": "EVENT",
                "PlaylistWindowSeconds": 60,
                "ProgramDateTimeIntervalSeconds": 0,
                "SegmentDurationSeconds": 6,
                "StreamSelection": {
                    "MaxVideoBitsPerSecond": 2147483647,
                    "MinVideoBitsPerSecond": 0,
                    "StreamOrder": "ORIGINAL"
                },
                "UseAudioRenditionGroup": false
            },
            "Id": "tester",
            "ManifestName": "index",
            "StartoverWindowSeconds": 0,
            "Tags": {},
            "TimeDelaySeconds": 0,
            "Url": "https://8343f7014c0ea438.mediapackage.us-west-2.amazonaws.com/
out/v1/869e237f851549e9bcf10e3bc2830839/index.m3u8",
            "Whitelist": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看与通道关联的所有端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOriginEndpoints。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出分配给资源的标签

以下 `list-tags-for-resource` 命令列出分配给指定资源的标签。

```
aws mediapackage list-tags-for-resource \
    --resource-arn arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0
```

输出：

```
{
    "Tags": {
        "region": "west"
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的在 Amazon Elemental MediaPackage 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

rotate-ingest-endpoint-credentials

以下代码示例演示了如何使用 `rotate-ingest-endpoint-credentials`。

Amazon CLI

轮换采集凭证

以下 `rotate-ingest-endpoint-credentials` 命令轮换指定采集端点的 WebDAV 用户名和密码。

```
aws mediapackage rotate-ingest-endpoint-credentials \
    --id test \
    --ingest-endpoint-id 584797f1740548c389a273585dd22a63
```

输出：

```
{
```

```
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/584797f1740548c389a273585dd22a63",
    "HlsIngest": {
        "IngestEndpoints": [
            {
                "Id": "584797f1740548c389a273585dd22a63",
                "Password": "webdavregeneratedpassword1",
                "Url": "https://9be9c4405c474882.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
                "Username": "webdavregeneratedusername1"
            },
            {
                "Id": "7d187c8616fd455f88aaa5a9fcf74442",
                "Password": "webdavgeneratedpassword2",
                "Url": "https://7bf454c57220328d.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",
                "Username": "webdavgeneratedusername2"
            }
        ]
    },
    "Id": "test",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的轮换凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RotateIngestEndpointCredentials。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签添加到资源中

以下 tag-resource 命令将 region=west 键值对添加到指定资源。

```
aws mediapackage tag-resource \
    --resource-arn arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0 \
```

```
    --tags region=west
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的在 Amazon Elemental MediaPackage 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 命令从指定通道中移除具有键 region 的标签。

```
aws mediapackage untag-resource \
    --resource-arn arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0 \
    --tag-keys region
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的在 Amazon Elemental MediaPackage 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-channel

以下代码示例演示了如何使用 update-channel。

Amazon CLI

更新通道

以下 update-channel 命令更新名为 sportschannel 的通道以包含描述 24x7 sports。

```
aws mediapackage update-channel \
    --id sportschannel \
```

```
    --description "24x7 sports"
```

输出：

```
{
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0",
    "Description": "24x7 sports",
    "HlsIngest": {
        "IngestEndpoints": [
            {
                "Id": "6d345804ec3f46c9b454a91d4a80d0e0",
                "Password": "generatedwebdavpassword1",
                "Url": "https://f31c86aed53b815a.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/6d345804ec3f46c9b454a91d4a80d0e0/channel",
                "Username": "generatedwebdavusername1"
            },
            {
                "Id": "2daa32878af24803b24183727211b8ff",
                "Password": "generatedwebdavpassword2",
                "Url": "https://6ebbe7e04c4b0afa.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/2daa32878af24803b24183727211b8ff/channel",
                "Username": "generatedwebdavusername2"
            }
        ]
    },
    "Id": "sportschannel",
    "Tags": {}
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的编辑通道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateChannel。

update-origin-endpoint

以下代码示例演示了如何使用 update-origin-endpoint。

Amazon CLI

更新源端点

以下 update-origin-endpoint 命令更新名为 cmaf_sports 的源端点。它将延迟时间改为 0 秒。

```
aws mediapackage update-origin-endpoint \
    --id cmaf_sports \
    --time-delay-seconds 0
```

输出：

```
{
    "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",
    "ChannelId": "sportschannel",
    "CmafPackage": {
        "HlsManifests": [
            {
                "AdMarkers": "NONE",
                "Id": "cmaf_sports_endpoint",
                "IncludeIframeOnlyStream": false,
                "PlaylistType": "EVENT",
                "PlaylistWindowSeconds": 60,
                "ProgramDateTimeIntervalSeconds": 0,
                "Url": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmaf_sports_endpoint/
index.m3u8"
            }
        ],
        "SegmentDurationSeconds": 2,
        "SegmentPrefix": "sportschannel"
    },
    "Id": "cmaf_sports",
    "ManifestName": "index",
    "StartoverWindowSeconds": 0,
    "Tags": {
        "region": "west",
        "media": "sports"
    },
    "TimeDelaySeconds": 0,
    "Url": "",
    "Whitelist": []
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的编辑端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOriginEndpoint。

使用 Amazon CLI 的 MediaPackage VOD 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaPackage VOD 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-asset`

以下代码示例演示了如何使用 `create-asset`。

Amazon CLI

创建资产

以下 `create-asset` 示例在当前 Amazon 账户中创建一个名为 `Chicken_Asset` 的资产。该资产将文件 `30sec_chicken.smil` 采集到 MediaPackage。

```
aws mediapackage-vod create-asset \
    --id chicken_asset \
    --packaging-group-id hls_chicken_gp \
    --source-role-arn arn:aws:iam::111122223333:role/EMP_Vod \
    --source-arn arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil
```

输出：

```
{
    "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:assets/chicken_asset",
    "Id":"chicken_asset",
    "PackagingGroupId":"hls_chicken_gp",
```

```
    "SourceArn":"arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",
    "SourceRoleArn":"arn:aws:iam::111122223333:role/EMP_Vod",
    "EgressEndpoints":[
        {
            "PackagingConfigurationId":"New_config_1",
            "Url":"https://c75ea2668ab49d02bca7ae10ef31c59e.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/
v1/6644b55df1744261ab3732a8e5cdaf07/904b06a58c7645e08d57d40d064216ac/
f5b2e633ff4942228095d164c10074f3/index.m3u8"
        },
        {
            "PackagingConfigurationId":"new_hls",
            "Url":" https://c75ea2668ab49d02bca7ae10ef31c59e.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/v1/6644b55df1744261ab3732a8e5cdaf07/
fe8f1f00a80e424cb4f8da4095835e9e/7370ec57432343af816332356d2bd5c6/string.m3u8"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的采集资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAsset。

create-packaging-configuration

以下代码示例演示了如何使用 create-packaging-configuration。

Amazon CLI

创建打包配置

以下 create-packaging-configuration 示例在名为 hls_chicken 的打包组中创建一个名为 new_hls 的打包配置。此示例使用磁盘上名为 hls_pc.json 的文件来提供详细信息。

```
aws mediapackage-vod create-packaging-configuration \
    --id new_hls \
    --packaging-group-id hls_chicken \
    --hls-package file://hls_pc.json
```

的内容hls_pc.json：

```
{
    "HlsManifests":[
```

```
        {
            "AdMarkers":"NONE",
            "IncludeIframeOnlyStream":false,
            "ManifestName":"string",
            "ProgramDateTimeIntervalSeconds":60,
            "RepeatExtXKey":true,
            "StreamSelection":{
                "MaxVideoBitsPerSecond":1000,
                "MinVideoBitsPerSecond":0,
                "StreamOrder":"ORIGINAL"
            }
        }
    ],
    "SegmentDurationSeconds":6,
    "UseAudioRenditionGroup":false
}
```

输出：

```
{
    "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-configurations/
new_hls",
    "Id":"new_hls",
    "PackagingGroupId":"hls_chicken",
    "HlsManifests":{
        "SegmentDurationSeconds":6,
        "UseAudioRenditionGroup":false,
        "HlsMarkers":[
            {
                "AdMarkers":"NONE",
                "IncludeIframeOnlyStream":false,
                "ManifestName":"string",
                "ProgramDateTimeIntervalSeconds":60,
                "RepeatExtXKey":true,
                "StreamSelection":{
                    "MaxVideoBitsPerSecond":1000,
                    "MinVideoBitsPerSecond":0,
                    "StreamOrder":"ORIGINAL"
                }
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的创建打包配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePackagingConfiguration。

create-packaging-group

以下代码示例演示了如何使用 create-packaging-group。

Amazon CLI

创建打包组

以下 create-packaging-group 示例列出当前 Amazon 账户中配置的所有打包组。

```
aws mediapackage-vod create-packaging-group \
    --id hls_chicken
```

输出：

```
{
    "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-groups/
hls_chicken",
    "Id": "hls_chicken"
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的创建打包组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePackagingGroup。

delete-asset

以下代码示例演示了如何使用 delete-asset。

Amazon CLI

删除资产

以下 delete-asset 示例删除名为 30sec_chicken 的资产。

```
aws mediapackage-vod delete-asset \
    --id 30sec_chicken
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的删除资产。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAsset。

delete-packaging-configuration

以下代码示例演示了如何使用 delete-packaging-configuration。

Amazon CLI

删除打包配置

以下 delete-packaging-configuration 示例删除名为 CMAF 的打包配置。

```
aws mediapackage-vod delete-packaging-configuration \
    --id CMAF
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的删除打包配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePackagingConfiguration。

delete-packaging-group

以下代码示例演示了如何使用 delete-packaging-group。

Amazon CLI

删除打包组

以下 delete-packaging-group 示例删除名为 Dash_widevine 的打包组。

```
aws mediapackage-vod delete-packaging-group \
    --id Dash_widevine
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的删除打包组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePackagingGroup。

describe-asset

以下代码示例演示了如何使用 describe-asset。

Amazon CLI

描述资产

以下 describe-asset 示例显示名为 30sec_chicken 的资产的所有详细信息。

```
aws mediapackage-vod describe-asset \
    --id 30sec_chicken
```

输出：

```
{
    "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:assets/30sec_chicken",
    "Id":"30sec_chicken",
    "PackagingGroupId":"Packaging_group_1",
    "SourceArn":"arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",
    "SourceRoleArn":"arn:aws:iam::111122223333:role/EMP_Vod",
    "EgressEndpoints":[
        {
            "PackagingConfigurationId":"DASH",
            "Url":"https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/v1/
aad7962c569946119c2d5a691be5663c/66c25aff456d463aae0855172b3beb27/4ddfda6da17c4c279a1b8401cb
index.mpd"
        },
        {
            "PackagingConfigurationId":"HLS",
            "Url":"https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/v1/
aad7962c569946119c2d5a691be5663c/6e5bf286a3414254a2bf0d22ae148d7e/06b5875b4d004c3cbdc4da2dc4
index.m3u8"
        },
        {
            "PackagingConfigurationId":"CMAF",
            "Url":"https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/v1/
aad7962c569946119c2d5a691be5663c/628fb5d8d89e4702958b020af27fde0e/05eb062214064238ad6330a443
index.m3u8"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看资产详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAsset。

describe-packaging-configuration

以下代码示例演示了如何使用 describe-packaging-configuration。

Amazon CLI

描述打包配置

以下 describe-packaging-configuration 示例显示名为 DASH 的打包配置的所有详细信息。

```
aws mediapackage-vod describe-packaging-configuration \
    --id DASH
```

输出：

```
{
    "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-configurations/
DASH",
    "Id":"DASH",
    "PackagingGroupId":"Packaging_group_1",
    "DashPackage":[
        {
            "SegmentDurationSeconds":"2"
        },
        {
            "DashManifests":{
                "ManifestName":"index",
                "MinBufferTimeSeconds":"30",
                "Profile":"NONE"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看打包配置详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePackagingConfiguration。

describe-packaging-group

以下代码示例演示了如何使用 describe-packaging-group。

Amazon CLI

描述打包组

以下 describe-packaging-group 示例显示名为 Packaging_group_1 的打包组的所有详细信息。

```
aws mediapackage-vod describe-packaging-group \
    --id Packaging_group_1
```

输出：

```
{
    "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-groups/
Packaging_group_1",
    "Id": "Packaging_group_1"
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看打包组详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePackagingGroup。

list-assets

以下代码示例演示了如何使用 list-assets。

Amazon CLI

列出所有资产

以下 list-assets 示例列出当前 Amazon 账户中配置的所有资产。

```
aws mediapackage-vod list-assets
```

输出：

```
{
    "Assets": [
        "Arn": "arn:aws:mediapackage-vod:us-
west-2:111122223333:assets/30sec_chicken",
        "Id": "30sec_chicken",
        "PackagingGroupId": "Packaging_group_1",
        "SourceArn": "arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",
        "SourceRoleArn": "arn:aws:iam::111122223333:role/EMP_Vod"
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看资产详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAssets。

list-packaging-configurations

以下代码示例演示了如何使用 list-packaging-configurations。

Amazon CLI

列出所有打包配置

以下 list-packaging-configurations 示例列出名为 Packaging_group_1 的打包组上配置的所有打包配置。

```
aws mediapackage-vod list-packaging-configurations \
    --packaging-group-id Packaging_group_1
```

输出：

```
{
    "PackagingConfigurations":[
        {
            "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/CMAF",
            "Id":"CMAF",
            "PackagingGroupId":"Packaging_group_1",
```

```
            "CmafPackage":[
                {
                    "SegmentDurationSeconds":"2"
                },
                {
                    "HlsManifests":{
                        "AdMarkers":"NONE",
                        "RepeatExtXKey":"False",
                        "ManifestName":"index",
                        "ProgramDateTimeIntervalSeconds":"0",
                        "IncludeIframeOnlyStream":"False"
                    }
                }
            ]
        },
        {
            "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/DASH",
            "Id":"DASH",
            "PackagingGroupId":"Packaging_group_1",
            "DashPackage":[
                {
                    "SegmentDurationSeconds":"2"
                },
                {
                    "DashManifests":{
                        "ManifestName":"index",
                        "MinBufferTimeSeconds":"30",
                        "Profile":"NONE"
                    }
                }
            ]
        },
        {
            "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/HLS",
            "Id":"HLS",
            "PackagingGroupId":"Packaging_group_1",
            "HlsPackage":[
                {
                    "SegmentDurationSeconds":"6",
                    "UseAudioRenditionGroup":"False"
                },
                {
```

```
                    "HlsManifests":{
                        "AdMarkers":"NONE",
                        "RepeatExtXKey":"False",
                        "ManifestName":"index",
                        "ProgramDateTimeIntervalSeconds":"0",
                        "IncludeIframeOnlyStream":"False"
                    }
                }
            ]
        },
        {
            "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/New_config_0_copy",
            "Id":"New_config_0_copy",
            "PackagingGroupId":"Packaging_group_1",
            "HlsPackage":[
                {
                    "SegmentDurationSeconds":"6",
                    "UseAudioRenditionGroup":"False"
                },
                {
                    "Encryption":{
                        "EncryptionMethod":"AWS_128",
                        "SpekeKeyProvider":{
                           "RoleArn":"arn:aws:iam:111122223333::role/SPEKERole",
                            "Url":"https://lfgubdvs97.execute-api.us-
west-2.amazonaws.com/EkeStage/copyProtection/",
                            "SystemIds":[
                                "81376844-f976-481e-a84e-cc25d39b0b33"
                            ]
                        }
                    }
                },
                {
                    "HlsManifests":{
                        "AdMarkers":"NONE",
                        "RepeatExtXKey":"False",
                        "ManifestName":"index",
                        "ProgramDateTimeIntervalSeconds":"0",
                        "IncludeIframeOnlyStream":"False"
                    }
                }
            ]
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看打包配置详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackagingConfigurations。

`list-packaging-groups`

以下代码示例演示了如何使用 `list-packaging-groups`。

Amazon CLI

列出所有打包组

以下 `list-packaging-groups` 示例列出当前 Amazon 账户中配置的所有打包组。

```
aws mediapackage-vod list-packaging-groups
```

输出：

```
{
    "PackagingGroups": [
        {
            "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Dash_widevine",
            "Id": "Dash_widevine"
        },
        {
            "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Encrypted_HLS",
            "Id": "Encrypted_HLS"
        },
        {
            "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Packaging_group_1",
            "Id": "Packaging_group_1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaPackage 用户指南》中的查看打包组详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPackagingGroups。

使用 Amazon CLI 的 MediaStore Data Plane 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaStore Data Plane 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-object`

以下代码示例演示了如何使用 `delete-object`。

Amazon CLI

删除对象

以下 `delete-object` 示例删除指定对象。

```
aws mediastore-data delete-object \
    --endpoint=https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path=/folder_name/README.md
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的删除对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObject。

describe-object

以下代码示例演示了如何使用 describe-object。

Amazon CLI

查看对象的标题

以下 describe-object 示例显示指定路径上某个对象的标题。

```
aws mediastore-data describe-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path events/baseball/setup.jpg
```

输出：

```
{
    "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT",
    "ContentType": "image/jpeg",
    "ContentLength": "3860266",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看对象的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeObject。

get-object

以下代码示例演示了如何使用 get-object。

Amazon CLI

示例 1：下载整个对象

以下 get-object 示例下载指定对象。

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path events/baseball/setup.jpg setup.jpg
```

输出：

```
{
    "ContentType": "image/jpeg",
    "StatusCode": 200,
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "ContentLength": "3860266",
    "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT"
}
```

示例 2：下载对象的一部分

以下 get-object 示例下载一个对象的指定部分。

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path events/baseball/setup.jpg setup.jpg \
    --range "bytes=0-100"
```

输出：

```
{
    "StatusCode": 206,
    "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT",
    "ContentType": "image/jpeg",
    "ContentRange": "bytes 0-100/3860266",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "ContentLength": "101"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的下载对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObject。

list-items

以下代码示例演示了如何使用 list-items。

Amazon CLI

示例 1：查看存储在容器中的项目（对象和文件夹）列表

以下 list-items 示例显示存储在指定容器中的项目（对象和文件夹）的列表。

```
aws mediastore-data list-items \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com
```

输出：

```
{
    "Items": [
        {
            "Type": "OBJECT",
            "ContentLength": 3784,
            "Name": "setup.jpg",
            "ETag":
 "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
            "ContentType": "image/jpeg",
            "LastModified": 1563571859.379
        },
        {
            "Type": "FOLDER",
            "Name": "events"
        }
    ]
}
```

示例 2：查看存储在文件夹中的项目（对象和文件夹）列表

以下 `list-items` 示例显示存储在指定文件夹中的项目（对象和文件夹）的列表。

```
aws mediastore-data list-items \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path events/baseball
```

输出：

```
{
    "Items": [
        {
            "ETag":
 "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
            "ContentType": "image/jpeg",
            "Type": "OBJECT",
            "ContentLength": 3860266,
            "LastModified": 1563573031.872,
```

```
            "Name": "setup.jpg"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的查看对象列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListItems。

put-object

以下代码示例演示了如何使用 put-object。

Amazon CLI

示例 1：将对象上传到容器

以下 put-object 示例将对象上传到指定容器。

```
aws mediastore-data put-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --body ReadMe.md \
    --path ReadMe.md \
    --cache-control "max-age=6, public" \
    --content-type binary/octet-stream
```

输出：

```
{
    "ContentSHA256":
 "f29bc64a9d3732b4b9035125fdb3285f5b6455778edca72414671e0ca3b2e0de",
    "StorageClass": "TEMPORAL",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

示例 2：将对象上传到容器内的文件夹

以下 put-object 示例将对象上传到容器内的指定文件夹。

```
aws mediastore-data put-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --body ReadMe.md \
    --path /september-events/ReadMe.md \
```

```
    --cache-control "max-age=6, public" \
    --content-type binary/octet-stream
```

输出：

```
{
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "ContentSHA256":
 "f29bc64a9d3732b4b9035125fdb3285f5b6455778edca72414671e0ca3b2e0de",
    "StorageClass": "TEMPORAL"
}
```

有关更多信息，请参阅《Amazon Elemental MediaStore 用户指南》中的上传对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObject。

使用 Amazon CLI 的 MediaTailor 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MediaTailor 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-playback-configuration`

以下代码示例演示了如何使用 `delete-playback-configuration`。

Amazon CLI

删除配置

以下 `delete-playback-configuration` 示例删除名为 `campaign_short` 的配置。

```
aws mediatailor delete-playback-configuration \
    --name campaign_short
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Elemental MediaTailor 用户指南》中的删除配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePlaybackConfiguration。

get-playback-configuration

以下代码示例演示了如何使用 get-playback-configuration。

Amazon CLI

描述配置

以下 get-playback-configuration 命令显示名为 west_campaign 的配置的所有详细信息。

```
aws mediatailor get-playback-configuration \
    --name west_campaign
```

输出：

```
{
     "AdDecisionServerUrl": "http://your.ads.url",
     "CdnConfiguration": {},
     "DashConfiguration": {
         "ManifestEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
         "MpdLocation": "EMT_DEFAULT",
         "OriginManifestType": "MULTI_PERIOD"
     },
     "HlsConfiguration": {
         "ManifestEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/"
     },
     "Name": "west_campaign",
     "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/west_campaign",
```

```
    "PlaybackEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com",
    "SessionInitializationEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
    "Tags": {},
    "VideoContentSourceUrl": "https://8343f7014c0ea438.mediapackage.us-
west-2.amazonaws.com/out/v1/683f0f2ff7cd43a48902e6dcd5e16dcf/index.m3u8"
}
```

有关更多信息，请参阅《Amazon Elemental MediaTailor 用户指南》中的查看配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPlaybackConfiguration。

list-playback-configurations

以下代码示例演示了如何使用 list-playback-configurations。

Amazon CLI

列出所有配置

以下 list-playback-configurations 命令显示当前 Amazon 账户上的配置的所有详细信息。

```
aws mediatailor list-playback-configurations
```

输出：

```
{
    "Items": [
        {
            "AdDecisionServerUrl": "http://your.ads.url",
            "CdnConfiguration": {},
            "DashConfiguration": {
                "ManifestEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
                "MpdLocation": "EMT_DEFAULT",
                "OriginManifestType": "MULTI_PERIOD"
            },
            "HlsConfiguration": {
```

```
                "ManifestEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/"
            },
            "Name": "west_campaign",
            "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/west_campaign",
            "PlaybackEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com",
            "SessionInitializationEndpointPrefix":
 "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
            "Tags": {},
            "VideoContentSourceUrl": "https://8343f7014c0ea438.mediapackage.us-
west-2.amazonaws.com/out/v1/683f0f2ff7cd43a48902e6dcd5e16dcf/index.m3u8"
        },
        {
            "AdDecisionServerUrl": "http://your.ads.url",
            "CdnConfiguration": {},
            "DashConfiguration": {
                "ManifestEndpointPrefix":
 "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/",
                "MpdLocation": "DISABLED",
                "OriginManifestType": "MULTI_PERIOD"
            },
            "HlsConfiguration": {
                "ManifestEndpointPrefix":
 "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/"
            },
            "Name": "sports_campaign",
            "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/sports_campaign",
            "PlaybackEndpointPrefix":
 "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com",
            "SessionInitializationEndpointPrefix":
 "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/",
            "SlateAdUrl": "http://s3.bucket/slate_ad.mp4",
            "Tags": {},
            "VideoContentSourceUrl": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/sports_endpoint/
index.m3u8"
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elemental MediaTailor 用户指南》中的“查看配置”<https://docs.aws.amazon.com/mediatailor/latest/ug/configurations-view.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPlaybackConfigurations。

put-playback-configuration

以下代码示例演示了如何使用 put-playback-configuration。

Amazon CLI

创建配置

以下 put-playback-configuration 命令创建一个名为 campaign_short 的配置。

```
aws mediatailor put-playback-configuration \
    --name campaign_short \
    --ad-decision-server-url http://your.ads.url \
    --video-content-source-url http://video.bucket/index.m3u8
```

输出：

```
{
    "AdDecisionServerUrl": "http://your.ads.url",
    "CdnConfiguration": {},
    "DashConfiguration": {
         "ManifestEndpointPrefix":
 "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/",
         "MpdLocation": "EMT_DEFAULT",
         "OriginManifestType": "MULTI_PERIOD"
    },
    "HlsConfiguration": {
        "ManifestEndpointPrefix":
 "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/"
    },
    "Name": "campaign_short",
```

```
    "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/campaign_short",
    "PlaybackEndpointPrefix":
 "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com",
    "SessionInitializationEndpointPrefix":
 "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/",
    "Tags": {},
    "VideoContentSourceUrl": "http://video.bucket/index.m3u8"
}
```

有关更多信息，请参阅《Amazon Elemental MediaTailor 用户指南》中的创建配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPlaybackConfiguration。

使用 Amazon CLI 的 MemoryDB 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 MemoryDB 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`copy-snapshot`

以下代码示例演示了如何使用 `copy-snapshot`。

Amazon CLI

复制快照

以下 `copy-snapshot` 示例创建快照的副本。

```
aws memorydb copy-snapshot \
```

```
    --source-snapshot-name my-cluster-snapshot \
    --target-snapshot-name my-cluster-snapshot-copy
```

Output

```
{
    "Snapshot": {
        "Name": "my-cluster-snapshot-copy",
        "Status": "creating",
        "Source": "manual",
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:snapshot/my-cluster-
snapshot-copy",
        "ClusterConfiguration": {
            "Name": "my-cluster",
            "Description": " ",
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "Port": 6379,
            "ParameterGroupName": "default.memorydb-redis6",
            "SubnetGroupName": "my-sg",
            "VpcId": "vpc-xx2574fc",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "04:30-05:30",
            "NumShards": 2
        }
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的复制快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopySnapshot。

create-acl

以下代码示例演示了如何使用 create-acl。

Amazon CLI

创建 ACL

以下 create-acl 示例创建新的访问控制列表。

```
aws memorydb create-acl \
    --acl-name "new-acl-1" \
    --user-names "my-user"
```

输出：

```
{
    "ACL": {
        "Name": "new-acl-1",
        "Status": "creating",
        "UserNames": [
            "my-user"
        ],
        "MinimumEngineVersion": "6.2",
        "Clusters": [],
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/new-acl-1"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAcl。

create-cluster

以下代码示例演示了如何使用 create-cluster。

Amazon CLI

创建集群

以下 create-cluster 示例创建一个新的集群。

```
aws memorydb create-cluster \
    --cluster-name my-new-cluster \
    --node-type db.r6g.large \
    --acl-name my-acl \
    --subnet-group my-sg
```

输出：

```
{
```

```
    "Cluster": {
        "Name": "my-new-cluster",
        "Status": "creating",
        "NumberOfShards": 1,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:cluster/my-new-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "sat:10:00-sat:11:00",
        "SnapshotWindow": "07:30-08:30",
        "ACLName": "my-acl",
        "AutoMinorVersionUpgrade": true
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的管理集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-parameter-group

以下代码示例演示了如何使用 create-parameter-group。

Amazon CLI

创建参数组

以下 create-parameter-group 示例创建一个参数组。

```
aws memorydb create-parameter-group \
    --parameter-group-name myRedis6x \
    --family memorydb_redis6 \
    --description "my-parameter-group"
```

输出：

```
{
    "ParameterGroup": {
        "Name": "myredis6x",
        "Family": "memorydb_redis6",
        "Description": "my-parameter-group",
        "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:parametergroup/myredis6x"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的创建参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateParameterGroup。

create-snapshot

以下代码示例演示了如何使用 create-snapshot。

Amazon CLI

创建快照

以下 create-snapshot 示例创建快照。

```
aws memorydb create-snapshot \
    --cluster-name my-cluster \
    --snapshot-name my-cluster-snapshot
```

输出：

```
{
    "Snapshot": {
        "Name": "my-cluster-snapshot1",
        "Status": "creating",
        "Source": "manual",
        "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:snapshot/my-cluster-
snapshot",
        "ClusterConfiguration": {
            "Name": "my-cluster",
            "Description": "",
            "NodeType": "db.r6g.large",
```

```
            "EngineVersion": "6.2",
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "Port": 6379,
            "ParameterGroupName": "default.memorydb-redis6",
            "SubnetGroupName": "my-sg",
            "VpcId": "vpc-862xxxxc",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "04:30-05:30",
            "NumShards": 2
        }
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的生成手动快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshot。

create-subnet-group

以下代码示例演示了如何使用 create-subnet-group。

Amazon CLI

创建子网组

以下 create-subnet-group 示例创建一个子网组。

```
aws memorydb create-subnet-group \
    --subnet-group-name mysubnetgroup \
    --description "my subnet group" \
    --subnet-ids subnet-5623xxxx
```

输出：

```
{
    "SubnetGroup": {
        "Name": "mysubnetgroup",
        "Description": "my subnet group",
        "VpcId": "vpc-86257xxx",
        "Subnets": [
            {
                "Identifier": "subnet-5623xxxx",
```

```
                "AvailabilityZone": {
                    "Name": "us-east-1a"
                }
            }
        ],
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/mysubnetgroup"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的创建子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubnetGroup。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

创建用户

以下 create-user 示例创建一个新的用户。

```
aws memorydb create-user \
    --user-name user-name-1 \
    --access-string "~objects:* ~items:* ~public:*" \
     --authentication-mode \
         Passwords="enterapasswordhere",Type=password
```

输出：

```
{
    "User": {
        "Name": "user-name-1",
        "Status": "active",
        "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",
        "ACLNames": [],
        "MinimumEngineVersion": "6.2",
        "Authentication": {
            "Type": "password",
            "PasswordCount": 1
        },
        "ARN": "arn:aws:memorydb:us-west-2:491658xxxxxx:user/user-name-1"
```

```
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

delete-acl

以下代码示例演示了如何使用 delete-acl。

Amazon CLI

删除 ACL

以下 delete-acl 示例删除访问控制列表。

```
aws memorydb delete-acl \
    --acl-name "new-acl-1"
```

输出：

```
{
    "ACL": {
        "Name": "new-acl-1",
        "Status": "deleting",
        "UserNames": [
            "pat"
        ],
        "MinimumEngineVersion": "6.2",
        "Clusters": [],
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/new-acl-1"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAcl。

delete-cluster

以下代码示例演示了如何使用 delete-cluster。

Amazon CLI

删除集群

以下 delete-cluster 示例删除集群。

```
aws memorydb delete-cluster \
    --cluster-name my-new-cluster
```

输出：

```
{
    "Cluster": {
        "Name": "my-new-cluster",
        "Status": "deleting",
        "NumberOfShards": 1,
        "ClusterEndpoint": {
            "Address": "clustercfg.my-new-cluster.xxxxx.memorydb.us-
east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-new-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "sat:10:00-sat:11:00",
        "SnapshotWindow": "07:30-08:30",
        "AutoMinorVersionUpgrade": true
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的删除集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCluster。

delete-parameter-group

以下代码示例演示了如何使用 delete-parameter-group。

Amazon CLI

删除参数组

以下 delete-parameter-group 示例删除一个参数组。

```
aws memorydb delete-parameter-group \
    --parameter-group-name myRedis6x
```

输出：

```
{
    "ParameterGroup": {
        "Name": "myredis6x",
        "Family": "memorydb_redis6",
        "Description": "my-parameter-group",
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/myredis6x"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的删除参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteParameterGroup。

delete-snapshot

以下代码示例演示了如何使用 delete-snapshot。

Amazon CLI

删除快照

以下 delete-snapshot 示例删除快照。

```
aws memorydb delete-snapshot \
    --snapshot-name my-cluster-snapshot
```

输出：

```
{
    "Snapshot": {
```

```
        "Name": "my-cluster-snapshot",
        "Status": "deleting",
        "Source": "manual",
        "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:snapshot/my-cluster-
snapshot",
        "ClusterConfiguration": {
            "Name": "my-cluster",
            "Description": "",
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "Port": 6379,
            "ParameterGroupName": "default.memorydb-redis6",
            "SubnetGroupName": "my-sg",
            "VpcId": "vpc-862xxxxc",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "04:30-05:30",
            "NumShards": 2
        }
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的删除快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSnapshot。

delete-subnet-group

以下代码示例演示了如何使用 delete-subnet-group。

Amazon CLI

删除子网组

以下 delete-subnet-group 示例删除子网组。

```
aws memorydb delete-subnet-group \
    --subnet-group-name mysubnetgroup
```

输出：

```
{
```

```
    "SubnetGroup": {
        "Name": "mysubnetgroup",
        "Description": "my subnet group",
        "VpcId": "vpc-86xxxx4fc",
        "Subnets": [
            {
                "Identifier": "subnet-56xxx61b",
                "AvailabilityZone": {
                    "Name": "us-east-1a"
                }
            }
        ],
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/mysubnetgroup"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的删除子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubnetGroup。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

以下 delete-user 示例删除用户。

```
aws memorydb delete-user \
    --user-name my-user
```

输出：

```
{
    "User": {
        "Name": "my-user",
        "Status": "deleting",
        "AccessString": "on ~app::* resetchannels -@all +@read",
        "ACLNames": [
            "my-acl"
        ],
```

```
        "MinimumEngineVersion": "6.2",
        "Authentication": {
            "Type": "password",
            "PasswordCount": 1
        },
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-acls

以下代码示例演示了如何使用 describe-acls。

Amazon CLI

返回 ACL 列表

以下 describe-acls 命令返回 ACL 列表。

```
aws memorydb describe-acls
```

输出：

```
{
    "ACLs": [
        {
            "Name": "open-access",
            "Status": "active",
            "UserNames": [
                "default"
            ],
            "MinimumEngineVersion": "6.2",
            "Clusters": [],
            "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/open-access"
        },
        {
            "Name": my-acl",
            "Status": "active",
            "UserNames": [],
```

```
            "MinimumEngineVersion": "6.2",
            "Clusters": [
                "my-cluster"
            ],
            "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxxx:acl/my-acl"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAcls。

describe-clusters

以下代码示例演示了如何使用 describe-clusters。

Amazon CLI

返回集群列表

以下 describe-clusters 命令返回集群的列表。

```
aws memorydb describe-clusters
```

输出：

```
{
    "Clusters": [
        {
                "Name": "my-cluster",
                "Status": "available",
                "NumberOfShards": 2,
                "ClusterEndpoint": {
                    "Address": "clustercfg.my-cluster.llru6f.memorydb.us-
east-1.amazonaws.com",
                    "Port": 6379
                },
                "NodeType": "db.r6g.large",
                "EngineVersion": "6.2",
                "EnginePatchVersion": "6.2.6",
                "ParameterGroupName": "default.memorydb-redis6",
                "ParameterGroupStatus": "in-sync",
```

```
                "SecurityGroups": [
                    {
                        "SecurityGroupId": "sg-0a1434xxxxxc9fae",
                        "Status": "active"
                    }
                ],
                "SubnetGroupName": "pat-sg",
                "TLSEnabled": true,
                "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:cluster/my-cluster",
                "SnapshotRetentionLimit": 0,
                "MaintenanceWindow": "wed:03:00-wed:04:00",
                "SnapshotWindow": "04:30-05:30",
                "ACLName": "my-acl",
                "AutoMinorVersionUpgrade": true
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的管理集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusters。

describe-engine-versions

以下代码示例演示了如何使用 describe-engine-versions。

Amazon CLI

返回引擎版本列表

以下 describe-engine-versions 命令返回引擎版本列表。

```
aws memorydb describe-engine-versions
```

输出：

```
{
    "EngineVersions": [
        {
            "EngineVersion": "6.2",
            "EnginePatchVersion": "6.2.6",
            "ParameterGroupFamily": "memorydb_redis6"
        }
```

```
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的引擎版本和升级。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngineVersions。

`describe-events`

以下代码示例演示了如何使用 `describe-events`。

Amazon CLI

返回事件列表

以下 describe-events 命令返回事件列表。

```
aws memorydb describe-events
```

输出：

```
{
    "Events": [
        {
            "SourceName": "my-cluster",
            "SourceType": "cluster",
            "Message": "Increase replica count started for replication group my-
cluster on 2022-07-22T14:09:01.440Z",
            "Date": "2022-07-22T07:09:01.443000-07:00"
        },
        {
            "SourceName": "my-user",
            "SourceType": "user",
            "Message": "Create user my-user operation completed.",
            "Date": "2022-07-22T07:00:02.975000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的监控事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-parameter-groups

以下代码示例演示了如何使用 describe-parameter-groups。

Amazon CLI

返回参数组列表

以下 describe-parameter-groups 返回参数组列表。

```
aws memorydb describe-parameter-groups
```

输出：

```
{
    "ParameterGroups": [
        {
            "Name": "default.memorydb-redis6",
            "Family": "memorydb_redis6",
            "Description": "Default parameter group for memorydb_redis6",
            "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/
default.memorydb-redis6"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用参数组配置引擎参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeParameterGroups。

describe-parameters

以下代码示例演示了如何使用 describe-parameters。

Amazon CLI

返回参数列表

以下 describe-parameters 返回参数列表。

```
aws memorydb describe-parameters
```

输出：


```
{
    "Parameters": [
        {
            "Name": "acllog-max-len",
            "Value": "128",
            "Description": "The maximum length of the ACL Log",
            "DataType": "integer",
            "AllowedValues": "1-10000",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "activedefrag",
            "Value": "no",
            "Description": "Enabled active memory defragmentation",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-defrag-cycle-max",
            "Value": "75",
            "Description": "Maximal effort for defrag in CPU percentage",
            "DataType": "integer",
            "AllowedValues": "1-75",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-defrag-cycle-min",
            "Value": "5",
            "Description": "Minimal effort for defrag in CPU percentage",
            "DataType": "integer",
            "AllowedValues": "1-75",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-defrag-ignore-bytes",
            "Value": "104857600",
            "Description": "Minimum amount of fragmentation waste to start active
 defrag",
            "DataType": "integer",
            "AllowedValues": "1048576-",
            "MinimumEngineVersion": "6.2.4"
```

```
        },
        {
            "Name": "active-defrag-max-scan-fields",
            "Value": "1000",
            "Description": "Maximum number of set/hash/zset/list fields that will be
processed from the main dictionary scan",
            "DataType": "integer",
            "AllowedValues": "1-1000000",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-defrag-threshold-lower",
            "Value": "10",
            "Description": "Minimum percentage of fragmentation to start active
defrag",
            "DataType": "integer",
            "AllowedValues": "1-100",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-defrag-threshold-upper",
            "Value": "100",
            "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
            "DataType": "integer",
            "AllowedValues": "1-100",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "active-expire-effort",
            "Value": "1",
            "Description": "The amount of effort that redis uses to expire items in
the active expiration job",
            "DataType": "integer",
            "AllowedValues": "1-10",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "activerehashing",
            "Value": "yes",
            "Description": "Apply rehashing or not",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
```

```
        },
        {
            "Name": "client-output-buffer-limit-normal-hard-limit",
            "Value": "0",
            "Description": "Normal client output buffer hard limit in bytes",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "client-output-buffer-limit-normal-soft-limit",
            "Value": "0",
            "Description": "Normal client output buffer soft limit in bytes",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "client-output-buffer-limit-normal-soft-seconds",
            "Value": "0",
            "Description": "Normal client output buffer soft limit in seconds",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "client-output-buffer-limit-pubsub-hard-limit",
            "Value": "33554432",
            "Description": "Pubsub client output buffer hard limit in bytes",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "client-output-buffer-limit-pubsub-soft-limit",
            "Value": "8388608",
            "Description": "Pubsub client output buffer soft limit in bytes",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "client-output-buffer-limit-pubsub-soft-seconds",
            "Value": "60",
```

```
            "Description": "Pubsub client output buffer soft limit in seconds",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "hash-max-ziplist-entries",
            "Value": "512",
            "Description": "The maximum number of hash entries in order for the
dataset to be compressed",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "hash-max-ziplist-value",
            "Value": "64",
            "Description": "The threshold of biggest hash entries in order for the
dataset to be compressed",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "hll-sparse-max-bytes",
            "Value": "3000",
            "Description": "HyperLogLog sparse representation bytes limit",
            "DataType": "integer",
            "AllowedValues": "1-16000",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lazyfree-lazy-eviction",
            "Value": "no",
            "Description": "Perform an asynchronous delete on evictions",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lazyfree-lazy-expire",
            "Value": "no",
            "Description": "Perform an asynchronous delete on expired keys",
            "DataType": "string",
```

```
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lazyfree-lazy-server-del",
            "Value": "no",
            "Description": "Perform an asynchronous delete on key updates",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lazyfree-lazy-user-del",
            "Value": "no",
            "Description": "Specifies whether the default behavior of DEL command
acts the same as UNLINK",
            "DataType": "string",
            "AllowedValues": "yes,no",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lfu-decay-time",
            "Value": "1",
            "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policyd",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "lfu-log-factor",
            "Value": "10",
            "Description": "The log factor for incrementing key counter for LFU
eviction policy",
            "DataType": "integer",
            "AllowedValues": "1-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "list-compress-depth",
            "Value": "0",
            "Description": "Number of quicklist ziplist nodes from each side of
the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
```

```
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "maxmemory-policy",
            "Value": "noeviction",
            "Description": "Max memory policy",
            "DataType": "string",
            "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-
lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "maxmemory-samples",
            "Value": "3",
            "Description": "Max memory samples",
            "DataType": "integer",
            "AllowedValues": "1-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "notify-keyspace-events",
            "Description": "The keyspace events for Redis to notify Pub/Sub clients
 about. By default all notifications are disabled",
            "DataType": "string",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "set-max-intset-entries",
            "Value": "512",
            "Description": "The limit in the size of the set in order for the
 dataset to be compressed",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "slowlog-log-slower-than",
            "Value": "10000",
            "Description": "The execution time, in microseconds, to exceed in order
 for the command to get logged. Note that a negative number disables the slow log,
 while a value of zero forces the logging of every command",
            "DataType": "integer",
```

```
            "AllowedValues": "-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "slowlog-max-len",
            "Value": "128",
            "Description": "The length of the slow log. There is no limit to this
length. Just be aware that it will consume memory. You can reclaim memory used by
the slow log with SLOWLOG RESET.",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "stream-node-max-bytes",
            "Value": "4096",
            "Description": "The maximum size of a single node in a stream in bytes",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "stream-node-max-entries",
            "Value": "100",
            "Description": "The maximum number of items a single node in a stream
can contain",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "tcp-keepalive",
            "Value": "300",
            "Description": "If non-zero, send ACKs every given number of seconds",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "timeout",
            "Value": "0",
            "Description": "Close connection if client is idle for a given number of
seconds, or never if 0",
            "DataType": "integer",
```

```
            "AllowedValues": "0,20-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "tracking-table-max-keys",
            "Value": "1000000",
            "Description": "The maximum number of keys allowed for the tracking
 table for client side caching",
            "DataType": "integer",
            "AllowedValues": "1-100000000",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "zset-max-ziplist-entries",
            "Value": "128",
            "Description": "The maximum number of sorted set entries in order for
 the dataset to be compressed",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        },
        {
            "Name": "zset-max-ziplist-value",
            "Value": "64",
            "Description": "The threshold of biggest sorted set entries in order for
 the dataset to be compressed",
            "DataType": "integer",
            "AllowedValues": "0-",
            "MinimumEngineVersion": "6.2.4"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用参数组配置引擎参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeParameters。

describe-snapshots

以下代码示例演示了如何使用 describe-snapshots。

Amazon CLI

返回快照列表

以下 describe-snapshots 命令返回快照列表。

```
aws memorydb describe-snapshots
```

输出：

```
{
"Snapshots": [
    {
        "Name": "my-cluster-snapshot",
        "Status": "available",
        "Source": "manual",
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx2:snapshot/my-cluster-
snapshot",
        "ClusterConfiguration": {
            "Name": "my-cluster",
            "Description": " ",
            "NodeType": "db.r6g.large",
            "EngineVersion": "6.2",
            "MaintenanceWindow": "wed:03:00-wed:04:00",
            "Port": 6379,
            "ParameterGroupName": "default.memorydb-redis6",
            "SubnetGroupName": "my-sg",
            "VpcId": "vpc-862574fc",
            "SnapshotRetentionLimit": 0,
            "SnapshotWindow": "04:30-05:30",
            "NumShards": 2
        }
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的快照和恢复。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshots。

describe-subnet-groups

以下代码示例演示了如何使用 describe-subnet-groups。

Amazon CLI

返回子网组列表

以下 describe-subnet-groups 命令返回子网组列表。

```
aws memorydb describe-subnet-groups
```

Output

```
{
    "SubnetGroups": [
        {
            "Name": "my-sg",
            "Description": "pat-sg",
            "VpcId": "vpc-86xxx4fc",
            "Subnets": [
                {
                    "Identifier": "subnet-faxx84a6",
                    "AvailabilityZone": {
                        "Name": "us-east-1b"
                    }
                },
                {
                    "Identifier": "subnet-56xxf61b",
                    "AvailabilityZone": {
                        "Name": "us-east-1a"
                    }
                }
            ],
            "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:subnetgroup/my-sg"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的子网和子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSubnetGroups。

describe-users

以下代码示例演示了如何使用 describe-users。

Amazon CLI

返回用户列表

以下 describe-users 命令返回用户的列表。

```
aws memorydb describe-users
```

Output

```
{
    "Users": [
        {
            "Name": "default",
            "Status": "active",
            "AccessString": "on ~* &* +@all",
            "ACLNames": [
                "open-access"
            ],
            "MinimumEngineVersion": "6.0",
            "Authentication": {
                "Type": "no-password"
            },
            "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/default"
        },
        {
            "Name": "my-user",
            "Status": "active",
            "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",
            "ACLNames": [],
            "MinimumEngineVersion": "6.2",
            "Authentication": {
                "Type": "password",
                "PasswordCount": 2
            },
            "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUsers。

failover-shard

以下代码示例演示了如何使用 failover-shard。

Amazon CLI

转移失效分片

以下 failover-shard 命令对分片进行失效转移。

```
aws memorydb failover-shard \
    --cluster-name my-cluster --shard-name 0001
```

输出：

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "available",
        "NumberOfShards": 2,
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SecurityGroups": [
            {
                "SecurityGroupId": "sg-0a143xxxx45c9fae",
                "Status": "active"
            }
        ],
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
        "AutoMinorVersionUpgrade": true
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的通过多可用区最大程度地减少停机时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 FailoverShard。

`list-allowed-node-type-updates`

以下代码示例演示了如何使用 `list-allowed-node-type-updates`。

Amazon CLI

返回允许的节点类型更新的列表

以下 list-allowed-node-type-updates 返回可用节点类型更新的列表。

```
aws memorydb list-allowed-node-type-updates
```

输出：

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "available",
        "NumberOfShards": 2,
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "default.memorydb-redis6",
        "ParameterGroupStatus": "in-sync",
        "SecurityGroups": [
            {
                "SecurityGroupId": "sg-0a143xxxx45c9fae",
                "Status": "active"
            }
        ],
        "SubnetGroupName": "my-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
        "SnapshotWindow": "04:30-05:30",
```

```
        "AutoMinorVersionUpgrade": true
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的扩展。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAllowedNodeTypeUpdates。

`list-tags`

以下代码示例演示了如何使用 `list-tags`。

Amazon CLI

返回标签列表

以下 list-tags 命令返回标签列表。

```
aws memorydb list-tags \
    --resource-arn arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster
```

输出：

```
{
    "TagList": [
        {
            "Key": "mytag",
            "Value": "myvalue"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTags。

`reset-parameter-group`

以下代码示例演示了如何使用 `reset-parameter-group`。

Amazon CLI

重置参数组

以下 reset-parameter-group 命令重置参数组。

```
aws memorydb reset-parameter-group \
    --parameter-group-name my-parameter-group \
    --all-parameters
```

输出：

```
{
    "ParameterGroup": {
        "Name": "my-parameter-group",
        "Family": "memorydb_redis6",
        "Description": "my parameter group",
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/my-parameter-
group"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用参数组配置引擎参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetParameterGroup。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

标记资源

以下 tag-resource 命令将标签添加到资源中。

```
aws memorydb tag-resource \
    --resource-arn arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster \
    --tags Key="mykey",Value="myvalue"
```

输出：

```
{
    "TagList": [
        {
```

```
            "Key": "mytag",
            "Value": "myvalue"
        },
        {
            "Key": "mykey",
            "Value": "myvalue"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

更新 ACL

以下 update-acl 通过添加用户来更新 ACL。

```
aws memorydb untag-resource \
    --resource-arn arn:aws:memorydb:us-east-1:491658xxxxx:cluster/my-cluster \
    --tag-keys mykey
```

输出：

```
{
    "TagList": [
        {
            "Key": "mytag",
            "Value": "myvalue"
        }
    ]
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-cluster

以下代码示例演示了如何使用 update-cluster。

Amazon CLI

更新集群

以下 update-cluster 命令将集群的参数组更新为 my-parameter-group。

```
aws memorydb update-cluster \
    --cluster-name my-cluster \
    --parameter-group-name my-parameter-group
```

输出：

```
{
    "Cluster": {
        "Name": "my-cluster",
        "Status": "available",
        "NumberOfShards": 2,
        "AvailabilityMode": "MultiAZ",
        "ClusterEndpoint": {
            "Address": "clustercfg.my-cluster.llru6f.memorydb.us-
east-1.amazonaws.com",
            "Port": 6379
        },
        "NodeType": "db.r6g.large",
        "EngineVersion": "6.2",
        "EnginePatchVersion": "6.2.6",
        "ParameterGroupName": "my-parameter-group",
        "ParameterGroupStatus": "in-sync",
        "SecurityGroups": [
            {
                "SecurityGroupId": "sg-0a143xxxxxc9fae",
                "Status": "active"
            }
        ],
        "SubnetGroupName": "pat-sg",
        "TLSEnabled": true,
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",
        "SnapshotRetentionLimit": 0,
        "MaintenanceWindow": "wed:03:00-wed:04:00",
```

```
        "SnapshotWindow": "04:30-05:30",
        "ACLName": "my-acl",
        "AutoMinorVersionUpgrade": true
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的修改集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCluster。

update-parameter-group

以下代码示例演示了如何使用 update-parameter-group。

Amazon CLI

更新参数组

以下 update-parameter-group 更新一个参数组。

```
aws memorydb update-parameter-group \
    --parameter-group-name my-parameter-group \
    --parameter-name-values "ParameterName=activedefrag, ParameterValue=no"
```

输出：

```
{
    "ParameterGroup": {
        "Name": "my-parameter-group",
        "Family": "memorydb_redis6",
        "Description": "my parameter group",
        "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:parametergroup/my-parameter-
group"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的修改参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateParameterGroup。

update-subnet-group

以下代码示例演示了如何使用 update-subnet-group。

Amazon CLI

更新子网组

以下 update-subnet-group 命令更新子网组的子网 ID。

```
aws memorydb update-subnet-group \
    --subnet-group-name my-sg \
    --subnet-ids subnet-01f29d458f3xxxxx
```

输出：

```
{
    "SubnetGroup": {
        "Name": "my-sg-1",
        "Description": "my-sg",
        "VpcId": "vpc-09d2cfc01xxxxxxx",
        "Subnets": [
            {
                "Identifier": "subnet-01f29d458fxxxxxx",
                "AvailabilityZone": {
                    "Name": "us-east-1a"
                }
            }
        ],
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/my-sg"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的子网和子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubnetGroup。

update-user

以下代码示例演示了如何使用 update-user。

Amazon CLI

更新用户

以下 update-user 命令修改用户的访问字符串。

```
aws memorydb update-user \
    --user-name my-user \
    --access-string "off ~objects:* ~items:* ~public:* resetchannels -@all"
```

输出：

```
{
    "User": {
        "Name": "my-user",
        "Status": "modifying",
        "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",
        "ACLNames": [
            "myt-acl"
        ],
        "MinimumEngineVersion": "6.2",
        "Authentication": {
            "Type": "password",
            "PasswordCount": 2
        },
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"
    }
}
```

有关更多信息，请参阅《MemoryDB 用户指南》中的使用访问控制列表对用户进行身份验证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUser。

使用 Amazon CLI 的 Amazon MSK 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon MSK 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-cluster`

以下代码示例演示了如何使用 `create-cluster`。

Amazon CLI

创建 Amazon MSK 集群

以下 `create-cluster` 示例创建了一个名为 `MessagingCluster` 的 MSK 集群，其中包含三个代理节点。名为 `brokernodegroupinfo.json` 的 JSON 文件指定了三个子网，您希望 Amazon MSK 在这三个子网上分配代理节点。此示例未指定监控级别，因此集群获得 `DEFAULT` 级别。

```
aws kafka create-cluster \
    --cluster-name "MessagingCluster" \
    --broker-node-group-info file://brokernodegroupinfo.json \
    --kafka-version "2.2.1" \
    --number-of-broker-nodes 3
```

的内容`brokernodegroupinfo.json`：

```
{
    "InstanceType": "kafka.m5.xlarge",
    "BrokerAZDistribution": "DEFAULT",
    "ClientSubnets": [
        "subnet-0123456789111abcd",
        "subnet-0123456789222abcd",
        "subnet-0123456789333abcd"
    ]
}
```

输出：

```
{
    "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
    "ClusterName": "MessagingCluster",
    "State": "CREATING"
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka》中的创建 Amazon MSK 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-configuration

以下代码示例演示了如何使用 create-configuration。

Amazon CLI

创建自定义 Amazon MSK 配置

以下 create-configuration 示例使用在输入文件中指定的服务器属性创建一个自定义 MSK 配置。

```
aws kafka create-configuration \
    --name "CustomConfiguration" \
    --description "Topic autocreation enabled; Apache ZooKeeper timeout 2000 ms; Log
 rolling 604800000 ms." \
    --kafka-versions "2.2.1" \
    --server-properties file://configuration.txt
```

的内容configuration.txt：

```
auto.create.topics.enable = true
zookeeper.connection.timeout.ms = 2000
log.roll.ms = 604800000
```

此命令不生成任何输出。输出：

```
{
    "Arn": "arn:aws:kafka:us-west-2:123456789012:configuration/CustomConfiguration/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
    "CreationTime": "2019-10-09T15:26:05.548Z",
    "LatestRevision":
        {
            "CreationTime": "2019-10-09T15:26:05.548Z",
            "Description": "Topic autocreation enabled; Apache ZooKeeper timeout
 2000 ms; Log rolling 604800000 ms.",
            "Revision": 1
```

```
        },
    "Name": "CustomConfiguration"
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的 Amazon MSK 配置操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConfiguration。

describe-cluster

以下代码示例演示了如何使用 describe-cluster。

Amazon CLI

描述集群

以下 describe-cluster 示例描述一个 Amazon MSK 集群。

```
aws kafka describe-cluster \
    --cluster-arn arn:aws:kafka:us-east-1:123456789012:cluster/demo-cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5
```

输出：

```
{
    "ClusterInfo": {
        "BrokerNodeGroupInfo": {
            "BrokerAZDistribution": "DEFAULT",
            "ClientSubnets": [
                "subnet-cbfff283",
                "subnet-6746046b"
            ],
            "InstanceType": "kafka.m5.large",
            "SecurityGroups": [
                "sg-f839b688"
            ],
            "StorageInfo": {
                "EbsStorageInfo": {
                    "VolumeSize": 100
                }
            }
```

```
        },
        "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/demo-
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5",
        "ClusterName": "demo-cluster-1",
        "CreationTime": "2020-07-09T02:31:36.223000+00:00",
        "CurrentBrokerSoftwareInfo": {
            "KafkaVersion": "2.2.1"
        },
        "CurrentVersion": "K3AEGXETSR30VB",
        "EncryptionInfo": {
            "EncryptionAtRest": {
                "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a7ca56d5-0768-4b64-a670-339a9fbef81c"
            },
            "EncryptionInTransit": {
                "ClientBroker": "TLS_PLAINTEXT",
                "InCluster": true
            }
        },
        "EnhancedMonitoring": "DEFAULT",
        "OpenMonitoring": {
            "Prometheus": {
                "JmxExporter": {
                    "EnabledInBroker": false
                },
                "NodeExporter": {
                    "EnabledInBroker": false
                }
            }
        },
        "NumberOfBrokerNodes": 2,
        "State": "ACTIVE",
        "Tags": {},
        "ZookeeperConnectString": "z-2.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-1.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-3.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181"
    }
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的列出 Amazon MSK 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCluster。

get-bootstrap-brokers

以下代码示例演示了如何使用 get-bootstrap-brokers。

Amazon CLI

获取引导代理

以下 get-bootstrap-brokers 示例检索 Amazon MSK 集群的引导代理信息。

```
aws kafka get-bootstrap-brokers \
    --cluster-arn arn:aws:kafka:us-east-1:123456789012:cluster/demo-cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5
```

输出：

```
{
    "BootstrapBrokerString": "b-1.demo-cluster-1.xuy0sb.c5.kafka.us-east-1.amazonaws.com:9092,b-2.demo-cluster-1.xuy0sb.c5.kafka.us-east-1.amazonaws.com:9092",
    "BootstrapBrokerStringTls": "b-1.demo-cluster-1.xuy0sb.c5.kafka.us-east-1.amazonaws.com:9094,b-2.demo-cluster-1.xuy0sb.c5.kafka.us-east-1.amazonaws.com:9094"
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的获取引导代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBootstrapBrokers。

list-clusters

以下代码示例演示了如何使用 list-clusters。

Amazon CLI

列出可用集群

以下 list-clusters 示例列出您的 Amazon 账户中的 Amazon MSK 集群。

```
aws kafka list-clusters
```

输出：


```
{
    "ClusterInfoList": [
        {
            "BrokerNodeGroupInfo": {
                "BrokerAZDistribution": "DEFAULT",
                "ClientSubnets": [
                    "subnet-cbfff283",
                    "subnet-6746046b"
                ],
                "InstanceType": "kafka.m5.large",
                "SecurityGroups": [
                    "sg-f839b688"
                ],
                "StorageInfo": {
                    "EbsStorageInfo": {
                        "VolumeSize": 100
                    }
                }
            },
            "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/demo-
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5",
            "ClusterName": "demo-cluster-1",
            "CreationTime": "2020-07-09T02:31:36.223000+00:00",
            "CurrentBrokerSoftwareInfo": {
                "KafkaVersion": "2.2.1"
            },
            "CurrentVersion": "K3AEGXETSR30VB",
            "EncryptionInfo": {
                "EncryptionAtRest": {
                    "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a7ca56d5-0768-4b64-a670-339a9fbef81c"
                },
                "EncryptionInTransit": {
                    "ClientBroker": "TLS_PLAINTEXT",
                    "InCluster": true
                }
            },
            "EnhancedMonitoring": "DEFAULT",
            "OpenMonitoring": {
                "Prometheus": {
                    "JmxExporter": {
                        "EnabledInBroker": false
```

```
                    },
                    "NodeExporter": {
                        "EnabledInBroker": false
                    }
                }
            },
            "NumberOfBrokerNodes": 2,
            "State": "ACTIVE",
            "Tags": {},
            "ZookeeperConnectString": "z-2.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-1.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-3.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的列出 Amazon MSK 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListClusters。

update-broker-storage

以下代码示例演示了如何使用 update-broker-storage。

Amazon CLI

更新代理的 EBS 存储

以下 update-broker-storage 示例更新了集群中所有代理的 EBS 存储量。Amazon MSK 将每个代理的目标存储量设置为示例中指定的量。您可以通过描述集群或列出所有集群来获取集群的当前版本。

```
aws kafka update-broker-storage \
    --cluster-arn "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2" \
    --current-version "K21V3IB1VIZYYH" \
    --target-broker-ebs-volume-info "KafkaBrokerNodeId=ALL,VolumeSizeGB=1100"
```

输出返回此 update-broker-storage 操作的 ARN。要确定此操作是否已完成，请使用 describe-cluster-operation 命令并以此 ARN 作为输入。

```
{
    "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
    "ClusterOperationArn": "arn:aws:kafka:us-west-2:123456789012:cluster-
operation/V123450123/a1b2c3d4-1234-abcd-cdef-22222EXAMPLE-2/a1b2c3d4-abcd-1234-
bcde-33333EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的更新代理的 EBS 存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateBrokerStorage。

update-cluster-configuration

以下代码示例演示了如何使用 update-cluster-configuration。

Amazon CLI

更新 Amazon MSK 集群的配置

以下 update-cluster-configuration 示例更新了指定的现有 MSK 集群的配置。它使用自定义 MSK 配置。

```
aws kafka update-cluster-configuration \
    --cluster-arn "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2" \
    --configuration-info file://configuration-info.json \
    --current-version "K21V3IB1VIZYYH"
```

的内容configuration-info.json：

```
{
    "Arn": "arn:aws:kafka:us-west-2:123456789012:configuration/CustomConfiguration/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
    "Revision": 1
}
```

输出返回此 update-cluster-configuration 操作的 ARN。要确定此操作是否已完成，请使用 describe-cluster-operation 命令并以此 ARN 作为输入。

```
{
    "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
    "ClusterOperationArn": "arn:aws:kafka:us-west-2:123456789012:cluster-
operation/V123450123/a1b2c3d4-1234-abcd-cdef-22222EXAMPLE-2/a1b2c3d4-abcd-1234-
bcde-33333EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Managed Streaming for Apache Kafka 开发人员指南》中的更新 Amazon MSK 集群的配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateClusterConfiguration。

使用 Amazon CLI 的 Network Flow Monitor 示例

以下代码示例演示如何通过将 Amazon Command Line Interface 与 Network Flow Monitor 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-monitor`

以下代码示例演示了如何使用 `create-monitor`。

Amazon CLI

创建监测仪

以下 `create-monitor` 示例在指定账户中创建名为 demo 的监测仪。

```
aws networkflowmonitor create-monitor \
```

```
    --monitor-name demo \
    --local-resources type="AWS::EC2::VPC",identifier="arn:aws:ec2:us-
east-1:123456789012:vpc/vpc-03ea55eeda25adbb0"  \
    --scope-arn arn:aws:networkflowmonitor:us-east-1:123456789012:scope/
e21cda79-30a0-4c12-9299-d8629d76d8cf
```

输出：

```
{
    "monitorArn": "arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/demo",
    "monitorName": "demo",
    "monitorStatus": "ACTIVE",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的在 Network Flow Monitor 中创建监测仪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMonitor。

create-scope

以下代码示例演示了如何使用 create-scope。

Amazon CLI

创建范围

以下 create-scope 示例创建一个范围，其中包括一组资源，Network Flow Monitor 将为这些资源生成网络流量指标。

```
aws networkflowmonitor create-scope \
    --targets '[{"targetIdentifier":{"targetId":
{"accountId":"123456789012"},"targetType":"ACCOUNT"},"region":"us-east-1"}]'
```

输出：

```
{
    "scopeId": "97626f8d-8a21-4b5d-813a-1a0962dd4615",
    "status": "IN_PROGRESS",
    "tags": {}
```

```
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateScope。

delete-monitor

以下代码示例演示了如何使用 delete-monitor。

Amazon CLI

删除监测仪

以下 delete-monitor 示例从指定账户中删除名为 demo 的监测仪。

```
aws networkflowmonitor delete-monitor \
    --monitor-name demo
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的在 Network Flow Monitor 中删除监测仪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMonitor。

delete-scope

以下代码示例演示了如何使用 delete-scope。

Amazon CLI

删除范围

以下 delete-scope 示例删除指定的范围。

```
aws networkflowmonitor delete-scope \
    --scope-id fdc20616-6bb4-4242-a24e-a748e65ca7ac
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteScope。

`get-monitor`

以下代码示例演示了如何使用 `get-monitor`。

Amazon CLI

检索有关监测仪的信息

以下 `get-monitor` 示例显示有关指定账户中名为 demo 的监测仪的信息。

```
aws networkflowmonitor get-monitor \
    --monitor-name Demo
```

输出：

```
{
    "monitorArn": "arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/Demo",
    "monitorName": "Demo",
    "monitorStatus": "ACTIVE",
    "localResources": [
        {
            "type": "AWS::EC2::VPC",
            "identifier": "arn:aws:ec2:us-east-1:123456789012:vpc/
vpc-03ea55eeda25adbb0"
        }
    ],
    "remoteResources": [],
    "createdAt": "2024-12-09T12:21:51.616000-06:00",
    "modifiedAt": "2024-12-09T12:21:55.412000-06:00",
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMonitor。

get-query-results-workload-insights-top-contributors-data

以下代码示例演示了如何使用 get-query-results-workload-insights-top-contributors-data。

Amazon CLI

检索有关工作负载见解的排名靠前的贡献者数据

以下 get-query-results-workload-insights-top-contributors-data 示例返回指定查询的数据。

```
aws networkflowmonitor get-query-results-workload-insights-top-contributors-data \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id cc4f4ab3-3103-33b8-80ff-d6597a0c6cea
```

输出：

```
{
    "datapoints": [
        {
            "timestamps": [
                "2024-12-09T19:00:00+00:00",
                "2024-12-09T19:05:00+00:00",
                "2024-12-09T19:10:00+00:00"
            ],
            "values": [
                259943.0,
                194856.0,
                216432.0
            ],
            "label": "use1-az6"
        }
    ],
    "unit": "Bytes"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetQueryResultsWorkloadInsightsTopContributorsData。

get-query-results-workload-insights-top-contributors

以下代码示例演示了如何使用 get-query-results-workload-insights-top-contributors。

Amazon CLI

检索有关工作负载见解的排名靠前的贡献者

以下 get-query-results-workload-insights-top-contributors 示例返回指定查询的数据。

```
aws networkflowmonitor get-query-results-workload-insights-top-contributors \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id 1fc423d3-b144-37a6-80e6-e2c7d26eea0c
```

输出：

```
{
    "topContributors": [
        {
            "accountId": "123456789012",
            "localSubnetId": "subnet-0a5b30fb95dca2c14",
            "localAz": "use1-az6",
            "localVpcId": "vpc-03ea55eeda25adbb0",
            "localRegion": "us-east-1",
            "remoteIdentifier": "",
            "value": 908443,
            "localSubnetArn": "arn:aws:ec2:us-east-1:123456789012:subnet/
subnet-0a5b30fb95dca2c14",
            "localVpcArn": "arn:aws:ec2:us-east-1:123456789012:vpc/
vpc-03ea55eeda25adbb0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetQueryResultsWorkloadInsightsTopContributors。

get-query-status-monitor-top-contributors

以下代码示例演示了如何使用 get-query-status-monitor-top-contributors。

Amazon CLI

检索查询的状态

以下 get-query-status-monitor-top-contributors 示例显示指定账户中查询的当前状态。

```
aws networkflowmonitor get-query-status-monitor-top-contributors \
    --monitor-name Demo \
    --query-id 5398eabd-bc40-3f5f-aba3-bcb639d3c7ca
```

输出：

```
{
    "status": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetQueryStatusMonitorTopContributors。

get-query-status-workload-insights-top-contributors-data

以下代码示例演示了如何使用 get-query-status-workload-insights-top-contributors-data。

Amazon CLI

检索查询的状态

以下 get-query-status-workload-insights-top-contributors-data 示例显示指定账户中查询的当前状态。

```
aws networkflowmonitor get-query-status-workload-insights-top-contributors-data \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id 4333754d-8ae1-3f29-b6b7-c36db2e7f8ac
```

输出：

```
{
    "status": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetQueryStatusWorkloadInsightsTopContributorsData。

get-query-status-workload-insights-top-contributors

以下代码示例演示了如何使用 get-query-status-workload-insights-top-contributors。

Amazon CLI

检索查询的状态

以下 get-query-status-workload-insights-top-contributors 示例显示指定账户中查询的当前状态。

```
aws networkflowmonitor get-query-status-workload-insights-top-contributors \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id f2a87c70-3e5a-362e-8beb-4747d13d8419
```

输出：

```
{
    "status": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetQueryStatusWorkloadInsightsTopContributors。

get-scope

以下代码示例演示了如何使用 get-scope。

Amazon CLI

检索有关范围的信息

以下 get-scope 示例显示有关范围的信息，例如状态、标签、名称和目标详细信息。

```
aws networkflowmonitor get-scope \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf
```

输出：

```
{
    "scopeId": "e21cda79-30a0-4c12-9299-d8629d76d8cf",
    "status": "SUCCEEDED",
    "scopeArn": "arn:aws:networkflowmonitor:us-east-1:123456789012:scope/
e21cda79-30a0-4c12-9299-d8629d76d8cf",
    "targets": [
        {
            "targetIdentifier": {
                "targetId": {
                    "accountId": "123456789012"
                },
                "targetType": "ACCOUNT"
            },
            "region": "us-east-1"
        }
    ],
    "tags": {}
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetScope。

list-monitors

以下代码示例演示了如何使用 list-monitors。

Amazon CLI

检索监测仪列表

以下 list-monitors 示例返回指定账户中的所有监测仪。

```
aws networkflowmonitor list-monitors
```

输出：

```
{
    "monitors": [
        {
            "monitorArn": "arn:aws:networkflowmonitor:us-
east-1:123456789012:monitor/Demo",
            "monitorName": "Demo",
            "monitorStatus": "ACTIVE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMonitors。

list-scopes

以下代码示例演示了如何使用 list-scopes。

Amazon CLI

检索范围列表

以下 list-scopes 示例列出指定账户中的所有范围。

```
aws networkflowmonitor list-scopes
```

输出：

```
{
    "scopes": [
        {
            "scopeId": "fdc20616-6bb4-4242-a24e-a748e65ca7ac",
```

```
            "status": "SUCCEEDED",
            "scopeArn": "arn:aws:networkflowmonitor:us-east-1:123456789012:scope/
fdc20616-6bb4-4242-a24e-a748e65ca7ac"
        }
    ]
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListScopes。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出标签

以下 `list-tags-for-resource` 示例返回所有与指定资源关联的标签。

```
aws networkflowmonitor list-tags-for-resource \
    --resource-arn arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/Demo
```

输出：

```
{
    "tags": {
        "Value": "Production",
        "Key": "stack"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的标记 Amazon CloudWatch 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`start-query-monitor-top-contributors`

以下代码示例演示了如何使用 `start-query-monitor-top-contributors`。

Amazon CLI

开始查询

以下 start-query-monitor-top-contributors 示例开始查询，该查询返回查询 ID 以检索排名靠前的贡献者。

```
aws networkflowmonitor start-query-monitor-top-contributors \
    --monitor-name Demo \
    --start-time 2024-12-09T19:00:00Z \
    --end-time 2024-12-09T19:15:00Z \
    --metric-name DATA_TRANSFERRED \
    --destination-category UNCLASSIFIED
```

输出：

```
{
    "queryId": "aecd3a88-0283-35b0-a17d-6e944dc8531d"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartQueryMonitorTopContributors。

start-query-workload-insights-top-contributors-data

以下代码示例演示了如何使用 start-query-workload-insights-top-contributors-data。

Amazon CLI

开始查询

以下 start-query-workload-insights-top-contributors-data 示例开始查询，该查询返回查询 ID 以检索排名靠前的贡献者。

```
aws networkflowmonitor start-query-workload-insights-top-contributors-data \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --start-time 2024-12-09T19:00:00Z \
    --end-time 2024-12-09T19:15:00Z \
    --metric-name DATA_TRANSFERRED \
```

```
    --destination-category UNCLASSIFIED
```

输出：

```
{
    "queryId": "cc4f4ab3-3103-33b8-80ff-d6597a0c6cea"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartQueryWorkloadInsightsTopContributorsData。

start-query-workload-insights-top-contributors

以下代码示例演示了如何使用 start-query-workload-insights-top-contributors。

Amazon CLI

开始查询

以下 start-query-workload-insights-top-contributors 示例开始查询，该查询返回查询 ID 以检索排名靠前的贡献者。

```
aws networkflowmonitor start-query-workload-insights-top-contributors \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --start-time 2024-12-09T19:00:00Z \
    --end-time 2024-12-09T19:15:00Z \
    --metric-name DATA_TRANSFERRED \
    --destination-category UNCLASSIFIED
```

输出：

```
{
    "queryId": "1fc423d3-b144-37a6-80e6-e2c7d26eea0c"
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartQueryWorkloadInsightsTopContributors。

stop-query-monitor-top-contributors

以下代码示例演示了如何使用 stop-query-monitor-top-contributors。

Amazon CLI

停止查询

以下 stop-query-monitor-top-contributors 示例停止指定账户中的查询。

```
aws networkflowmonitor stop-query-monitor-top-contributors \
    --monitor-name Demo \
    --query-id aecd3a88-0283-35b0-a17d-6e944dc8531d
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopQueryMonitorTopContributors。

stop-query-workload-insights-top-contributors-data

以下代码示例演示了如何使用 stop-query-workload-insights-top-contributors-data。

Amazon CLI

停止查询

以下 stop-query-workload-insights-top-contributors-data 示例停止指定账户中的查询。

```
aws networkflowmonitor stop-query-workload-insights-top-contributors-data \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id cc4f4ab3-3103-33b8-80ff-d6597a0c6cea
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopQueryWorkloadInsightsTopContributorsData。

stop-query-workload-insights-top-contributors

以下代码示例演示了如何使用 stop-query-workload-insights-top-contributors。

Amazon CLI

停止查询

以下 stop-query-workload-insights-top-contributors 示例停止指定账户中的查询。

```
aws networkflowmonitor stop-query-workload-insights-top-contributors \
    --scope-id e21cda79-30a0-4c12-9299-d8629d76d8cf \
    --query-id 1fc423d3-b144-37a6-80e6-e2c7d26eea0c
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的利用工作负载见解评估网络流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopQueryWorkloadInsightsTopContributors。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为指定资源添加标签

以下 tag-resource 示例为指定账户中的监测仪添加标签。

```
aws networkflowmonitor tag-resource \
    --resource-arn arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/Demo \
    --tags Key=stack,Value=Production
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的标记 Amazon CloudWatch 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

移除指定资源的标签

以下 untag-resource 示例移除指定账户中的监测仪的标签。

```
aws networkflowmonitor untag-resource \
    --resource-arn arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/Demo \
    --tag-keys stack
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的标记 Amazon CloudWatch 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-monitor

以下代码示例演示了如何使用 update-monitor。

Amazon CLI

更新现有监测仪

以下 update-monitor 示例更新指定账户中名为 Demo 的监测仪。

```
aws networkflowmonitor update-monitor \
    --monitor-name Demo \
    --local-resources-to-add type="AWS::EC2::VPC",identifier="arn:aws:ec2:us-
east-1:123456789012:vpc/vpc-048d08dfbec623f94"
```

输出：

```
{
    "monitorArn": "arn:aws:networkflowmonitor:us-east-1:123456789012:monitor/Demo",
    "monitorName": "Demo",
    "monitorStatus": "ACTIVE",
    "tags": {
        "Value": "Production",
```

```
        "Key": "stack"
    }
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Network Flow Monitor 的组件和功能。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMonitor。

使用 Amazon CLI 的 Network Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Network Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-customer-gateway`

以下代码示例演示了如何使用 `associate-customer-gateway`。

Amazon CLI

关联客户网关

以下 `associate-customer-gateway` 示例将指定全局网络中的客户网关 `cgw-11223344556677889` 与设备 `device-07f6fd08867abc123` 相关联。

```
aws networkmanager associate-customer-gateway \
    --customer-gateway-arn arn:aws:ec2:us-west-2:123456789012:customer-gateway/
cgw-11223344556677889  \
    --global-network-id global-network-01231231231231231 \
    --device-id device-07f6fd08867abc123 \
```

```
    --region us-west-2
```

输出：

```
{
    "CustomerGatewayAssociation": {
        "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-gateway/
cgw-11223344556677889",
        "GlobalNetworkId": "global-network-01231231231231231",
        "DeviceId": "device-07f6fd08867abc123",
        "State": "PENDING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的客户网关关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateCustomerGateway。

associate-link

以下代码示例演示了如何使用 associate-link。

Amazon CLI

关联链接

以下 associate-link 示例将链接 link-11112222aaaabbbb1 与设备 device-07f6fd08867abc123 关联。该链接和设备位于指定全局网络中。

```
aws networkmanager associate-link \
    --global-network-id global-network-01231231231231231 \
    --device-id device-07f6fd08867abc123 \
    --link-id link-11112222aaaabbbb1 \
    --region us-west-2
```

输出：

```
{
    "LinkAssociation": {
        "GlobalNetworkId": "global-network-01231231231231231",
        "DeviceId": "device-07f6fd08867abc123",
        "LinkId": "link-11112222aaaabbbb1",
```

```
        "LinkAssociationState": "PENDING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的设备和链接关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateLink。

`create-core-network`

以下代码示例演示了如何使用 `create-core-network`。

Amazon CLI

创建核心网络

以下 `create-core-network` 示例使用可选描述和标签在 Amazon Cloud WAN 全局网络中创建一个核心网络。

```
aws networkmanager create-core-network \
    --global-network-id global-network-cdef-EXAMPLE22222 \
    --description "Main headquarters location" \
    --tags Key=Name,Value="New York City office"
```

输出：

```
{
    "CoreNetwork": {
        "GlobalNetworkId": "global-network-cdef-EXAMPLE22222",
        "CoreNetworkId": "core-network-cdef-EXAMPLE33333",
        "CoreNetworkArn": "arn:aws:networkmanager::987654321012:core-network/core-
network-cdef-EXAMPLE33333",
        "Description": "Main headquarters location",
        "CreatedAt": "2022-01-10T19:53:59+00:00",
        "State": "AVAILABLE",
        "Tags": [
            {
                "Key": "Name",
                "Value": "New York City office"
            }
        ]
    }
```

```
}
```

有关更多信息，请参阅《Amazon Cloud WAN 用户指南》中的全局和核心网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCoreNetwork。

create-device

以下代码示例演示了如何使用 create-device。

Amazon CLI

创建设备

以下 create-device 示例在指定全局网络中创建一个设备。该设备的详细信息包括描述、类型、供应商、型号和序列号。

```
aws networkmanager create-device
    --global-network-id global-network-01231231231231231 \
    --description "New York office device" \
    --type "office device" \
    --vendor "anycompany" \
    --model "abcabc" \
    --serial-number "1234" \
    --region us-west-2
```

输出：

```
{
    "Device": {
        "DeviceId": "device-07f6fd08867abc123",
        "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "New York office device",
        "Type": "office device",
        "Vendor": "anycompany",
        "Model": "abcabc",
        "SerialNumber": "1234",
        "CreatedAt": 1575554005.0,
        "State": "PENDING"
    }
```

```
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDevice。

create-global-network

以下代码示例演示了如何使用 create-global-network。

Amazon CLI

创建全局网络

以下 create-global-network 示例创建一个新的全局网络。创建时的初始状态为 PENDING。

```
aws networkmanager create-global-network
```

输出：

```
{
    "GlobalNetwork": {
        "GlobalNetworkId": "global-network-00a77fc0f722dae74",
        "GlobalNetworkArn": "arn:aws:networkmanager::987654321012:global-network/
global-network-00a77fc0f722dae74",
        "CreatedAt": "2022-03-14T20:31:56+00:00",
        "State": "PENDING"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGlobalNetwork。

create-link

以下代码示例演示了如何使用 create-link。

Amazon CLI

创建链接

以下 create-link 示例在指定全局网络中创建一个链接。该链接包括有关链接类型、带宽和提供商的描述和详细信息。站点 ID 表示链接所关联的站点。

```
aws networkmanager create-link \
    --global-network-id global-network-01231231231231231 \
    --description "VPN Link" \
    --type "broadband" \
    --bandwidth UploadSpeed=10,DownloadSpeed=20 \
    --provider "AnyCompany" \
    --site-id site-444555aaabbb11223 \
    --region us-west-2
```

输出：

```
{
    "Link": {
        "LinkId": "link-11112222aaaabbbb1",
        "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
        "GlobalNetworkId": "global-network-01231231231231231",
        "SiteId": "site-444555aaabbb11223",
        "Description": "VPN Link",
        "Type": "broadband",
        "Bandwidth": {
            "UploadSpeed": 10,
            "DownloadSpeed": 20
        },
        "Provider": "AnyCompany",
        "CreatedAt": 1575555811.0,
        "State": "PENDING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLink。

create-site

以下代码示例演示了如何使用 create-site。

Amazon CLI

创建站点

以下 create-site 示例在指定全局网络中创建一个站点。该站点的详细信息包括描述和位置信息。

```
aws networkmanager create-site \
    --global-network-id global-network-01231231231231231 \
    --description  "New York head office" \
    --location Latitude=40.7128,Longitude=-74.0060 \
    --region us-west-2
```

输出：

```
{
    "Site": {
        "SiteId": "site-444555aaabbb11223",
        "SiteArn": "arn:aws:networkmanager::123456789012:site/global-
network-01231231231231231/site-444555aaabbb11223",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "New York head office",
        "Location": {
            "Latitude": "40.7128",
            "Longitude": "-74.0060"
        },
        "CreatedAt": 1575554300.0,
        "State": "PENDING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用站点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSite。

create-vpc-attachment

以下代码示例演示了如何使用 create-vpc-attachment。

Amazon CLI

创建 VPC 挂载

以下 create-vpc-attachment 示例在核心网络中创建支持 IPv6 的 VPC 挂载。

```
aws networkmanager create-vpc-attachment \
```

```
    --core-network-id core-network-0fab62fe438d94db6 \
    --vpc-arn arn:aws:ec2:us-east-1:987654321012:vpc/vpc-09f37f69e2786eeb8  \
    --subnet-arns arn:aws:ec2:us-east-1:987654321012:subnet/subnet-04ca4e010857e7bb7
 \
    --Ipv6Support=true
```

输出：

```
{
    "VpcAttachment": {
        "Attachment": {
            "CoreNetworkId": "core-network-0fab62fe438d94db6",
            "AttachmentId": "attachment-05e1da6eba87a06e6",
            "OwnerAccountId": "987654321012",
            "AttachmentType": "VPC",
            "State": "CREATING",
            "EdgeLocation": "us-east-1",
            "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/
vpc-09f37f69e2786eeb8",
            "Tags": [],
            "CreatedAt": "2022-03-10T20:59:14+00:00",
            "UpdatedAt": "2022-03-10T20:59:14+00:00"
        },
        "SubnetArns": [
            "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-04ca4e010857e7bb7"
        ],
        "Options": {
            "Ipv6Support": true
        }
    }
}
```

有关更多信息，请参阅《Cloud WAN 用户指南》中的创建挂载。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVpcAttachment。

`delete-attachment`

以下代码示例演示了如何使用 `delete-attachment`。

Amazon CLI

删除挂载

以下 delete-attachment 示例删除一个连接挂载。

```
aws networkmanager delete-attachment \
    --attachment-id attachment-01feddaeae26ab68c
```

输出：

```
{
    "Attachment": {
        "CoreNetworkId": "core-network-0f4b0a9d5ee7761d1",
        "AttachmentId": "attachment-01feddaeae26ab68c",
        "OwnerAccountId": "987654321012",
        "AttachmentType": "CONNECT",
        "State": "DELETING",
        "EdgeLocation": "us-east-1",
        "ResourceArn": "arn:aws:networkmanager::987654321012:attachment/
attachment-02c3964448fedf5aa",
        "CreatedAt": "2022-03-15T19:18:41+00:00",
        "UpdatedAt": "2022-03-15T19:28:59+00:00"
    }
}
```

有关更多信息，请参阅《Cloud WAN 用户指南》中的删除挂载。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAttachment。

delete-bucket-analytics-configuration

以下代码示例演示了如何使用 delete-bucket-analytics-configuration。

Amazon CLI

删除存储桶的分析配置

以下 delete-bucket-analytics-configuration 示例移除指定存储桶和 ID 的分析配置。

```
aws s3api delete-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketAnalyticsConfiguration。

delete-bucket-metrics-configuration

以下代码示例演示了如何使用 delete-bucket-metrics-configuration。

Amazon CLI

删除存储桶的指标配置

以下 delete-bucket-metrics-configuration 示例移除指定存储桶和 ID 的指标配置。

```
aws s3api delete-bucket-metrics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 123
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketMetricsConfiguration。

delete-core-network

以下代码示例演示了如何使用 delete-core-network。

Amazon CLI

删除核心网络

以下 delete-core-network 示例从 Cloud WAN 全局网络中删除核心网络。

```
aws networkmanager delete-core-network \
    --core-network-id core-network-0fab62fe438d94db6
```

输出：

```
{
    "CoreNetwork": {
        "GlobalNetworkId": "global-network-0d59060f16a73bc41",
        "CoreNetworkId": "core-network-0fab62fe438d94db6",
```

```
        "Description": "Main headquarters location",
        "CreatedAt": "2021-12-09T18:31:11+00:00",
        "State": "DELETING",
        "Segments": [
            {
                "Name": "dev",
                "EdgeLocations": [
                    "us-east-1"
                ],
                "SharedSegments": []
            }
        ],
        "Edges": [
            {
                "EdgeLocation": "us-east-1",
                "Asn": 64512,
                "InsideCidrBlocks": []
            }
        ]
    }
}
```

有关更多信息，请参阅《Cloud WAN 用户指南》中的核心网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCoreNetwork。

delete-device

以下代码示例演示了如何使用 delete-device。

Amazon CLI

删除设备

以下 delete-device 示例从指定的全局网络中删除指定设备。

```
aws networkmanager delete-device \
    --global-network-id global-network-01231231231231231 \
    --device-id device-07f6fd08867abc123 \
    --region us-west-2
```

输出：

```
{
    "Device": {
        "DeviceId": "device-07f6fd08867abc123",
        "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "New York office device",
        "Type": "office device",
        "Vendor": "anycompany",
        "Model": "abcabc",
        "SerialNumber": "1234",
        "SiteId": "site-444555aaabbb11223",
        "CreatedAt": 1575554005.0,
        "State": "DELETING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDevice。

delete-global-network

以下代码示例演示了如何使用 delete-global-network。

Amazon CLI

删除全局网络

以下 delete-global-network 示例删除一个全局网络。

```
aws networkmanager delete-global-network \
    --global-network-id global-network-052bedddccb193b6b
```

输出：

```
{
    "GlobalNetwork": {
        "GlobalNetworkId": "global-network-052bedddccb193b6b",
        "GlobalNetworkArn": "arn:aws:networkmanager::987654321012:global-network/
global-network-052bedddccb193b6b",
        "CreatedAt": "2021-12-09T18:19:12+00:00",
```

```
        "State": "DELETING"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGlobalNetwork。

delete-link

以下代码示例演示了如何使用 delete-link。

Amazon CLI

删除链接

以下 delete-link 示例从指定的全局网络中删除指定的链接。

```
aws networkmanager delete-link \
    --global-network-id global-network-01231231231231231  \
    --link-id link-11112222aaaabbbb1 \
    --region us-west-2
```

输出：

```
{
    "Link": {
        "LinkId": "link-11112222aaaabbbb1",
        "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
        "GlobalNetworkId": "global-network-01231231231231231",
        "SiteId": "site-444555aaabbb11223",
        "Description": "VPN Link",
        "Type": "broadband",
        "Bandwidth": {
            "UploadSpeed": 20,
            "DownloadSpeed": 20
        },
        "Provider": "AnyCompany",
        "CreatedAt": 1575555811.0,
        "State": "DELETING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLink。

delete-public-access-block

以下代码示例演示了如何使用 delete-public-access-block。

Amazon CLI

删除存储桶的屏蔽公共访问权限配置

以下 delete-public-access-block 示例移除指定存储桶上的屏蔽公共访问权限配置。

```
aws s3api delete-public-access-block \
    --bucket amzn-s3-demo-bucket
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePublicAccessBlock。

delete-site

以下代码示例演示了如何使用 delete-site。

Amazon CLI

删除站点

以下 delete-site 示例删除指定全局网络中的指定站点（site-444555aaabbb11223）。

```
aws networkmanager delete-site \
    --global-network-id global-network-01231231231231231  \
    --site-id site-444555aaabbb11223 \
    --region us-west-2
```

输出：

```
{
    "Site": {
        "SiteId": "site-444555aaabbb11223",
```

```
        "SiteArn": "arn:aws:networkmanager::123456789012:site/global-
network-01231231231231231/site-444555aaabbb11223",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "New York head office",
        "Location": {
            "Latitude": "40.7128",
            "Longitude": "-74.0060"
        },
        "CreatedAt": 1575554300.0,
        "State": "DELETING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用站点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSite。

deregister-transit-gateway

以下代码示例演示了如何使用 deregister-transit-gateway。

Amazon CLI

从全局网络中注销中转网关

以下 deregister-transit-gateway 示例从指定的全局网络注销指定的中转网关。

```
aws networkmanager deregister-transit-gateway \
    --global-network-id global-network-01231231231231231 \
    --transit-gateway-arn arn:aws:ec2:us-west-2:123456789012:transit-gateway/
tgw-123abc05e04123abc \
    --region us-west-2
```

输出：

```
{
    "TransitGatewayRegistration": {
        "GlobalNetworkId": "global-network-01231231231231231",
        "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-gateway/
tgw-123abc05e04123abc",
        "State": {
            "Code": "DELETING"
```

```
        }
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的中转网关注册。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTransitGateway。

describe-global-networks

以下代码示例演示了如何使用 describe-global-networks。

Amazon CLI

描述您的全局网络

以下 describe-global-networks 示例描述您的账户中的所有全局网络。

```
aws networkmanager describe-global-networks \
    --region us-west-2
```

输出：

```
{
    "GlobalNetworks": [
        {
            "GlobalNetworkId": "global-network-01231231231231231",
            "GlobalNetworkArn": "arn:aws:networkmanager::123456789012:global-
network/global-network-01231231231231231",
            "Description": "Company 1 global network",
            "CreatedAt": 1575553525.0,
            "State": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGlobalNetworks。

disassociate-customer-gateway

以下代码示例演示了如何使用 disassociate-customer-gateway。

Amazon CLI

取消关联客户网关

以下 disassociate-customer-gateway 示例取消指定客户网关（cgw-11223344556677889）与指定全局网络的关联。

```
aws networkmanager disassociate-customer-gateway \
    --global-network-id global-network-01231231231231231 \
    --customer-gateway-arn arn:aws:ec2:us-west-2:123456789012:customer-gateway/
cgw-11223344556677889 \
    --region us-west-2
```

输出：

```
{
    "CustomerGatewayAssociation": {
        "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-gateway/
cgw-11223344556677889",
        "GlobalNetworkId": "global-network-01231231231231231",
        "DeviceId": "device-07f6fd08867abc123",
        "State": "DELETING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的客户网关关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateCustomerGateway。

disassociate-link

以下代码示例演示了如何使用 disassociate-link。

Amazon CLI

取消关联链接

以下 disassociate-link 示例取消指定的全局网络中指定链接与设备 device-07f6fd08867abc123 的关联。

```
aws networkmanager disassociate-link \
    --global-network-id global-network-01231231231231231 \
    --device-id device-07f6fd08867abc123 \
```

```
    --link-id link-11112222aaaabbbb1 \
    --region us-west-2
```

输出：

```
{
    "LinkAssociation": {
        "GlobalNetworkId": "global-network-01231231231231231",
        "DeviceId": "device-07f6fd08867abc123",
        "LinkId": "link-11112222aaaabbbb1",
        "LinkAssociationState": "DELETING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的设备和链接关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateLink。

get-bucket-analytics-configuration

以下代码示例演示了如何使用 get-bucket-analytics-configuration。

Amazon CLI

检索具有特定 ID 的存储桶的分析配置

以下 get-bucket-analytics-configuration 示例显示了指定存储桶和 ID 的分析配置。

```
aws s3api get-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

输出：

```
{
    "AnalyticsConfiguration": {
        "StorageClassAnalysis": {},
        "Id": "1"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketAnalyticsConfiguration。

get-bucket-metrics-configuration

以下代码示例演示了如何使用 get-bucket-metrics-configuration。

Amazon CLI

检索具有特定 ID 的存储桶的指标配置

以下 get-bucket-metrics-configuration 示例显示了指定存储桶和 ID 的指标配置。

```
aws s3api get-bucket-metrics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 123
```

输出：

```
{
    "MetricsConfiguration": {
        "Filter": {
            "Prefix": "logs"
        },
        "Id": "123"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketMetricsConfiguration。

get-customer-gateway-associations

以下代码示例演示了如何使用 get-customer-gateway-associations。

Amazon CLI

获取您的客户网关关联

以下 get-customer-gateway-associations 示例获取指定全局网络的客户网关关联。

```
aws networkmanager get-customer-gateway-associations \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "CustomerGatewayAssociations": [
        {
            "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-
gateway/cgw-11223344556677889",
            "GlobalNetworkId": "global-network-01231231231231231",
            "DeviceId": "device-07f6fd08867abc123",
            "State": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCustomerGatewayAssociations。

get-devices

以下代码示例演示了如何使用 get-devices。

Amazon CLI

获取您的设备

以下 get-devices 示例获取指定全局网络中的设备。

```
aws networkmanager get-devices \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "Devices": [
        {
            "DeviceId": "device-07f6fd08867abc123",
            "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123",
            "GlobalNetworkId": "global-network-01231231231231231",
            "Description": "NY office device",
            "Type": "office device",
            "Vendor": "anycompany",
            "Model": "abcabc",
```

```
            "SerialNumber": "1234",
            "CreatedAt": 1575554005.0,
            "State": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDevices。

get-link-associations

以下代码示例演示了如何使用 get-link-associations。

Amazon CLI

获取您的链接关联

以下 get-link-associations 示例获取指定全局网络中的链接关联。

```
aws networkmanager get-link-associations \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "LinkAssociations": [
        {
            "GlobalNetworkId": "global-network-01231231231231231",
            "DeviceId": "device-07f6fd08867abc123",
            "LinkId": "link-11112222aaaabbbb1",
            "LinkAssociationState": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLinkAssociations。

get-links

以下代码示例演示了如何使用 get-links。

Amazon CLI

获取您的链接

以下 get-links 示例获取指定全局网络中的链接。

```
aws networkmanager get-links \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "Links": [
        {
            "LinkId": "link-11112222aaaabbbb1",
            "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
            "GlobalNetworkId": "global-network-01231231231231231",
            "SiteId": "site-444555aaabbb11223",
            "Description": "VPN Link",
            "Type": "broadband",
            "Bandwidth": {
                "UploadSpeed": 10,
                "DownloadSpeed": 20
            },
            "Provider": "AnyCompany",
            "CreatedAt": 1575555811.0,
            "State": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLinks。

get-object-retention

以下代码示例演示了如何使用 get-object-retention。

Amazon CLI

检索对象的对象保留配置

以下 get-object-retention 示例检索指定对象的对象保留配置。

```
aws s3api get-object-retention \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf
```

输出：

```
{
    "Retention": {
        "Mode": "GOVERNANCE",
        "RetainUntilDate": "2025-01-01T00:00:00.000Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectRetention。

get-public-access-block

以下代码示例演示了如何使用 get-public-access-block。

Amazon CLI

设置或修改存储桶的屏蔽公共访问权限配置

以下 get-public-access-block 示例显示了指定存储桶的屏蔽公共访问权限配置。

```
aws s3api get-public-access-block --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "PublicAccessBlockConfiguration": {
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "BlockPublicAcls": true,
        "RestrictPublicBuckets": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicAccessBlock。

get-sites

以下代码示例演示了如何使用 get-sites。

Amazon CLI

获取您的站点

以下 get-sites 示例获取指定全局网络中的站点。

```
aws networkmanager get-sites \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "Sites": [
        {
            "SiteId": "site-444555aaabbb11223",
            "SiteArn": "arn:aws:networkmanager::123456789012:site/global-
network-01231231231231231/site-444555aaabbb11223",
            "GlobalNetworkId": "global-network-01231231231231231",
            "Description": "NY head office",
            "Location": {
                "Latitude": "40.7128",
                "Longitude": "-74.0060"
            },
            "CreatedAt": 1575554528.0,
            "State": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSites。

get-transit-gateway-registrations

以下代码示例演示了如何使用 get-transit-gateway-registrations。

Amazon CLI

获取中转网关注册

以下 get-transit-gateway-registrations 示例获取注册到指定全局网络的中转网关。

```
aws networkmanager get-transit-gateway-registrations \
    --global-network-id global-network-01231231231231231 \
    --region us-west-2
```

输出：

```
{
    "TransitGatewayRegistrations": [
        {
            "GlobalNetworkId": "global-network-01231231231231231",
            "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-
gateway/tgw-123abc05e04123abc",
            "State": {
                "Code": "AVAILABLE"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTransitGatewayRegistrations。

get-vpc-attachment

以下代码示例演示了如何使用 get-vpc-attachment。

Amazon CLI

获取 VPC 挂载

以下 get-vpc-attachment 示例返回有关 VPC 挂载的信息。

```
aws networkmanager get-vpc-attachment \
    --attachment-id  attachment-03b7ea450134787da
```

输出：

```
{
    "VpcAttachment": {
        "Attachment": {
```

```
            "CoreNetworkId": "core-network-0522de1b226a5d7b3",
            "AttachmentId": "attachment-03b7ea450134787da",
            "OwnerAccountId": "987654321012",
            "AttachmentType": "VPC",
            "State": "CREATING",
            "EdgeLocation": "us-east-1",
            "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/vpc-a7c4bbda",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "DevVPC"
                }
            ],
            "CreatedAt": "2022-03-11T17:48:58+00:00",
            "UpdatedAt": "2022-03-11T17:48:58+00:00"
        },
        "SubnetArns": [
            "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-202cde6c",
            "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-e5022dba",
            "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-2387ae02",
            "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-cda9dffc"
        ],
        "Options": {
            "Ipv6Support": false
        }
    }
}
```

有关更多信息，请参阅《Cloud WAN 用户指南》中的挂载。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVpcAttachment。

list-bucket-analytics-configurations

以下代码示例演示了如何使用 list-bucket-analytics-configurations。

Amazon CLI

检索存储桶的分析配置列表

下面的 list-bucket-analytics-configurations 检索指定存储桶的分析配置列表。

```
aws s3api list-bucket-analytics-configurations \
```

```
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "AnalyticsConfigurationList": [
        {
            "StorageClassAnalysis": {},
            "Id": "1"
        }
    ],
    "IsTruncated": false
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketAnalyticsConfigurations。

list-bucket-metrics-configurations

以下代码示例演示了如何使用 list-bucket-metrics-configurations。

Amazon CLI

检索存储桶的指标配置列表

以下 list-bucket-metrics-configurations 示例检索指定存储桶的指标配置列表。

```
aws s3api list-bucket-metrics-configurations \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "IsTruncated": false,
    "MetricsConfigurationList": [
        {
            "Filter": {
                "Prefix": "logs"
            },
            "Id": "123"
        },
        {
            "Filter": {
```

```
                "Prefix": "tmp"
            },
            "Id": "234"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketMetricsConfigurations。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 list-tags-for-resource 示例列出指定设备资源（device-07f6fd08867abc123）的标签。

```
aws networkmanager list-tags-for-resource \
    --resource-arn arn:aws:networkmanager::123456789012:device/global-network-01231231231231231/device-07f6fd08867abc123 \
    --region us-west-2
```

输出：

```
{
    "TagList": [
        {
            "Key": "Network",
            "Value": "Northeast"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-bucket-metrics-configuration

以下代码示例演示了如何使用 put-bucket-metrics-configuration。

Amazon CLI

设置存储桶的指标配置

以下 put-bucket-metrics-configuration 示例为指定存储桶设置 ID 为 123 的指标配置。

```
aws s3api put-bucket-metrics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 123 \
    --metrics-configuration '{"Id": "123", "Filter": {"Prefix": "logs"}}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketMetricsConfiguration。

put-object-retention

以下代码示例演示了如何使用 put-object-retention。

Amazon CLI

为对象设置对象保留配置

以下 put-object-retention 示例为指定对象设置直到 2025 年 1 月 1 日的对象保留配置。

```
aws s3api put-object-retention \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf \
    --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectRetention。

put-public-access-block

以下代码示例演示了如何使用 put-public-access-block。

Amazon CLI

为存储桶设置阻止公共访问配置

以下 put-public-access-block 示例为指定存储桶设置限制性阻止公共访问配置。

```
aws s3api put-public-access-block \
    --bucket amzn-s3-demo-bucket \
    --public-access-block-
configuration "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPub
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPublicAccessBlock。

register-transit-gateway

以下代码示例演示了如何使用 register-transit-gateway。

Amazon CLI

在全局网络中注册中转网关

以下 register-transit-gateway 示例在指定全局网络中注册中转网关 tgw-123abc05e04123abc。

```
aws networkmanager register-transit-gateway \
    --global-network-id global-network-01231231231231231 \
    --transit-gateway-arn arn:aws:ec2:us-west-2:123456789012:transit-gateway/
tgw-123abc05e04123abc \
    --region us-west-2
```

输出：

```
{
    "TransitGatewayRegistration": {
        "GlobalNetworkId": "global-network-01231231231231231",
        "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-gateway/
tgw-123abc05e04123abc",
        "State": {
            "Code": "PENDING"
        }
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的中转网关注册。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTransitGateway。

`reject-attachment`

以下代码示例演示了如何使用 `reject-attachment`。

Amazon CLI

拒绝挂载

以下 `reject-attachment` 示例拒绝 VPC 挂载请求。

```
aws networkmanager reject-attachment \
    --attachment-id  attachment-03b7ea450134787da
```

输出：

```
{
    "Attachment": {
        "CoreNetworkId": "core-network-0522de1b226a5d7b3",
        "AttachmentId": "attachment-03b7ea450134787da",
        "OwnerAccountId": "987654321012",
        "AttachmentType": "VPC",
        "State": "AVAILABLE",
        "EdgeLocation": "us-east-1",
        "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/vpc-a7c4bbda",
        "CreatedAt": "2022-03-11T17:48:58+00:00",
        "UpdatedAt": "2022-03-11T17:51:25+00:00"
    }
}
```

有关更多信息，请参阅《Cloud WAN 用户指南》中的接受挂载。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectAttachment。

`start-route-analysis`

以下代码示例演示了如何使用 `start-route-analysis`。

Amazon CLI

启动路由分析

以下 start-route-analysis 示例启动源和目标之间的分析，包括可选的 include-return-path。

```
aws networkmanager start-route-analysis \
    --global-network-id global-network-00aa0aaa0b0aaa000 \
    --source TransitGatewayAttachmentArn=arn:aws:ec2:us-east-1:503089527312:transit-gateway-attachment/tgw-attach-0d4a2d491bf68c093,IpAddress=10.0.0.0 \
    --destination TransitGatewayAttachmentArn=arn:aws:ec2:us-west-1:503089527312:transit-gateway-attachment/tgw-attach-002577f30bb181742,IpAddress=11.0.0.0 \
    --include-return-path
```

输出：

```
{
    "RouteAnalysis": {
        "GlobalNetworkId": "global-network-00aa0aaa0b0aaa000
        "OwnerAccountId": "1111222233333",
        "RouteAnalysisId": "a1873de1-273c-470c-1a2bc2345678",
        "StartTimestamp": 1695760154.0,
        "Status": "RUNNING",
        "Source": {
            "TransitGatewayAttachmentArn": "arn:aws:ec2:us-
east-1:111122223333:transit-gateway-attachment/tgw-attach-1234567890abcdef0,
            "TransitGatewayArn": "arn:aws:ec2:us-east-1:111122223333:transit-
gateway/tgw-abcdef01234567890",
            "IpAddress": "10.0.0.0"
        },
        "Destination": {
            "TransitGatewayAttachmentArn": "arn:aws:ec2:us-
west-1:555555555555:transit-gateway-attachment/tgw-attach-021345abcdef6789",
            "TransitGatewayArn": "arn:aws:ec2:us-west-1:111122223333:transit-
gateway/tgw-09876543210fedcba0",
            "IpAddress": "11.0.0.0"
        },
        "IncludeReturnPath": true,
        "UseMiddleboxes": false
    }
}
```

有关更多信息，请参阅《Amazon Global Networks for Transit Gateways 用户指南》中的路由分析器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartRouteAnalysis。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签应用于资源

以下 tag-resource 示例将标签 Network=Northeast 应用于设备 device-07f6fd08867abc123。

```
aws networkmanager tag-resource \
    --resource-arn arn:aws:networkmanager::123456789012:device/global-network-01231231231231231/device-07f6fd08867abc123 \
    --tags Key=Network,Value=Northeast \
    --region us-west-2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签

以下 untag-resource 示例从设备 device-07f6fd08867abc123 移除键为 Network 的标签。

```
aws networkmanager untag-resource \
    --resource-arn arn:aws:networkmanager::123456789012:device/global-network-01231231231231231/device-07f6fd08867abc123 ]
    --tag-keys Network \
    --region us-west-2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-device

以下代码示例演示了如何使用 update-device。

Amazon CLI

更新设备

以下 update-device 示例通过为设备 device-07f6fd08867abc123 指定站点 ID 来更新设备。

```
aws networkmanager update-device \
    --global-network-id global-network-01231231231231231 \
    --device-id device-07f6fd08867abc123 \
    --site-id site-444555aaabbb11223 \
    --region us-west-2
```

输出：

```
{
    "Device": {
        "DeviceId": "device-07f6fd08867abc123",
        "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "NY office device",
        "Type": "Office device",
        "Vendor": "anycompany",
        "Model": "abcabc",
        "SerialNumber": "1234",
        "SiteId": "site-444555aaabbb11223",
        "CreatedAt": 1575554005.0,
        "State": "UPDATING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用设备。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDevice。

update-global-network

以下代码示例演示了如何使用 update-global-network。

Amazon CLI

更新全局网络

以下 update-global-network 示例更新全局网络 global-network-01231231231231231 的描述。

```
aws networkmanager update-global-network \
    --global-network-id global-network-01231231231231231 \
    --description "Head offices" \
    --region us-west-2
```

输出：

```
{
    "GlobalNetwork": {
        "GlobalNetworkId": "global-network-01231231231231231",
        "GlobalNetworkArn": "arn:aws:networkmanager::123456789012:global-network/
global-network-01231231231231231",
        "Description": "Head offices",
        "CreatedAt": 1575553525.0,
        "State": "UPDATING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的全局网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGlobalNetwork。

update-link

以下代码示例演示了如何使用 update-link。

Amazon CLI

更新链接

以下 update-link 示例更新链接 link-11112222aaaabbbb1 的带宽信息。

```
aws networkmanager update-link \
    --global-network-id global-network-01231231231231231 \
    --link-id link-11112222aaaabbbb1 \
    --bandwidth  UploadSpeed=20,DownloadSpeed=20 \
```

```
    --region us-west-2
```

输出：

```
{
    "Link": {
        "LinkId": "link-11112222aaaabbbb1",
        "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
        "GlobalNetworkId": "global-network-01231231231231231",
        "SiteId": "site-444555aaabbb11223",
        "Description": "VPN Link",
        "Type": "broadband",
        "Bandwidth": {
            "UploadSpeed": 20,
            "DownloadSpeed": 20
        },
        "Provider": "AnyCompany",
        "CreatedAt": 1575555811.0,
        "State": "UPDATING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用链接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateLink。

update-site

以下代码示例演示了如何使用 update-site。

Amazon CLI

更新站点

以下 update-site 示例更新指定全局网络中站点 site-444555aaabbb11223 的描述。

```
aws networkmanager update-site \
    --global-network-id global-network-01231231231231231 \
    --site-id site-444555aaabbb11223 \
    --description "New York Office site" \
    --region us-west-2
```

输出：

```
{
    "Site": {
        "SiteId": "site-444555aaabbb11223",
        "SiteArn": "arn:aws:networkmanager::123456789012:site/global-
network-01231231231231231/site-444555aaabbb11223",
        "GlobalNetworkId": "global-network-01231231231231231",
        "Description": "New York Office site",
        "Location": {
            "Latitude": "40.7128",
            "Longitude": "-74.0060"
        },
        "CreatedAt": 1575554528.0,
        "State": "UPDATING"
    }
}
```

有关更多信息，请参阅《Transit Gateway Network Manager 指南》中的使用站点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSite。

使用 Amazon CLI 的 OpenSearch Service 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 OpenSearch Service 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-elasticsearch-domain`

以下代码示例演示了如何使用 `create-elasticsearch-domain`。

Amazon CLI

创建 Amazon Elasticsearch Service 域

以下 `create-elasticsearch-domain` 命令在 VPC 中创建新的 Amazon Elasticsearch Service 域，并限制对单个用户的访问。Amazon ES 从指定的子网和安全组 ID 推断 VPC ID。

```
aws es create-elasticsearch-domain \
    --domain-name vpc-cli-example \
    --elasticsearch-version 6.2 \
    --elasticsearch-cluster-
config InstanceType=m4.large.elasticsearch,InstanceCount=1 \
    --ebs-options EBSEnabled=true,VolumeType=standard,VolumeSize=10 \
    --access-policies '{"Version":"2012-10-17",        "Statement": [ { "Effect":
 "Allow", "Principal": {"AWS": "arn:aws:iam::123456789012:root" }, "Action":"es:*",
 "Resource": "arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example/*" } ] }' \
    --vpc-options SubnetIds=subnet-1a2a3a4a,SecurityGroupIds=sg-2a3a4a5a
```

输出：

```
{
    "DomainStatus": {
        "ElasticsearchClusterConfig": {
            "DedicatedMasterEnabled": false,
            "InstanceCount": 1,
            "ZoneAwarenessEnabled": false,
            "InstanceType": "m4.large.elasticsearch"
        },
        "DomainId": "123456789012/vpc-cli-example",
        "CognitoOptions": {
            "Enabled": false
        },
        "VPCOptions": {
            "SubnetIds": [
                "subnet-1a2a3a4a"
            ],
            "VPCId": "vpc-3a4a5a6a",
            "SecurityGroupIds": [
                "sg-2a3a4a5a"
            ],
            "AvailabilityZones": [
                "us-west-1c"
            ]
```

```
        },
        "Created": true,
        "Deleted": false,
        "EBSOptions": {
            "VolumeSize": 10,
            "VolumeType": "standard",
            "EBSEnabled": true
        },
        "Processing": true,
        "DomainName": "vpc-cli-example",
        "SnapshotOptions": {
            "AutomatedSnapshotStartHour": 0
        },
        "ElasticsearchVersion": "6.2",
        "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:root\"},\"Action\":
\"es:*\",\"Resource\":\"arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example/*
\"}]}",
        "AdvancedOptions": {
            "rest.action.multi.allow_explicit_index": "true"
        },
        "EncryptionAtRestOptions": {
            "Enabled": false
        },
        "ARN": "arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example"
    }
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的创建和管理 Amazon Elasticsearch Service 域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateElasticsearchDomain。

describe-elasticsearch-domain-config

以下代码示例演示了如何使用 describe-elasticsearch-domain-config。

Amazon CLI

获取域配置详细信息

以下 describe-elasticsearch-domain-config 示例提供给定域的配置详细信息，以及每个域组成部分的状态信息。

```
aws es describe-elasticsearch-domain-config \
    --domain-name cli-example
```

输出：

```
{
    "DomainConfig": {
        "ElasticsearchVersion": {
            "Options": "7.4",
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "ElasticsearchClusterConfig": {
            "Options": {
                "InstanceType": "c5.large.elasticsearch",
                "InstanceCount": 1,
                "DedicatedMasterEnabled": true,
                "ZoneAwarenessEnabled": false,
                "DedicatedMasterType": "c5.large.elasticsearch",
                "DedicatedMasterCount": 3,
                "WarmEnabled": true,
                "WarmType": "ultrawarm1.medium.elasticsearch",
                "WarmCount": 2
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "EBSOptions": {
            "Options": {
                "EBSEnabled": true,
                "VolumeType": "gp2",
                "VolumeSize": 10
            },
```

```
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "AccessPolicies": {
            "Options": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":
\"arn:aws:es:us-east-1:123456789012:domain/cli-example/*\"}]}",
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "SnapshotOptions": {
            "Options": {
                "AutomatedSnapshotStartHour": 0
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "VPCOptions": {
            "Options": {},
            "Status": {
                "CreationDate": 1591210426.162,
                "UpdateDate": 1591210426.162,
                "UpdateVersion": 18,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "CognitoOptions": {
            "Options": {
```

```
                "Enabled": false
            },
            "Status": {
                "CreationDate": 1591210426.163,
                "UpdateDate": 1591210426.163,
                "UpdateVersion": 18,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "EncryptionAtRestOptions": {
            "Options": {
                "Enabled": true,
                "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "NodeToNodeEncryptionOptions": {
            "Options": {
                "Enabled": true
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "AdvancedOptions": {
            "Options": {
                "rest.action.multi.allow_explicit_index": "true"
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
```

```
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "LogPublishingOptions": {
            "Options": {},
            "Status": {
                "CreationDate": 1591210426.164,
                "UpdateDate": 1591210426.164,
                "UpdateVersion": 18,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "DomainEndpointOptions": {
            "Options": {
                "EnforceHTTPS": true,
                "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589395827.325,
                "UpdateVersion": 8,
                "State": "Active",
                "PendingDeletion": false
            }
        },
        "AdvancedSecurityOptions": {
            "Options": {
                "Enabled": true,
                "InternalUserDatabaseEnabled": true
            },
            "Status": {
                "CreationDate": 1589395034.946,
                "UpdateDate": 1589827485.577,
                "UpdateVersion": 14,
                "State": "Active",
                "PendingDeletion": false
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的创建和管理 Amazon Elasticsearch Service 域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeElasticsearchDomainConfig。

describe-elasticsearch-domain

以下代码示例演示了如何使用 describe-elasticsearch-domain。

Amazon CLI

获取单个域的详细信息

以下 describe-elasticsearch-domain 示例提供了给定域的配置详细信息。

```
aws es describe-elasticsearch-domain \
    --domain-name cli-example
```

输出：

```
{
    "DomainStatus": {
        "DomainId": "123456789012/cli-example",
        "DomainName": "cli-example",
        "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example",
        "Created": true,
        "Deleted": false,
        "Endpoint": "search-cli-example-1a2a3a4a5a6a7a8a9a0a.us-
east-1.es.amazonaws.com",
        "Processing": false,
        "UpgradeProcessing": false,
        "ElasticsearchVersion": "7.4",
        "ElasticsearchClusterConfig": {
            "InstanceType": "c5.large.elasticsearch",
            "InstanceCount": 1,
            "DedicatedMasterEnabled": true,
            "ZoneAwarenessEnabled": false,
            "DedicatedMasterType": "c5.large.elasticsearch",
            "DedicatedMasterCount": 3,
            "WarmEnabled": true,
            "WarmType": "ultrawarm1.medium.elasticsearch",
            "WarmCount": 2
```

```
        },
        "EBSOptions": {
            "EBSEnabled": true,
            "VolumeType": "gp2",
            "VolumeSize": 10
        },
        "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":
\"arn:aws:es:us-east-1:123456789012:domain/cli-example/*\"}]}",
        "SnapshotOptions": {
            "AutomatedSnapshotStartHour": 0
        },
        "CognitoOptions": {
            "Enabled": false
        },
        "EncryptionAtRestOptions": {
            "Enabled": true,
            "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
        },
        "NodeToNodeEncryptionOptions": {
            "Enabled": true
        },
        "AdvancedOptions": {
            "rest.action.multi.allow_explicit_index": "true"
        },
        "ServiceSoftwareOptions": {
            "CurrentVersion": "R20200522",
            "NewVersion": "",
            "UpdateAvailable": false,
            "Cancellable": false,
            "UpdateStatus": "COMPLETED",
            "Description": "There is no software update available for this domain.",
            "AutomatedUpdateDate": 0.0
        },
        "DomainEndpointOptions": {
            "EnforceHTTPS": true,
            "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
        },
        "AdvancedSecurityOptions": {
            "Enabled": true,
            "InternalUserDatabaseEnabled": true
        }
    }
```

```
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的创建和管理 Amazon Elasticsearch Service 域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeElasticsearchDomain。

describe-elasticsearch-domains

以下代码示例演示了如何使用 describe-elasticsearch-domains。

Amazon CLI

获取一个或多个域的详细信息

以下 describe-elasticsearch-domains 示例提供了一个或多个域的配置详细信息。

```
aws es describe-elasticsearch-domains \
    --domain-names cli-example-1 cli-example-2
```

输出：

```
{
    "DomainStatusList": [{
            "DomainId": "123456789012/cli-example-1",
            "DomainName": "cli-example-1",
            "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example-1",
            "Created": true,
            "Deleted": false,
            "Endpoint": "search-cli-example-1-1a2a3a4a5a6a7a8a9a0a.us-
east-1.es.amazonaws.com",
            "Processing": false,
            "UpgradeProcessing": false,
            "ElasticsearchVersion": "7.4",
            "ElasticsearchClusterConfig": {
                "InstanceType": "c5.large.elasticsearch",
                "InstanceCount": 1,
                 "DedicatedMasterEnabled": true,
                "ZoneAwarenessEnabled": false,
                "DedicatedMasterType": "c5.large.elasticsearch",
                "DedicatedMasterCount": 3,
                "WarmEnabled": true,
                "WarmType": "ultrawarm1.medium.elasticsearch",
```

```
                "WarmCount": 2
            },
            "EBSOptions": {
                "EBSEnabled": true,
                "VolumeType": "gp2",
                "VolumeSize": 10
            },
            "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":
\"arn:aws:es:us-east-1:123456789012:domain/cli-example-1/*\"}]}",
            "SnapshotOptions": {
                "AutomatedSnapshotStartHour": 0
            },
            "CognitoOptions": {
                "Enabled": false
            },
            "EncryptionAtRestOptions": {
                "Enabled": true,
                "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
            },
            "NodeToNodeEncryptionOptions": {
                "Enabled": true
            },
            "AdvancedOptions": {
                "rest.action.multi.allow_explicit_index": "true"
            },
            "ServiceSoftwareOptions": {
                "CurrentVersion": "R20200522",
                "NewVersion": "",
                "UpdateAvailable": false,
                "Cancellable": false,
                "UpdateStatus": "COMPLETED",
                "Description": "There is no software update available for this
 domain.",
                "AutomatedUpdateDate": 0.0
            },
            "DomainEndpointOptions": {
                "EnforceHTTPS": true,
                "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
            },
            "AdvancedSecurityOptions": {
                "Enabled": true,
                "InternalUserDatabaseEnabled": true
```

```
            }
        },
        {
            "DomainId": "123456789012/cli-example-2",
            "DomainName": "cli-example-2",
            "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example-2",
            "Created": true,
            "Deleted": false,
            "Processing": true,
            "UpgradeProcessing": false,
            "ElasticsearchVersion": "7.4",
            "ElasticsearchClusterConfig": {
                "InstanceType": "r5.large.elasticsearch",
                "InstanceCount": 1,
                "DedicatedMasterEnabled": false,
                "ZoneAwarenessEnabled": false,
                "WarmEnabled": false
            },
            "EBSOptions": {
                "EBSEnabled": true,
                "VolumeType": "gp2",
                "VolumeSize": 10
            },
            "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect
\":\"Deny\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":
\"arn:aws:es:us-east-1:123456789012:domain/cli-example-2/*\"}]}",
            "SnapshotOptions": {
                "AutomatedSnapshotStartHour": 0
            },
            "CognitoOptions": {
                "Enabled": false
            },
            "EncryptionAtRestOptions": {
                "Enabled": false
            },
            "NodeToNodeEncryptionOptions": {
                "Enabled": false
            },
            "AdvancedOptions": {
                "rest.action.multi.allow_explicit_index": "true"
            },
            "ServiceSoftwareOptions": {
                "CurrentVersion": "",
                "NewVersion": "",
```

```
                "UpdateAvailable": false,
                "Cancellable": false,
                "UpdateStatus": "COMPLETED",
                "Description": "There is no software update available for this
 domain.",
                "AutomatedUpdateDate": 0.0
            },
            "DomainEndpointOptions": {
                "EnforceHTTPS": false,
                "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
            },
            "AdvancedSecurityOptions": {
                "Enabled": false,
                "InternalUserDatabaseEnabled": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的创建和管理 Amazon Elasticsearch Service 域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeElasticsearchDomains。

describe-reserved-elasticsearch-instances

以下代码示例演示了如何使用 describe-reserved-elasticsearch-instances。

Amazon CLI

查看所有预留实例

以下 describe-elasticsearch-domains 示例提供了您在某个区域中预留的所有实例的摘要。

```
aws es describe-reserved-elasticsearch-instances
```

输出：

```
{
    "ReservedElasticsearchInstances": [{
        "FixedPrice": 100.0,
```

```
        "ReservedElasticsearchInstanceOfferingId":
 "1a2a3a4a5-1a2a-3a4a-5a6a-1a2a3a4a5a6a",
        "ReservationName": "my-reservation",
        "PaymentOption": "PARTIAL_UPFRONT",
        "UsagePrice": 0.0,
        "ReservedElasticsearchInstanceId": "9a8a7a6a-5a4a-3a2a-1a0a-9a8a7a6a5a4a",
        "RecurringCharges": [{
            "RecurringChargeAmount": 0.603,
            "RecurringChargeFrequency": "Hourly"
        }],
        "State": "payment-pending",
        "StartTime": 1522872571.229,
        "ElasticsearchInstanceCount": 3,
        "Duration": 31536000,
        "ElasticsearchInstanceType": "m4.2xlarge.elasticsearch",
        "CcurrencyCode": "USD"
    }]
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的预留实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedElasticsearchInstances。

`list-domain-names`

以下代码示例演示了如何使用 `list-domain-names`。

Amazon CLI

列出所有域

以下 `list-domain-names` 示例提供了相应区域中所有域的快速摘要。

```
aws es list-domain-names
```

输出：

```
{
    "DomainNames": [{
            "DomainName": "cli-example-1"
        },
```

```
        {
            "DomainName": "cli-example-2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Elasticsearch Service 开发人员指南》中的创建和管理 Amazon Elasticsearch Service 域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDomainNames。

使用Amazon CLI的 Organizations 示例

以下代码示例显示了如何通过将 Amazon Command Line Interface与 Organizations 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-handshake`

以下代码示例演示了如何使用 `accept-handshake`。

Amazon CLI

接受来自其他账户的握手

组织的所有者 Bill 此前曾邀请 Juan 的账户加入他的组织。以下示例显示 Juan 的账户接受了握手，也就是同意了邀请。

```
aws organizations accept-handshake --handshake-id h-examplehandshakeid111
```

输出显示以下内容：

```
{
        "Handshake": {
                "Action": "INVITE",
                "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                "RequestedTimestamp": 1481656459.257,
                "ExpirationTimestamp": 1482952459.257,
                "Id": "h-examplehandshakeid111",
                "Parties": [
                        {
                                "Id": "o-exampleorgid",
                                "Type": "ORGANIZATION"
                        },
                        {
                                "Id": "juan@example.com",
                                "Type": "EMAIL"
                        }
                ],
                "Resources": [
                        {
                                "Resources": [
                                        {
                                                "Type": "MASTER_EMAIL",
                                                "Value": "bill@amazon.com"
                                        },
                                        {
                                                "Type": "MASTER_NAME",
                                                "Value": "Org Master Account"
                                        },
                                        {
                                                "Type": "ORGANIZATION_FEATURE_SET",
                                                "Value": "ALL"
                                        }
                                ],
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid"
                        },
                        {
                                "Type": "EMAIL",
                                "Value": "juan@example.com"
                        }
                ],
```

```
                "State": "ACCEPTED"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptHandshake。

attach-policy

以下代码示例演示了如何使用 attach-policy。

Amazon CLI

将策略附加到根、OU 或账户

示例 1：

以下示例演示如何将服务控制策略（SCP）附加到 OU：

```
aws organizations attach-policy
                --policy-id p-examplepolicyid111
                --target-id ou-examplerootid111-exampleouid111
```

示例 2：

以下示例演示如何将服务控制策略直接附加到账户：

```
aws organizations attach-policy
                --policy-id p-examplepolicyid111
                --target-id 333333333333
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachPolicy。

cancel-handshake

以下代码示例演示了如何使用 cancel-handshake。

Amazon CLI

取消其他账户发送的握手

Bill 之前曾向 Susan 的账户发送过加入其组织的邀请。现在他改变了主意，决定在 Susan 接受邀请之前取消邀请。以下示例显示了 Bill 取消邀请：

```
aws organizations cancel-handshake --handshake-id h-examplehandshakeid111
```

输出包括一个握手对象，以显示状态目前为 CANCELED：

```
{
        "Handshake": {
                "Id": "h-examplehandshakeid111",
                "State":"CANCELED",
                "Action": "INVITE",
                "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                "Parties": [
                        {
                                "Id": "o-exampleorgid",
                                "Type": "ORGANIZATION"
                        },
                        {
                                "Id": "susan@example.com",
                                "Type": "EMAIL"
                        }
                ],
                "Resources": [
                        {
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid",
                                "Resources": [
                                        {
                                                "Type": "MASTER_EMAIL",
                                                "Value": "bill@example.com"
                                        },
                                        {
                                                "Type": "MASTER_NAME",
                                                "Value": "Master Account"
                                        },
                                        {
                                                "Type": "ORGANIZATION_FEATURE_SET",
                                                "Value": "CONSOLIDATED_BILLING"
                                        }
                                ]
                        },
                        {
                                "Type": "EMAIL",
                                "Value": "anika@example.com"
```

```
                        },
                        {
                                "Type": "NOTES",
                                "Value": "This is a request for Susan's account to
 join Bob's organization."
                        }
                ],
                "RequestedTimestamp": 1.47008383521E9,
                "ExpirationTimestamp": 1.47137983521E9
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelHandshake。

`create-account`

以下代码示例演示了如何使用 `create-account`。

Amazon CLI

创建自动属于组织的成员账户

以下示例演示如何创建组织的成员账户。为成员账户配置的名称为 Production Account，电子邮件地址为 susan@example.com。由于未指定 roleName 参数，组织会使用默认名称 OrganizationAccountAccessRole 自动创建 IAM 角色。此外，由于未指定 IamUserAccessToBilling 参数，支持具有足够权限的 IAM 用户或角色访问账户账单数据的设置被设置为默认值 ALLOW。组织会自动向 Susan 发送一封“欢迎使用 Amazon”电子邮件：

```
aws organizations create-account --email susan@example.com --account-
name "Production Account"
```

输出包括一个请求对象，以显示状态目前为 IN_PROGRESS：

```
{
        "CreateAccountStatus": {
                "State": "IN_PROGRESS",
                "Id": "car-examplecreateaccountrequestid111"
        }
}
```

稍后，您可以通过向 describe-create-account-status 命令提供 ID 响应值作为 create-account-request-id 参数的值，来查询请求的当前状态。

有关更多信息，请参阅《Amazon Organizations 用户指南》中的“在您的组织中创建 Amazon 账户”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccount。

`create-organization`

以下代码示例演示了如何使用 `create-organization`。

Amazon CLI

示例 1：创建新组织

Bill 想使用账户 111111111111 中的凭证创建一个组织。以下示例显示该账户成为新组织中的主账户。由于他没有指定功能集，因此，新组织默认为在根上启用所有功能并启用服务控制策略。

```
aws organizations create-organization
```

输出包括一个组织对象，其中包含有关新组织的详细信息：

```
{
        "Organization": {
                "AvailablePolicyTypes": [
                        {
                                "Status": "ENABLED",
                                "Type": "SERVICE_CONTROL_POLICY"
                        }
                ],
                "MasterAccountId": "111111111111",
                "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/111111111111",
                "MasterAccountEmail": "bill@example.com",
                "FeatureSet": "ALL",
                "Id": "o-exampleorgid",
                "Arn": "arn:aws:organizations::111111111111:organization/o-
exampleorgid"
        }
}
```

示例 2：创建仅启用整合账单功能的新组织

以下示例创建仅支持整合账单功能的组织：

```
aws organizations create-organization --feature-set CONSOLIDATED_BILLING
```

输出包括一个组织对象，其中包含有关新组织的详细信息：

```
{
        "Organization": {
                "Arn": "arn:aws:organizations::111111111111:organization/o-
exampleorgid",
                "AvailablePolicyTypes": [],
                "Id": "o-exampleorgid",
                "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/111111111111",
                "MasterAccountEmail": "bill@example.com",
                "MasterAccountId": "111111111111",
                "FeatureSet": "CONSOLIDATED_BILLING"
        }
}
```

有关更多信息，请参阅《Amazon Organizations 用户指南》中的“创建组织”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOrganization。

create-organizational-unit

以下代码示例演示了如何使用 create-organizational-unit。

Amazon CLI

在根 OU 或父 OU 中创建 OU

以下示例演示如何创建名为 AccountingOU 的 OU：

```
aws organizations create-organizational-unit --parent-id r-examplerootid111 --
name AccountingOU
```

输出包括一个 organizationalUnit 对象，其中包含有关新 OU 的详细信息：

```
{
```

```
        "OrganizationalUnit": {
                "Id": "ou-examplerootid111-exampleouid111",
                "Arn": "arn:aws:organizations::111111111111:ou/o-exampleorgid/ou-
examplerootid111-exampleouid111",
                "Name": "AccountingOU"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOrganizationalUnit。

create-policy

以下代码示例演示了如何使用 create-policy。

Amazon CLI

示例 1：使用 JSON 策略的文本源文件创建策略

以下示例演示如何创建名为 AllowAllS3Actions 的服务控制策略（SCP）。策略内容取自本地计算机上名为 policy.json 的文件。

```
aws organizations create-policy --content file://policy.json --
name AllowAllS3Actions, --type SERVICE_CONTROL_POLICY --description "Allows
 delegation of all S3 actions"
```

输出包括一个策略对象，其中包含有关新策略的详细信息：

```
{
        "Policy": {
                "Content": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Action\":[\"s3:*\"],\"Resource\":[\"*\"]}]}",
                "PolicySummary": {
                        "Arn": "arn:aws:organizations::o-exampleorgid:policy/
service_control_policy/p-examplepolicyid111",
                        "Description": "Allows delegation of all S3 actions",
                        "Name": "AllowAllS3Actions",
                        "Type":"SERVICE_CONTROL_POLICY"
                }
        }
}
```

示例 2：创建以 JSON 策略作为参数的策略

以下示例演示了如何创建相同的 SCP，这次是将策略内容作为 JSON 字符串嵌入到参数中。字符串必须在双引号前使用反斜杠进行转义，以确保在参数中将其视为文本，参数本身用双引号引起来：

```
aws organizations create-policy --content "{\"Version\":\"2012-10-17\",\"Statement
\":[{\"Effect\":\"Allow\",\"Action\":[\"s3:*\"],\"Resource\":[\"*\"]}]}" --
name AllowAllS3Actions --type SERVICE_CONTROL_POLICY --description "Allows
 delegation of all S3 actions"
```

有关在组织中创建和使用策略的更多信息，请参阅《Amazon Organizations 用户指南》中的“管理组织策略”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicy。

decline-handshake

以下代码示例演示了如何使用 decline-handshake。

Amazon CLI

拒绝其他账户发送的握手

以下示例显示，账户 222222222222 的所有者 Susan 拒绝了加入 Bill 组织的邀请。DeclineHandshake 操作返回一个握手对象，显示该状态现在为“DECLINED”：

```
aws organizations decline-handshake --handshake-id h-examplehandshakeid111
```

输出包括一个握手对象，以显示新状态 DECLINED：

```
{
        "Handshake": {
                "Id": "h-examplehandshakeid111",
                "State": "DECLINED",
                "Resources": [
                        {
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid",
                                "Resources": [
                                        {
                                                "Type": "MASTER_EMAIL",
                                                "Value": "bill@example.com"
                                        },
```

```
                                        {
                                                "Type": "MASTER_NAME",
                                                "Value": "Master Account"
                                        }
                                ]
                        },
                        {
                                "Type": "EMAIL",
                                "Value": "susan@example.com"
                        },
                        {
                                "Type": "NOTES",
                                "Value": "This is an invitation to Susan's account
 to join the Bill's organization."
                        }
                ],
                "Parties": [
                        {
                                "Type": "EMAIL",
                                "Id": "susan@example.com"
                        },
                        {
                                "Type": "ORGANIZATION",
                                "Id": "o-exampleorgid"
                        }
                ],
                "Action": "INVITE",
                "RequestedTimestamp": 1470684478.687,
                "ExpirationTimestamp": 1471980478.687,
                "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeclineHandshake。

delete-organization

以下代码示例演示了如何使用 delete-organization。

Amazon CLI

删除组织

以下示例演示如何删除组织。要执行此操作，您必须是组织中主账户的管理员。该示例假设您之前已从组织中删除所有成员账户、OU 和策略：

```
aws organizations delete-organization
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOrganization。

delete-organizational-unit

以下代码示例演示了如何使用 delete-organizational-unit。

Amazon CLI

删除 OU

以下示例说明如何删除 OU。该示例假设您之前已从 OU 中删除所有账户和其他 OU：

```
aws organizations delete-organizational-unit --organizational-unit-id ou-examplerootid111-exampleouid111
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOrganizationalUnit。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除策略

以下示例演示如何删除组织的策略。该示例假设您之前已将策略与所有实体分离：

```
aws organizations delete-policy --policy-id p-examplepolicyid111
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

describe-account

以下代码示例演示了如何使用 describe-account。

Amazon CLI

获取有关账户的详细信息

以下示例说明如何请求有关账户的详细信息：

```
aws organizations describe-account --account-id 555555555555
```

输出显示了一个账户对象，其中包含有关该账户的详细信息：

```
{
        "Account": {
                "Id": "555555555555",
                "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/555555555555",
                "Name": "Beta account",
                "Email": "anika@example.com",
                "JoinedMethod": "INVITED",
                "JoinedTimeStamp": 1481756563.134,
                "Status": "ACTIVE"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccount。

describe-create-account-status

以下代码示例演示了如何使用 describe-create-account-status。

Amazon CLI

获取有关账户创建请求的最新状态

以下示例演示如何请求先前在组织中创建账户的请求的最新状态。指定的 --request-id 来自最初调用 create-account 的响应。账户创建请求通过状态字段显示组织已成功完成账户的创建。

命令:

```
aws organizations describe-create-account-status --create-account-request-id car-
examplecreateaccountrequestid111
```

输出：

```
{
  "CreateAccountStatus": {
    "State": "SUCCEEDED",
    "AccountId": "555555555555",
    "AccountName": "Beta account",
    "RequestedTimestamp": 1470684478.687,
    "CompletedTimestamp": 1470684532.472,
    "Id": "car-examplecreateaccountrequestid111"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCreateAccountStatus。

describe-handshake

以下代码示例演示了如何使用 describe-handshake。

Amazon CLI

获取有关握手的信息

以下示例说明如何请求有关握手的详细信息。握手 ID 要么来自对 InviteAccountToOrganization 的原始调用，要么来自对 ListHandshakesForAccount 或 ListHandshakesForOrganization 的调用：

```
aws organizations describe-handshake --handshake-id h-examplehandshakeid111
```

输出包括一个握手对象，其中包含有关请求的握手的所有详细信息：

```
{
        "Handshake": {
                "Id": "h-examplehandshakeid111",
                "State": "OPEN",
                "Resources": [
                        {
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid",
                                "Resources": [
                                        {
```

```
                                        "Type": "MASTER_EMAIL",
                                        "Value": "bill@example.com"
                                },
                                {
                                        "Type": "MASTER_NAME",
                                        "Value": "Master Account"
                                }
                        ]
                },
                {
                        "Type": "EMAIL",
                        "Value": "anika@example.com"
                }
        ],
        "Parties": [
                {
                        "Type": "ORGANIZATION",
                        "Id": "o-exampleorgid"
                },
                {
                        "Type": "EMAIL",
                        "Id": "anika@example.com"
                }
        ],
        "Action": "INVITE",
        "RequestedTimestamp": 1470158698.046,
        "ExpirationTimestamp": 1471454698.046,
        "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHandshake。

describe-organization

以下代码示例演示了如何使用 describe-organization。

Amazon CLI

获取有关当前组织的信息

以下示例说明如何请求有关组织的详细信息：

```
aws organizations describe-organization
```

输出包括一个组织对象，其中包含有关组织的详细信息：

```
{
        "Organization": {
                "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/111111111111",
                "MasterAccountEmail": "bill@example.com",
                "MasterAccountId": "111111111111",
                "Id": "o-exampleorgid",
                "FeatureSet": "ALL",
                "Arn": "arn:aws:organizations::111111111111:organization/o-
exampleorgid",
                "AvailablePolicyTypes": [
                        {
                                "Status": "ENABLED",
                                "Type": "SERVICE_CONTROL_POLICY"
                        }
                ]
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrganization。

describe-organizational-unit

以下代码示例演示了如何使用 describe-organizational-unit。

Amazon CLI

获取有关 OU 的信息

以下 describe-organizational-unit 示例请求有关 OU 的详细信息。

```
aws organizations describe-organizational-unit \
    --organizational-unit-id ou-examplerootid111-exampleouid111
```

输出：

```
{
```

```
    "OrganizationalUnit": {
        "Name": "Accounting Group",
        "Arn": "arn:aws:organizations::123456789012:ou/o-exampleorgid/ou-
examplerootid111-exampleouid111",
        "Id": "ou-examplerootid111-exampleouid111"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrganizationalUnit。

describe-policy

以下代码示例演示了如何使用 describe-policy。

Amazon CLI

获取有关策略的信息

以下示例演示如何请求有关策略的信息：

```
aws organizations describe-policy --policy-id p-examplepolicyid111
```

输出包括一个策略对象，其中包含有关策略的详细信息：

```
{
        "Policy": {
                "Content": "{\n  \"Version\": \"2012-10-17\",\n  \"Statement\": [\n
    {\n      \"Effect\": \"Allow\",\n      \"Action\": \"*\",\n      \"Resource\":
 \"*\"\n    }\n  ]\n}",
                "PolicySummary": {
                        "Arn": "arn:aws:organizations::111111111111:policy/o-
exampleorgid/service_control_policy/p-examplepolicyid111",
                        "Type": "SERVICE_CONTROL_POLICY",
                        "Id": "p-examplepolicyid111",
                        "AwsManaged": false,
                        "Name": "AllowAllS3Actions",
                        "Description": "Enables admins to delegate S3 permissions"
                }
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePolicy。

detach-policy

以下代码示例演示了如何使用 detach-policy。

Amazon CLI

从根、OU 或账户分离策略

以下示例演示了如何从 OU 分离策略：

```
aws organizations  detach-policy  --target-id ou-examplerootid111-exampleouid111 --
policy-id p-examplepolicyid111
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachPolicy。

disable-policy-type

以下代码示例演示了如何使用 disable-policy-type。

Amazon CLI

在根中禁用策略类型

以下示例演示如何在根中禁用服务控制策略（SCP）策略类型：

```
aws organizations disable-policy-type --root-id r-examplerootid111 --policy-
type SERVICE_CONTROL_POLICY
```

输出显示 PolicyTypes 响应元素不再包含 SERVICE_CONTROL_POLICY：

```
{
        "Root": {
                "PolicyTypes": [],
                "Name": "Root",
                "Id": "r-examplerootid111",
                "Arn": "arn:aws:organizations::111111111111:root/o-exampleorgid/r-
examplerootid111"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisablePolicyType。

enable-all-features

以下代码示例演示了如何使用 enable-all-features。

Amazon CLI

启用组织中的所有功能

此示例显示管理员要求组织中所有受邀账户批准启用组织中的所有功能。Amazon组织会向每个受邀成员账户注册的地址发送一封电子邮件，要求所有者接受发送的握手，以批准对所有功能的更改。在所有受邀成员账户接受握手后，组织管理员可以完成对所有功能的更改，拥有适当权限的成员可以创建策略并将其应用于根、OU 和账户：

```
aws organizations enable-all-features
```

输出是一个握手对象，发送到所有受邀成员账户进行审批：

```
{
        "Handshake": {
                "Action": "ENABLE_ALL_FEATURES",
                "Arn":"arn:aws:organizations::111111111111:handshake/o-exampleorgid/
enable_all_features/h-examplehandshakeid111",
                "ExpirationTimestamp":1.483127868609E9,
                "Id":"h-examplehandshakeid111",
                "Parties": [
                        {
                                "id":"o-exampleorgid",
                                "type":"ORGANIZATION"
                        }
                ],
                "requestedTimestamp":1.481831868609E9,
                "resources": [
                        {
                                "type":"ORGANIZATION",
                                "value":"o-exampleorgid"
                        }
                ],
                "state":"REQUESTED"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableAllFeatures。

enable-policy-type

以下代码示例演示了如何使用 enable-policy-type。

Amazon CLI

允许在根中使用策略类型

以下示例演示如何在根中启用服务控制策略（SCP）策略类型：

```
aws organizations enable-policy-type --root-id r-examplerootid111 --policy-
type SERVICE_CONTROL_POLICY
```

输出显示了一个带有 policyTypes 响应元素的根对象，该元素显示 SCP 现已启用：

```
{
        "Root": {
                "PolicyTypes": [
                        {
                                "Status":"ENABLED",
                                "Type":"SERVICE_CONTROL_POLICY"
                        }
                ],
                "Id": "r-examplerootid111",
                "Name": "Root",
                "Arn": "arn:aws:organizations::111111111111:root/o-exampleorgid/r-
examplerootid111"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnablePolicyType。

invite-account-to-organization

以下代码示例演示了如何使用 invite-account-to-organization。

Amazon CLI

邀请账户加入组织

以下示例显示了 bill@example.com 拥有的主账户邀请 juan@example.com 拥有的账户加入组织：

```
aws organizations invite-account-to-organization --target '{"Type": "EMAIL", "Id":
 "juan@example.com"}' --notes "This is a request for Juan's account to join Bill's
 organization."
```

输出包括一个握手结构，其中显示了发送到受邀账户的内容：

```
{
        "Handshake": {
                "Action": "INVITE",
                "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                "ExpirationTimestamp": 1482952459.257,
                "Id": "h-examplehandshakeid111",
                "Parties": [
                        {
                                "Id": "o-exampleorgid",
                                "Type": "ORGANIZATION"
                        },
                        {
                                "Id": "juan@example.com",
                                "Type": "EMAIL"
                        }
                ],
                "RequestedTimestamp": 1481656459.257,
                "Resources": [
                        {
                                "Resources": [
                                        {
                                                "Type": "MASTER_EMAIL",
                                                "Value": "bill@amazon.com"
                                        },
                                        {
                                                "Type": "MASTER_NAME",
                                                "Value": "Org Master Account"
                                        },
                                        {
                                                "Type": "ORGANIZATION_FEATURE_SET",
                                                "Value": "FULL"
                                        }
                                ],
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid"
                        },
```

```
                        {
                                "Type": "EMAIL",
                                "Value": "juan@example.com"
                        }
                ],
                "State": "OPEN"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InviteAccountToOrganization。

`leave-organization`

以下代码示例演示了如何使用 `leave-organization`。

Amazon CLI

作为成员账户退出组织

以下示例显示了一个成员账户的管理员请求离开其当前所属的组织：

```
aws organizations leave-organization
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 LeaveOrganization。

`list-accounts-for-parent`

以下代码示例演示了如何使用 `list-accounts-for-parent`。

Amazon CLI

检索指定父根或 OU 中所有账户的列表

以下示例演示了如何请求 OU 中的账户列表：

```
aws organizations list-accounts-for-parent --parent-id ou-examplerootid111-exampleouid111
```

输出包含账户摘要对象的列表。

```
{
```

```
        "Accounts": [
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/333333333333",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481835795.536,
                        "Id": "333333333333",
                        "Name": "Development Account",
                        "Email": "juan@example.com",
                        "Status": "ACTIVE"
                },
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/444444444444",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481835812.143,
                        "Id": "444444444444",
                        "Name": "Test Account",
                        "Email": "anika@example.com",
                        "Status": "ACTIVE"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccountsForParent。

`list-accounts`

以下代码示例演示了如何使用 `list-accounts`。

Amazon CLI

检索组织中所有账户的列表

以下示例演示了如何请求组织中的账户列表：

```
aws organizations list-accounts
```

输出包含账户摘要对象的列表。

```
{
        "Accounts": [
```

```
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/111111111111",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481830215.45,
                        "Id": "111111111111",
                        "Name": "Master Account",
                        "Email": "bill@example.com",
                        "Status": "ACTIVE"
                },
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/222222222222",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481835741.044,
                        "Id": "222222222222",
                        "Name": "Production Account",
                        "Email": "alice@example.com",
                        "Status": "ACTIVE"
                },
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/333333333333",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481835795.536,
                        "Id": "333333333333",
                        "Name": "Development Account",
                        "Email": "juan@example.com",
                        "Status": "ACTIVE"
                },
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/444444444444",
                        "JoinedMethod": "INVITED",
                        "JoinedTimestamp": 1481835812.143,
                        "Id": "444444444444",
                        "Name": "Test Account",
                        "Email": "anika@example.com",
                        "Status": "ACTIVE"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccounts。

`list-children`

以下代码示例演示了如何使用 `list-children`。

Amazon CLI

检索父 OU 或根的子账户和 OU

以下示例演示了如何列出包含该帐户 444444444444 的根或 OU：

```
aws organizations list-children --child-type ORGANIZATIONAL_UNIT --parent-id ou-examplerootid111-exampleouid111
```

输出显示了父 OU 包含的两个子 OU：

```
{
        "Children": [
                {
                        "Id": "ou-examplerootid111-exampleouid111",
                        "Type":"ORGANIZATIONAL_UNIT"
                },
                {
                        "Id":"ou-examplerootid111-exampleouid222",
                        "Type":"ORGANIZATIONAL_UNIT"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChildren。

`list-create-account-status`

以下代码示例演示了如何使用 `list-create-account-status`。

Amazon CLI

示例 1：检索在当前组织中提出的账户创建请求的列表

以下示例演示如何请求组织中已成功完成的账户创建请求的列表：

```
aws organizations list-create-account-status --states SUCCEEDED
```

输出包括一个对象数组，其中包含有关每个请求的信息。

```
{
        "CreateAccountStatuses": [
                {
                        "AccountId": "444444444444",
                        "AccountName": "Developer Test Account",
                        "CompletedTimeStamp": 1481835812.143,
                        "Id": "car-examplecreateaccountrequestid111",
                        "RequestedTimeStamp": 1481829432.531,
                        "State": "SUCCEEDED"
                }
        ]
}
```

示例 2：检索在当前组织中提出的进行中账户创建请求的列表

以下示例获取正在处理的组织账户创建请求的列表：

```
aws organizations list-create-account-status --states IN_PROGRESS
```

输出包括一个对象数组，其中包含有关每个请求的信息。

```
{
        "CreateAccountStatuses": [
                {
                  "State": "IN_PROGRESS",
                  "Id": "car-examplecreateaccountrequestid111",
                  "RequestedTimeStamp": 1481829432.531,
                  "AccountName": "Production Account"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCreateAccountStatus。

list-handshakes-for-account

以下代码示例演示了如何使用 list-handshakes-for-account。

Amazon CLI

检索发送到账户的握手清单

以下示例演示如何获取与用于调用操作的凭证账户关联的所有握手列表：

```
aws organizations list-handshakes-for-account
```

输出包括握手结构列表，其中包含有关每次握手的信息，包括其当前状态：

```
{
        "Handshake": {
                "Action": "INVITE",
                "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                "ExpirationTimestamp": 1482952459.257,
                "Id": "h-examplehandshakeid111",
                "Parties": [
                        {
                                "Id": "o-exampleorgid",
                                "Type": "ORGANIZATION"
                        },
                        {
                                "Id": "juan@example.com",
                                "Type": "EMAIL"
                        }
                ],
                "RequestedTimestamp": 1481656459.257,
                "Resources": [
                        {
                                "Resources": [
                                        {
                                                "Type": "MASTER_EMAIL",
                                                "Value": "bill@amazon.com"
                                        },
                                        {
                                                "Type": "MASTER_NAME",
                                                "Value": "Org Master Account"
                                        },
                                        {
                                                "Type": "ORGANIZATION_FEATURE_SET",
                                                "Value": "FULL"
                                        }
```

```
                                ],
                                "Type": "ORGANIZATION",
                                "Value": "o-exampleorgid"
                        },
                        {
                                "Type": "EMAIL",
                                "Value": "juan@example.com"
                        }
                ],
                "State": "OPEN"
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHandshakesForAccount。

`list-handshakes-for-organization`

以下代码示例演示了如何使用 `list-handshakes-for-organization`。

Amazon CLI

检索与组织相关的握手列表

以下示例演示如何获取与当前组织关联的握手列表：

```
aws organizations list-handshakes-for-organization
```

输出显示两个握手。第一个是对 Juan 账户的邀请，显示的状态为 OPEN。第二个是对 Anika 账户的邀请，显示的状态为 ACCEPTED：

```
{
        "Handshakes": [
                {
                        "Action": "INVITE",
                        "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                        "ExpirationTimestamp": 1482952459.257,
                        "Id": "h-examplehandshakeid111",
                        "Parties": [
                                {
                                        "Id": "o-exampleorgid",
                                        "Type": "ORGANIZATION"
```

```
                                },
                                {
                                        "Id": "juan@example.com",
                                        "Type": "EMAIL"
                                }
                        ],
                        "RequestedTimestamp": 1481656459.257,
                        "Resources": [
                                {
                                        "Resources": [
                                                {
                                                        "Type": "MASTER_EMAIL",
                                                        "Value": "bill@amazon.com"
                                                },
                                                {
                                                        "Type": "MASTER_NAME",
                                                        "Value": "Org Master
Account"
                                                },
                                                {
                                                        "Type":
"ORGANIZATION_FEATURE_SET",
                                                        "Value": "FULL"
                                                }
                                        ],
                                        "Type": "ORGANIZATION",
                                        "Value": "o-exampleorgid"
                                },
                                {
                                        "Type": "EMAIL",
                                        "Value": "juan@example.com"
                                },
                                {
                                        "Type":"NOTES",
                                        "Value":"This is an invitation to Juan's
account to join Bill's organization."
                                }
                        ],
                        "State": "OPEN"
                },
                {
                        "Action": "INVITE",
                        "State":"ACCEPTED",
```

```
                        "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
                        "ExpirationTimestamp": 1.471797437427E9,
                        "Id": "h-examplehandshakeid222",
                        "Parties": [
                                {
                                        "Id": "o-exampleorgid",
                                        "Type": "ORGANIZATION"
                                },
                                {
                                        "Id": "anika@example.com",
                                        "Type": "EMAIL"
                                }
                        ],
                        "RequestedTimestamp": 1.469205437427E9,
                        "Resources": [
                                {
                                        "Resources": [
                                                {
                                                        "Type":"MASTER_EMAIL",
                                                        "Value":"bill@example.com"
                                                },
                                                {
                                                        "Type":"MASTER_NAME",
                                                        "Value":"Master Account"
                                                }
                                        ],
                                        "Type":"ORGANIZATION",
                                        "Value":"o-exampleorgid"
                                },
                                {
                                        "Type":"EMAIL",
                                        "Value":"anika@example.com"
                                },
                                {
                                        "Type":"NOTES",
                                        "Value":"This is an invitation to Anika's
 account to join Bill's organization."
                                }
                        ]
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHandshakesForOrganization。

`list-organizational-units-for-parent`

以下代码示例演示了如何使用 `list-organizational-units-for-parent`。

Amazon CLI

检索父级 OU 或根中的 OU 列表

以下示例演示了如何获取指定根中的 OU 列表：

```
aws organizations list-organizational-units-for-parent --parent-id r-
examplerootid111
```

输出显示指定的根包含两个 OU，并显示了每个 OU 的详细信息：

```
{
        "OrganizationalUnits": [
                {
                        "Name": "AccountingDepartment",
                        "Arn": "arn:aws:organizations::o-exampleorgid:ou/r-
examplerootid111/ou-examplerootid111-exampleouid111"
                },
                {
                        "Name": "ProductionDepartment",
                        "Arn": "arn:aws:organizations::o-exampleorgid:ou/r-
examplerootid111/ou-examplerootid111-exampleouid222"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizationalUnitsForParent。

`list-parents`

以下代码示例演示了如何使用 `list-parents`。

Amazon CLI

列出账户或子 OU 的父 OU 或根

以下示例演示了如何列出包含该账户 444444444444 的根或父 OU：

```
aws organizations list-parents --child-id 444444444444
```

输出显示指定账户位于具有指定 ID 的 OU 中：

```
{
  "Parents": [
        {
          "Id": "ou-examplerootid111-exampleouid111",
          "Type": "ORGANIZATIONAL_UNIT"
        }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListParents。

list-policies-for-target

以下代码示例演示了如何使用 list-policies-for-target。

Amazon CLI

检索直接关联到账户的 SCP 列表

以下示例演示如何获取由筛选条件参数指定的、直接附加到账户的所有服务控制策略（SCP）的列表：

```
aws organizations list-policies-for-target --filter SERVICE_CONTROL_POLICY --target-
id 444444444444
```

输出包括含策略摘要信息的策略结构列表：该列表不包括适用于该账户的策略，因为这些策略是从账户在 OU 层次结构中的位置继承的：

```
{
        "Policies": [
                {
                        "Type": "SERVICE_CONTROL_POLICY",
                        "Name": "AllowAllEC2Actions",
                        "AwsManaged", false,
                        "Id": "p-examplepolicyid222",
```

```
                        "Arn": "arn:aws:organizations::o-exampleorgid:policy/
service_control_policy/p-examplepolicyid222",
                        "Description": "Enables account admins to delegate
 permissions for any EC2 actions to users and roles in their accounts."
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPoliciesForTarget。

`list-policies`

以下代码示例演示了如何使用 `list-policies`。

Amazon CLI

检索特定类型组织中所有策略的列表

以下示例演示了如何获取由筛选条件参数指定的 SCP 列表：

```
aws organizations list-policies --filter SERVICE_CONTROL_POLICY
```

输出包括含摘要信息的策略列表：

```
{
        "Policies": [
                {
                        "Type": "SERVICE_CONTROL_POLICY",
                        "Name": "AllowAllS3Actions",
                        "AwsManaged": false,
                        "Id": "p-examplepolicyid111",
                        "Arn": "arn:aws:organizations::111111111111:policy/
service_control_policy/p-examplepolicyid111",
                        "Description": "Enables account admins to delegate
 permissions for any S3 actions to users and roles in their accounts."
                },
                {
                        "Type": "SERVICE_CONTROL_POLICY",
                        "Name": "AllowAllEC2Actions",
                        "AwsManaged": false,
                        "Id": "p-examplepolicyid222",
                        "Arn": "arn:aws:organizations::111111111111:policy/
service_control_policy/p-examplepolicyid222",
```

```
                        "Description": "Enables account admins to delegate
 permissions for any EC2 actions to users and roles in their accounts."
                },
                {
                        "AwsManaged": true,
                        "Description": "Allows access to every operation",
                        "Type": "SERVICE_CONTROL_POLICY",
                        "Id": "p-FullAWSAccess",
                        "Arn": "arn:aws:organizations::aws:policy/
service_control_policy/p-FullAWSAccess",
                        "Name": "FullAWSAccess"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicies。

list-roots

以下代码示例演示了如何使用 `list-roots`。

Amazon CLI

检索组织中根的列表

本示例演示如何获取组织的根列表：

```
aws organizations list-roots
```

输出包括含摘要信息的根结构列表：

```
{
        "Roots": [
                {
                        "Name": "Root",
                        "Arn": "arn:aws:organizations::111111111111:root/o-
exampleorgid/r-examplerootid111",
                        "Id": "r-examplerootid111",
                        "PolicyTypes": [
                                {
                                        "Status":"ENABLED",
                                        "Type":"SERVICE_CONTROL_POLICY"
```

```
                                }
                        ]
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRoots。

`list-targets-for-policy`

以下代码示例演示了如何使用 `list-targets-for-policy`。

Amazon CLI

检索策略附加到的根、OU 和账户的列表

以下示例演示如何获取指定策略附加到的根、OU 和账户的列表。

```
aws organizations list-targets-for-policy --policy-id p-FullAWSAccess
```

输出显示附加到的根、OU 和账户的摘要信息的附件对象列表：

```
{
        "Targets": [
                {
                        "Arn": "arn:aws:organizations::111111111111:root/o-
exampleorgid/r-examplerootid111",
                        "Name": "Root",
                        "TargetId":"r-examplerootid111",
                        "Type":"ROOT"
                },
                {
                        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/333333333333;",
                        "Name": "Developer Test Account",
                        "TargetId": "333333333333",
                        "Type": "ACCOUNT"
                },
                {
                        "Arn":"arn:aws:organizations::111111111111:ou/o-
exampleorgid/ou-examplerootid111-exampleouid111",
                        "Name":"Accounting",
```

```
                        "TargetId":"ou-examplerootid111-exampleouid111",
                        "Type":"ORGANIZATIONAL_UNIT"
                }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetsForPolicy。

move-account

以下代码示例演示了如何使用 move-account。

Amazon CLI

在根或 OU 之间移动帐户

以下示例演示了如何将组织中的主账户从根移到 OU：

```
aws organizations move-account --account-id 333333333333 --source-parent-id r-
examplerootid111 --destination-parent-id ou-examplerootid111-exampleouid111
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MoveAccount。

remove-account-from-organization

以下代码示例演示了如何使用 remove-account-from-organization。

Amazon CLI

将账户作为主账户从组织中移除

以下示例演示了如何从组织中移除账户：

```
aws organizations remove-account-from-organization --account-id 333333333333
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveAccountFromOrganization。

update-organizational-unit

以下代码示例演示了如何使用 update-organizational-unit。

Amazon CLI

重命名 OU

此示例演示如何重命名 OU：在此示例中，OU 被重命名为“AccountingOU”：

```
aws organizations update-organizational-unit --organizational-unit-id ou-
examplerootid111-exampleouid111 --name AccountingOU
```

输出显示了新名称：

```
{
        "OrganizationalUnit": {
                "Id": "ou-examplerootid111-exampleouid111"
                "Name": "AccountingOU",
                "Arn": "arn:aws:organizations::111111111111:ou/o-exampleorgid/ou-
examplerootid111-exampleouid111""
        }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOrganizationalUnit。

update-policy

以下代码示例演示了如何使用 update-policy。

Amazon CLI

示例 1：重命名策略

以下 update-policy 示例重命名了策略并提供了新描述。

```
aws organizations update-policy \
    --policy-id p-examplepolicyid111 \
    --name Renamed-Policy \
    --description "This description replaces the original."
```

输出显示了新的名称和描述。

```
{
```

```
    "Policy": {
        "Content": "{\n  \"Version\":\"2012-10-17\",\n  \"Statement\":{\n
  \"Effect\":\"Allow\",\n    \"Action\":\"ec2:*\",\n    \"Resource\":\"*\"\n
  }\n}\n",
        "PolicySummary": {
            "Id": "p-examplepolicyid111",
            "AwsManaged": false,
            "Arn":"arn:aws:organizations::111111111111:policy/o-exampleorgid/
service_control_policy/p-examplepolicyid111",
            "Description": "This description replaces the original.",
            "Name": "Renamed-Policy",
            "Type": "SERVICE_CONTROL_POLICY"
        }
    }
}
```

示例 2：替换策略的 JSON 文本内容

以下示例演示了如何将上一个示例中 SCP 的 JSON 文本替换为支持使用 S3 而不是 EC2 的新 JSON 策略文本字符串：

```
aws organizations update-policy \
    --policy-id p-examplepolicyid111 \
    --content "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":\"Allow\",
\"Action\":\"s3:*\",\"Resource\":\"*\"}}"
```

输出显示了新名称：

```
{
    "Policy": {
        "Content": "{ \"Version\": \"2012-10-17\", \"Statement\": { \"Effect\":
 \"Allow\", \"Action\": \"s3:*\", \"Resource\": \"*\" } }",
        "PolicySummary": {
            "Arn": "arn:aws:organizations::111111111111:policy/o-exampleorgid/
service_control_policy/p-examplepolicyid111",
            "AwsManaged": false;
            "Description": "This description replaces the original.",
            "Id": "p-examplepolicyid111",
            "Name": "Renamed-Policy",
            "Type": "SERVICE_CONTROL_POLICY"
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePolicy。

Amazon Outposts使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Outposts 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`get-outpost-instance-types`

以下代码示例演示了如何使用 `get-outpost-instance-types`。

Amazon CLI

获取 Outpost 上的实例类型

以下 `get-outpost-instance-types` 示例获取指定 Outpost 的实例类型。

```
aws outposts get-outpost-instance-types \
    --outpost-id op-0ab23c4567EXAMPLE
```

输出：

```
{
    "InstanceTypes": [
        {
```

```
            "InstanceType": "c5d.large"
        },
        {
            "InstanceType": "i3en.24xlarge"
        },
        {
            "InstanceType": "m5d.large"
        },
        {
            "InstanceType": "r5d.large"
        }
    ],
    "OutpostId": "op-0ab23c4567EXAMPLE",
    "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的在 Outpost 中启动实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOutpostInstanceTypes。

`get-outpost`

以下代码示例演示了如何使用 `get-outpost`。

Amazon CLI

获取 Outpost 详细信息

以下 `get-outpost` 示例显示了指定 Outpost 的详细信息。

```
aws outposts get-outpost \
    --outpost-id op-0ab23c4567EXAMPLE
```

输出：

```
{
    "Outpost": {
        "OutpostId": "op-0ab23c4567EXAMPLE",
        "OwnerId": "123456789012",
        "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE",
```

```
        "SiteId": "os-0ab12c3456EXAMPLE",
        "Name": "EXAMPLE",
        "LifeCycleStatus": "ACTIVE",
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az1",
        "Tags": {}
    }
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用 Outposts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOutpost。

`list-outposts`

以下代码示例演示了如何使用 `list-outposts`。

Amazon CLI

列出 Outposts

以下 `list-outposts` 示例列出您的 Amazon 账户中的 Outposts。

```
aws outposts list-outposts
```

输出：

```
{
    "Outposts": [
        {
            "OutpostId": "op-0ab23c4567EXAMPLE",
            "OwnerId": "123456789012",
            "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE",
            "SiteId": "os-0ab12c3456EXAMPLE",
            "Name": "EXAMPLE",
            "Description": "example",
            "LifeCycleStatus": "ACTIVE",
            "AvailabilityZone": "us-west-2a",
            "AvailabilityZoneId": "usw2-az1",
            "Tags": {
                "Name": "EXAMPLE"
```

```
            }
        },
        {
            "OutpostId": "op-4fe3dc21baEXAMPLE",
            "OwnerId": "123456789012",
            "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-4fe3dc21baEXAMPLE",
            "SiteId": "os-0ab12c3456EXAMPLE",
            "Name": "EXAMPLE2",
            "LifeCycleStatus": "ACTIVE",
            "AvailabilityZone": "us-west-2a",
            "AvailabilityZoneId": "usw2-az1",
            "Tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用 Outposts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOutposts。

`list-sites`

以下代码示例演示了如何使用 `list-sites`。

Amazon CLI

列出站点

以下 `list-sites` 示例列出了您的 Amazon 账户中的可用 Outpost 站点。

```
aws outposts list-sites
```

输出：

```
{
    "Sites": [
        {
            "SiteId": "os-0ab12c3456EXAMPLE",
            "AccountId": "123456789012",
            "Name": "EXAMPLE",
            "Description": "example",
```

```
            "Tags": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Outposts 用户指南》中的使用 Outposts。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSites。

Amazon Payment Cryptography使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Payment Cryptography 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-alias`

以下代码示例演示了如何使用 `create-alias`。

Amazon CLI

为密钥创建别名

以下 `create-alias` 示例为密钥创建了别名。

```
aws payment-cryptography create-alias \
    --alias-name alias/sampleAlias1 \
    --key-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Alias": {
        "AliasName": "alias/sampleAlias1",
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的关于别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAlias。

create-key

以下代码示例演示了如何使用 create-key。

Amazon CLI

创建密钥

以下 create-key 示例生成 2KEY TDES 密钥，可用于生成和验证 CVV/CVV2 值。

```
aws payment-cryptography create-key \
    --exportable \
    --key-
attributes KeyAlgorithm=TDES_2KEY, KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "1686800690",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
```

```
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "F2E50F",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "1686800690"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的生成密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKey。

delete-alias

以下代码示例演示了如何使用 delete-alias。

Amazon CLI

删除别名

以下 delete-alias 示例删除了别名。该操作不会影响密钥。

```
aws payment-cryptography delete-alias \
    --alias-name alias/sampleAlias1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 支付加密用户指南》中的关于别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAlias。

delete-key

以下代码示例演示了如何使用 delete-key。

Amazon CLI

删除密钥

以下 delete-key 示例计划在 7 天后删除密钥，这是默认的等待期限。

```
aws payment-cryptography delete-key \
    --key-identifier arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "1686801198",
        "DeletePendingTimestamp": "1687405998",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "F2E50F",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
```

```
        "KeyState": "DELETE_PENDING",
        "UsageStartTimestamp": "1686801190"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的删除密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteKey。

export-key

以下代码示例演示了如何使用 export-key。

Amazon CLI

导出密钥

以下 export-key 示例导出密钥。

```
aws payment-cryptography export-key \
    --export-key-identifier arn:aws:payment-cryptography:us-west-2:123456789012:key/lco3w6agsk7zgu2l \
    --key-material '{"Tr34KeyBlock": { \
        "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-west-2:123456789012:key/ftobshq7pvioc5fx", \
        "ExportToken": "export-token-cu4lg26ofcziixny", \
        "KeyBlockFormat": "X9_TR34_2012", \
        "WrappingKeyCertificate": file://wrapping-key-certificate.pem }}'
```

的内容wrapping-key-certificate.pem：

```
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2VENDQXFXZ0F3SUJBZ0lSQU1ZZS8xMXFUK2svVzlRUDJQOElV
```

输出：

```
{
    "WrappedKey": {
        "KeyMaterial":
 "308205A106092A864886F70D010702A08205923082058E020101310D300B0609608648016503040201308203 F
        "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的导出密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExportKey。

get-alias

以下代码示例演示了如何使用 get-alias。

Amazon CLI

获取别名

以下 get-alias 示例返回与别名关联的密钥的 ARN。

```
aws payment-cryptography get-alias \
    --alias-name alias/sampleAlias1
```

输出：

```
{
    "Alias": {
        "AliasName": "alias/sampleAlias1",
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的关于别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAlias。

get-key

以下代码示例演示了如何使用 get-key。

Amazon CLI

获取密钥的元数据

以下 get-key 示例返回与别名关联的密钥的元数据。此操作不会返回加密材料。

```
aws payment-cryptography get-key \
```

```
    --key-identifier alias/sampleAlias1
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "1686800690",
        "DeletePendingTimestamp": "1687405998",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "F2E50F",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "DELETE_PENDING",
        "UsageStartTimestamp": "1686801190"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的获取密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetKey。

get-parameters-for-export

以下代码示例演示了如何使用 get-parameters-for-export。

Amazon CLI

初始化导出过程

以下 get-parameters-for-export 示例生成密钥对，对密钥进行签名，然后返回证书和证书根。

```
aws payment-cryptography get-parameters-for-export \
    --signing-key-algorithm RSA_2048 \
    --key-material-type TR34_KEY_BLOCK
```

输出：

```
{
    "ExportToken": "export-token-ep5cwyzune7oya53",
    "ParametersValidUntilTimestamp": "1687415640",
    "SigningKeyAlgorithm": "RSA_2048",
    "SigningKeyCertificate":

    "MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
    "SigningKeyCertificateChain":
    "NIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
```

```
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的导出密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetParametersForExport。

get-parameters-for-import

以下代码示例演示了如何使用 get-parameters-for-import。

Amazon CLI

初始化导入过程

以下 get-parameters-for-import 示例生成密钥对，对密钥进行签名，然后返回证书和证书根。

```
aws payment-cryptography get-parameters-for-import \
    --key-material-type TR34_KEY_BLOCK \
    --wrapping-key-algorithm RSA_2048
```

输出：

```
{
    "ImportToken": "import-token-qgmafpaa7nt2kfbb",
    "ParametersValidUntilTimestamp": "1687415640",
    "WrappingKeyAlgorithm": "RSA_2048",
    "WrappingKeyCertificate":
    "MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
    "WrappingKeyCertificateChain":
    "NIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的导入密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetParametersForImport。

get-public-key-certificate

以下代码示例演示了如何使用 get-public-key-certificate。

Amazon CLI

返回公钥

以下 get-public-key-certificate 示例返回密钥对的公钥部分。

```
aws payment-cryptography get-public-key-certificate \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "KeyCertificate":
```

```
    "MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
    "KeyCertificateChain":
    "NIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
    VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
    b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
    BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
    MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
    VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
    b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
    rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
    Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
    nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
    FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的获取与密钥对关联的公钥/证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicKeyCertificate。

`import-key`

以下代码示例演示了如何使用 `import-key`。

Amazon CLI

导入 TR-34 密钥

以下 `import-key` 示例导入 TR-34 密钥。

```
aws payment-cryptography import-key \
        --key-material='{ "Tr34KeyBlock": {" \
            CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-west-2:123456789012:key/rmm5wn2q564njnjm", \
            "ImportToken": "import-token-5ott6ho5nts7bbcg", \
            "KeyBlockFormat": "X9_TR34_2012", \
            "SigningKeyCertificate": file://signing-key-certificate.pem, \
            "WrappedKeyBlock": file://wrapped-key-block.pem }}'
```

的内容signing-key-certificate.pem：

```
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJBZ0lRYWVCK25IbE1WZU1PR1ZiNjU1Q2Jz
```

的内容wrapped-key-block.pem：

```
3082059806092A864886F70D010702A0820589308205850201013 10D300B0609608648016503040201308203160 6
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "2023-06-09T16:56:27.621000-07:00",
        "Enabled": true,
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
bzmvgyxdg3sktwxd",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
```

```
        "KeyCheckValue": "D9B20E",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "EXTERNAL",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2023-06-09T16:56:27.621000-07:00"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的导入密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportKey。

list-aliases

以下代码示例演示了如何使用 `list-aliases`。

Amazon CLI

获取别名列表

以下 `list-aliases` 示例显示了您在该地区中的账户的所有别名。

```
aws payment-cryptography list-aliases
```

输出：

```
{
    "Aliases": [
        {
            "AliasName": "alias/sampleAlias1",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h"
        },
        {
            "AliasName": "alias/sampleAlias2",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的关于别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAliases。

list-keys

以下代码示例演示了如何使用 list-keys。

Amazon CLI

获取密钥列表

以下 list-keys 示例显示了您在该地区中的账户的所有密钥。

```
aws payment-cryptography list-keys
```

输出：

```
{
    "Keys": [
    {
        "CreateTimestamp": "1666506840",
        "Enabled": false,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
```

```
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStopTimestamp": "1666938840"
    }
    ]
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的列表密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListKeys。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

获取密钥标签的列表

以下 list-tags-for-resource 示例获取密钥的标签。

```
aws payment-cryptography list-tags-for-resource \
    --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Tags": [
        {
            "Key": "BIN",
            "Value": "20151120"
        },
        {
            "Key": "Project",
            "Value": "Production"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的使用 API 操作管理密钥标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

restore-key

以下代码示例演示了如何使用 restore-key。

Amazon CLI

恢复计划删除的密钥

以下 restore-key 示例取消了对密钥的删除。

```
aws payment-cryptography restore-key \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_3KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": false,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "1686800690",
```

```
        "UsageStopTimestamp": "1687405998"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的删除密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreKey。

`start-key-usage`

以下代码示例演示了如何使用 `start-key-usage`。

Amazon CLI

启用密钥

以下 `start-key-usage` 示例允许使用密钥。

```
aws payment-cryptography start-key-usage \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "1686800690",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
```

```
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "1686800690"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的启用和禁用密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartKeyUsage。

stop-key-usage

以下代码示例演示了如何使用 stop-key-usage。

Amazon CLI

禁用密钥

以下 stop-key-usage 示例禁用密钥。

```
aws payment-cryptography stop-key-usage \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h
```

输出：

```
{
    "Key": {
        "CreateTimestamp": "1686800690",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
```

```
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "1686800690"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的启用和禁用密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopKeyUsage。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

为密钥添加标签

以下 `tag-resource` 示例为密钥添加标签。

```
aws payment-cryptography tag-resource \
    --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --tags Key=sampleTag,Value=sampleValue
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 支付加密用户指南》中的管理密钥标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从密钥中删除标签

以下 untag-resource 示例从密钥中删除标签。

```
aws payment-cryptography untag-resource \
    --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --tag-keys sampleTag
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 支付加密用户指南》中的管理密钥标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-alias

以下代码示例演示了如何使用 update-alias。

Amazon CLI

更新别名

以下 update-alias 示例将别名与其他密钥相关联。

```
aws payment-cryptography update-alias \
    --alias-name alias/sampleAlias1 \
    --key-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
tqv5yij6wtxx64pi
```

输出：

```
{
    "Alias": {
        "AliasName": "alias/sampleAlias1",
        "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
tqv5yij6wtxx64pi "
```

```
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的关于别名。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAlias。

使用 Amazon CLI 的 Amazon Payment Cryptography 数据面板

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Payment Cryptography 数据面板结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`decrypt-data`

以下代码示例演示了如何使用 `decrypt-data`。

Amazon CLI

解密加密文字

以下 `decrypt-data` 示例使用对称密钥解密加密文字数据。要执行此操作，密钥必须将 `KeyModesOfUse` 设置为 `Decrypt`，将 `KeyUsage` 设置为 `TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`。

```
aws payment-cryptography-data decrypt-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --cipher-text 33612AB9D6929C3A828EB6030082B2BD \
    --decryption-attributes 'Symmetric={Mode=CBC}'
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h",
    "KeyCheckValue": "71D7AE",
    "PlainText": "31323334313233343132333431323334"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的解密数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecryptData。

encrypt-data

以下代码示例演示了如何使用 encrypt-data。

Amazon CLI

加密数据

以下 encrypt-data 示例使用对称密钥对纯文本数据进行加密。要执行此操作，密钥必须将 KeyModesOfUse 设置为 Encrypt，将 KeyUsage 设置为 TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY。

```
aws payment-cryptography-data encrypt-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --plain-text 31323334313233343132333431323334 \
    --encryption-attributes 'Symmetric={Mode=CBC}'
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的加密数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EncryptData。

generate-card-validation-data

以下代码示例演示了如何使用 generate-card-validation-data。

Amazon CLI

生成 CVV

以下 generate-card-validation-data 示例生成了 CVV/CVV2。

```
aws payment-cryptography-data generate-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --primary-account-number=171234567890123 \
    --generation-attributes CardVerificationValue2={CardExpiryDate=0123}
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h",
    "KeyCheckValue": "CADDA1",
    "ValidationData": "801"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的生成卡数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateCardValidationData。

generate-mac

以下代码示例演示了如何使用 generate-mac。

Amazon CLI

生成 MAC

以下 generate-card-validation-data 示例使用 HMAC_SHA256 算法和 HMAC 加密密钥生成用于卡数据身份验证的 HMAC 散列消息认证码（HMAC）。密钥必须将 KeyUsage 设置为 TR31_M7_HMAC_KEY，并将 KeyModesOfUse 设置为 Generate。

```
aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h \
    --message-
data "3b31303838343930303130373333934313533d323430383232363030303730303033f33" \
    --generation-attributes Algorithm=HMAC_SHA256
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/
kwapwa6qaifllw2h,
    "KeyCheckValue": "2976E7",
    "Mac": "ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的生成 MAC。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateMac。

generate-pin-data

以下代码示例演示了如何使用 generate-pin-data。

Amazon CLI

生成 PIN

以下 generate-card-validation-data 示例使用 Visa PIN 方案生成一个新的随机 PIN 码。

```
aws payment-cryptography-data generate-pin-data \
    --generation-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2 \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --primary-account-number 171234567890123 \
    --pin-block-format ISO_FORMAT_0 \
    --generation-attributes VisaPin={PinVerificationKeyIndex=1}
```

输出：

```
{
```

```
    "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
    "GenerationKeyCheckValue": "7F2363",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
    "EncryptedPinBlock": "AC17DC148BDA645E",
    "PinData": {
        "VerificationValue": "5507"
    }
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的生成 PIN 数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GeneratePinData。

re-encrypt-data

以下代码示例演示了如何使用 re-encrypt-data。

Amazon CLI

使用不同的密钥重新加密数据

以下 re-encrypt-data 示例对使用 AES 对称密钥加密的密码文本进行解密，并使用每笔交易派生唯一密钥（DUKPT）密钥对其进行重新加密。

```
aws payment-cryptography-data re-encrypt-data \
    --incoming-key-identifier arn:aws:payment-cryptography:us-
west-2:111122223333:key/hyvv7ymboitd4vfy \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-
west-2:111122223333:key/jl6ythkcvzesbxen \
    --cipher-
text 4D2B0BDBA192D5AEFEAA5B3EC28E4A65383C313FFA25140101560F75FE1B99F27192A90980AB9334
 \
    --incoming-encryption-
attributes "Dukpt={Mode=ECB,KeySerialNumber=0123456789111111}" \
    --outgoing-encryption-attributes '{"Symmetric": {"Mode": "ECB"}}'
```

输出：

```
{
```

```
    "CipherText":
 "F94959DA30EEFF0C035483C6067667CF6796E3C1AD28C2B61F9CFEB772A8DD41C0D6822931E0D3B1",
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
jl6ythkcvzesbxen",
    "KeyCheckValue": "2E8CD9"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的加密和解密数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReEncryptData。

translate-pin-data

以下代码示例演示了如何使用 translate-pin-data。

Amazon CLI

转换 PIN 数据

以下 translate-pin-data 示例将使用 ISO 0 PIN 块将 PIN 从 PEK TDES 加密转换为使用 DUKPT 算法的 AES ISO 4 PIN 块。

```
aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block "AC17DC148BDA645E" \
    --incoming-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}' \
    --incoming-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/4pmyquwjs3yj4vwe \
    --outgoing-translation-attributes
 IsoFormat4="{PrimaryAccountNumber=171234567890123}" \
    --outgoing-dukpt-attributes KeySerialNumber="FFFF9876543210E00008"
```

输出：

```
{
    "PinBlock": "1F4209C670E49F83E75CC72E81B787D9",
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt
    "KeyCheckValue": "7CC9E2"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的转换 PIN 数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TranslatePinData。

verify-auth-request-cryptogram

以下代码示例演示了如何使用 verify-auth-request-cryptogram。

Amazon CLI

验证身份验证请求

以下 verify-auth-request-cryptogram 示例验证授权请求密码（ARQC）。

```
aws payment-cryptography-data verify-auth-request-cryptogram \
    --auth-request-cryptogram F6E1BD1E6037FB3E \
    --auth-response-attributes '{"ArpcMethod1": {"AuthResponseCode": "1111"}}' \
    --key-identifier arn:aws:payment-cryptography:us-west-2:111122223333:key/
pboipdfzd4mdklya \
    --major-key-derivation-mode "EMV_OPTION_A" \
    --session-key-derivation-attributes '{"EmvCommon":
 {"ApplicationTransactionCounter": "1234","PanSequenceNumber":
 "01","PrimaryAccountNumber": "471234567890123"}}' \
    --transaction-data "123456789ABCDEF"
```

输出：

```
{
    "AuthResponseValue": "D899B8C6FBF971AA",
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
pboipdfzd4mdklya",
    "KeyCheckValue": "985792"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的验证身份验证请求（ARQC）密码。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyAuthRequestCryptogram。

verify-card-validation-data

以下代码示例演示了如何使用 verify-card-validation-data。

Amazon CLI

验证 CVV

以下 verify-card-validation-data 示例验证了 PAN 的 CVV/CVV2。

```
aws payment-cryptography-data verify-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
    --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
    --validation-data 801
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1"
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的验证卡数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyCardValidationData。

verify-mac

以下代码示例演示了如何使用 verify-mac。

Amazon CLI

验证 MAC

以下 verify-mac 示例使用 HMAC_SHA256 算法和 HMAC 加密密钥验证用于卡数据身份验证的 HMAC 散列消息认证码（HMAC）。

```
aws payment-cryptography-data verify-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
    --message-
data "3b343038383439303031303733393431353d323430383232363030303730303f33" \
```

```
    --verification-attributes='Algorithm=HMAC_SHA256' \
    --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C
```

输出：

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6,
    "KeyCheckValue": "2976E7",
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的验证 MAC。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyMac。

verify-pin-data

以下代码示例演示了如何使用 verify-pin-data。

Amazon CLI

验证 PIN

以下 verify-pin-data 示例验证了 PAN 的 PIN。

```
aws payment-cryptography-data verify-pin-data \
    --verification-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2 \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --primary-account-number 171234567890123 \
    --pin-block-format ISO_FORMAT_0 \
    --verification-attributes
 VisaPin="{PinVerificationKeyIndex=1,VerificationValue=5507}" \
    --encrypted-pin-block AC17DC148BDA645E
```

输出：

```
{
    "VerificationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
```

```
    "VerificationKeyCheckValue": "7F2363",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
}
```

有关更多信息，请参阅《Amazon 支付加密用户指南》中的验证 PIN 数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyPinData。

使用 Amazon CLI 的 Amazon Pinpoint 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Pinpoint 结合使用，以执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-app`

以下代码示例演示了如何使用 `create-app`。

Amazon CLI

示例 1：创建应用程序

以下 `create-app` 示例创建一个新的应用程序（项目）。

```
aws pinpoint create-app \
    --create-application-request Name=ExampleCorp
```

输出：

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {}
    }
}
```

示例 2：创建带有标签的应用程序

以下 create-app 示例创建一个新的应用程序（项目），并将标签（键和值）与该应用程序关联。

```
aws pinpoint create-app \
    --create-application-request Name=ExampleCorp,tags={"Stack"="Test"}
```

输出：

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {
            "Stack": "Test"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateApp。

create-sms-template

以下代码示例演示了如何使用 create-sms-template。

Amazon CLI

为通过短信渠道发送的消息创建消息模板

以下 create-sms-template 示例创建了一个短信模板。

```
aws pinpoint create-sms-template \
    --template-name TestTemplate \
    --sms-template-request file://myfile.json \
    --region us-east-1
```

的内容myfile.json：

```
{
    "Body": "hello\n how are you?\n food is good",
    "TemplateDescription": "Test SMS Template"
}
```

输出：

```
{
    "CreateTemplateMessageBody": {
        "Arn": "arn:aws:mobiletargeting:us-east-1:AIDACKCEVSQ6C2EXAMPLE:templates/
TestTemplate/SMS",
        "Message": "Created",
        "RequestID": "8c36b17f-a0b0-400f-ac21-29e9b62a975d"
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint 消息模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSmsTemplate。

delete-app

以下代码示例演示了如何使用 delete-app。

Amazon CLI

删除应用程序

以下 delete-app 示例删除一个应用程序（项目）。

```
aws pinpoint delete-app \
    --application-id 810c7aab86d42fb2b56c8c966example
```

输出：

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {}
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteApp。

get-apns-channel

以下代码示例演示了如何使用 get-apns-channel。

Amazon CLI

检索有关应用程序的 APN 渠道的状态和设置的信息

以下 get-apns-channel 示例检索应用程序的 APN 渠道的状态和设置的信息。

```
aws pinpoint get-apns-channel \
    --application-id 9ab1068eb0a6461c86cce7f27ce0efd7 \
    --region us-east-1
```

输出：

```
{
    "APNSChannelResponse": {
        "ApplicationId": "9ab1068eb0a6461c86cce7f27ce0efd7",
        "CreationDate": "2019-05-09T21:54:45.082Z",
        "DefaultAuthenticationMethod": "CERTIFICATE",
        "Enabled": true,
        "HasCredential": true,
        "HasTokenKey": false,
        "Id": "apns",
        "IsArchived": false,
        "LastModifiedDate": "2019-05-09T22:04:01.067Z",
```

```
        "Platform": "APNS",
        "Version": 2
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApnsChannel。

get-app

以下代码示例演示了如何使用 get-app。

Amazon CLI

检索有关应用程序（项目）的信息

以下 get-app 示例检索有关应用程序（项目）的信息。

```
aws pinpoint get-app \
    --application-id 810c7aab86d42fb2b56c8c966example \
    --region us-east-1
```

输出：

```
{
    "ApplicationResponse": {
        "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {
                "Year": "2019",
                "Stack": "Production"
            }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApp。

get-apps

以下代码示例演示了如何使用 get-apps。

Amazon CLI

检索所有应用程序的相关信息

以下 get-apps 示例检索有关所有应用程序（项目）的信息。

```
aws pinpoint get-apps
```

输出：

```
{
    "ApplicationsResponse": {
        "Item": [
            {
                "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
                "Id": "810c7aab86d42fb2b56c8c966example",
                "Name": "ExampleCorp",
                "tags": {
                    "Year": "2019",
                    "Stack": "Production"
                }
            },
            {
                "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/42d8c7eb0990a57ba1d5476a3example",
                "Id": "42d8c7eb0990a57ba1d5476a3example",
                "Name": "AnyCompany",
                "tags": {}
            },
            {
                "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/80f5c382b638ffe5ad12376bbexample",
                "Id": "80f5c382b638ffe5ad12376bbexample",
                "Name": "ExampleCorp_Test",
                "tags": {
                    "Year": "2019",
                    "Stack": "Test"
                }
            }
        ],
        "NextToken":
 "eyJDcmVhdGlvbkRhdGUiOiIyMDE5LTA3LTE2VDE0OjM4OjUzLjkwM1oiLCJBY2NvdW50SWQiOiI1MTIzOTcxODM4Nz
```

```
    }
}
```

NextToken 响应值的存在表明还有更多可用的输出。再次调用该命令，并提供该值作为 NextToken 输入参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetApps。

get-campaign

以下代码示例演示了如何使用 get-campaign。

Amazon CLI

检索有关活动的状态、配置和其他设置的信息

以下 get-campaign 示例检索有关活动的状态、配置和其他设置的信息。

```
aws pinpoint get-campaign \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --campaign-id a1e63c6cc0eb43ed826ffcc3cc90b30d \
    --region us-east-1
```

输出：

```
{
    "CampaignResponse": {
        "AdditionalTreatments": [],
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/campaigns/
a1e63c6cc0eb43ed826ffcc3cc90b30d",
        "CreationDate": "2019-10-08T18:40:16.581Z",
        "Description": " ",
        "HoldoutPercent": 0,
        "Id": "a1e63c6cc0eb43ed826ffcc3cc90b30d",
        "IsPaused": false,
        "LastModifiedDate": "2019-10-08T18:40:16.581Z",
        "Limits": {
            "Daily": 0,
            "MaximumDuration": 60,
            "MessagesPerSecond": 50,
            "Total": 0
```

```
        },
        "MessageConfiguration": {
            "EmailMessage": {
                "FromAddress": "sender@example.com",
                "HtmlBody": "<!DOCTYPE html>\n  <html lang=\"en\">\n    <head>\n
 <meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\" />\n</head>
\n<body>Hello</body>\n</html>",
                "Title": "PinpointDemo"
            }
        },
        "Name": "MyCampaign",
        "Schedule": {
            "IsLocalTime": false,
            "StartTime": "IMMEDIATE",
            "Timezone": "utc"
        },
        "SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
        "SegmentVersion": 1,
        "State": {
            "CampaignStatus": "COMPLETED"
        },
        "tags": {},
        "TemplateConfiguration": {},
        "Version": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCampaign。

get-campaigns

以下代码示例演示了如何使用 get-campaigns。

Amazon CLI

检索与应用程序关联的所有活动的状态、配置和其他设置的相关信息

以下 get-campaigns 示例检索与应用程序关联的所有活动的状态、配置和其他设置的相关信息。

```
aws pinpoint get-campaigns \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：


```
{
    "CampaignsResponse": {
        "Item": [
            {
                "AdditionalTreatments": [],
                "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
                "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/
campaigns/7e1280344c8f4a9aa40a00b006fe44f1",
                "CreationDate": "2019-10-08T18:40:22.905Z",
                "Description": " ",
                "HoldoutPercent": 0,
                "Id": "7e1280344c8f4a9aa40a00b006fe44f1",
                "IsPaused": false,
                "LastModifiedDate": "2019-10-08T18:40:22.905Z",
                "Limits": {},
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": "sender@example.com",
                        "HtmlBody": "<!DOCTYPE html>\n    <html lang=\"en
\">\n    <head>\n    <meta http-equiv=\"Content-Type\" content=\"text/html;
 charset=utf-8\" />\n</head>\n<body>Hello</body>\n</html>",
                        "Title": "PInpointDemo Test"
                    }
                },
                "Name": "MyCampaign1",
                "Schedule": {
                    "IsLocalTime": false,
                    "QuietTime": {},
                    "StartTime": "IMMEDIATE",
                    "Timezone": "UTC"
                },
                "SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
                "SegmentVersion": 1,
                "State": {
                    "CampaignStatus": "COMPLETED"
                },
                "tags": {},
                "TemplateConfiguration": {},
                "Version": 1
            },
            {
```

```
                "AdditionalTreatments": [],
                "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
                "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/campaigns/
a1e63c6cc0eb43ed826ffcc3cc90b30d",
                "CreationDate": "2019-10-08T18:40:16.581Z",
                "Description": " ",
                "HoldoutPercent": 0,
                "Id": "a1e63c6cc0eb43ed826ffcc3cc90b30d",
                "IsPaused": false,
                "LastModifiedDate": "2019-10-08T18:40:16.581Z",
                "Limits": {
                    "Daily": 0,
                    "MaximumDuration": 60,
                    "MessagesPerSecond": 50,
                    "Total": 0
                },
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": "sender@example.com",
                        "HtmlBody": "<!DOCTYPE html>\n    <html lang=\"en
\">\n    <head>\n    <meta http-equiv=\"Content-Type\" content=\"text/html;
 charset=utf-8\" />\n</head>\n<body>Demo</body>\n</html>",
                        "Title": "PinpointDemo"
                    }
                },
                "Name": "MyCampaign2",
                "Schedule": {
                    "IsLocalTime": false,
                    "StartTime": "IMMEDIATE",
                    "Timezone": "utc"
                },
                "SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
                "SegmentVersion": 1,
                "State": {
                    "CampaignStatus": "COMPLETED"
                },
                "tags": {},
                "TemplateConfiguration": {},
                "Version": 1
            }
        ]
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCampaigns。

get-channels

以下代码示例演示了如何使用 get-channels。

Amazon CLI

检索有关应用程序的每个渠道的历史和状态的信息

以下 get-channels 示例检索有关应用程序的每个渠道的历史和状态的信息。

```
aws pinpoint get-channels \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：

```
{
    "ChannelsResponse": {
        "Channels": {
            "GCM": {
                "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
                "CreationDate": "2019-10-08T18:28:23.182Z",
                "Enabled": true,
                "HasCredential": true,
                "Id": "gcm",
                "IsArchived": false,
                "LastModifiedDate": "2019-10-08T18:28:23.182Z",
                "Version": 1
            },
            "SMS": {
                "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
                "CreationDate": "2019-10-08T18:39:18.511Z",
                "Enabled": true,
                "Id": "sms",
                "IsArchived": false,
                "LastModifiedDate": "2019-10-08T18:39:18.511Z",
                "Version": 1
            },
```

```
            "EMAIL": {
                "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
                "CreationDate": "2019-10-08T18:27:23.990Z",
                "Enabled": true,
                "Id": "email",
                "IsArchived": false,
                "LastModifiedDate": "2019-10-08T18:27:23.990Z",
                "Version": 1
            },
            "IN_APP": {
                "Enabled": true,
                "IsArchived": false,
                "Version": 0
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetChannels。

get-email-channel

以下代码示例演示了如何使用 get-email-channel。

Amazon CLI

检索有关应用程序的电子邮件渠道的状态和设置的信息

以下 get-email-channel 示例检索应用程序的电子邮件渠道的状态和设置。

```
aws pinpoint get-email-channel \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：

```
{
    "EmailChannelResponse": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:27:23.990Z",
        "Enabled": true,
        "FromAddress": "sender@example.com",
        "Id": "email",
```

```
        "Identity": "arn:aws:ses:us-east-1:AIDACKCEVSQ6C2EXAMPLE:identity/
sender@example.com",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:27:23.990Z",
        "MessagesPerSecond": 1,
        "Platform": "EMAIL",
        "RoleArn": "arn:aws:iam::AIDACKCEVSQ6C2EXAMPLE:role/pinpoint-events",
        "Version": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEmailChannel。

get-endpoint

以下代码示例演示了如何使用 get-endpoint。

Amazon CLI

检索有关应用程序特定端点的设置和属性的信息

以下 get-endpoint 示例检索有关应用程序特定端点的设置和属性的信息。

```
aws pinpoint get-endpoint \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --endpoint-id testendpoint \
    --region us-east-1
```

输出：

```
{
    "EndpointResponse": {
        "Address": "+11234567890",
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "Attributes": {},
        "ChannelType": "SMS",
        "CohortId": "63",
        "CreationDate": "2019-01-28T23:55:11.534Z",
        "EffectiveDate": "2021-08-06T00:04:51.763Z",
        "EndpointStatus": "ACTIVE",
        "Id": "testendpoint",
        "Location": {
```

```
            "Country": "USA"
        },
        "Metrics": {
            "SmsDelivered": 1.0
        },
        "OptOut": "ALL",
        "RequestId": "a204b1f2-7e26-48a7-9c80-b49a2143489d",
        "User": {
            "UserAttributes": {
                "Age": [
                    "24"
                ]
            },
        "UserId": "testuser"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEndpoint。

get-gcm-channel

以下代码示例演示了如何使用 get-gcm-channel。

Amazon CLI

检索有关应用程序的 GCM 渠道的状态和设置的信息

以下 get-gcm-channel 示例检索应用程序的 GCM 渠道的状态和设置的信息。

```
aws pinpoint get-gcm-channel \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：

```
{
    "GCMChannelResponse": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:28:23.182Z",
        "Enabled": true,
        "HasCredential": true,
```

```
        "Id": "gcm",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:28:23.182Z",
        "Platform": "GCM",
        "Version": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGcmChannel。

get-sms-channel

以下代码示例演示了如何使用 get-sms-channel。

Amazon CLI

检索有关应用程序的短信渠道的状态和设置的信息

以下 get-sms-channel 示例检索应用程序的短信渠道的状态和设置。

```
aws pinpoint get-sms-channel \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：

```
{
    "SMSChannelResponse": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:39:18.511Z",
        "Enabled": true,
        "Id": "sms",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:39:18.511Z",
        "Platform": "SMS",
        "PromotionalMessagesPerSecond": 20,
        "TransactionalMessagesPerSecond": 20,
        "Version": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSmsChannel。

get-sms-template

以下代码示例演示了如何使用 get-sms-template。

Amazon CLI

检索通过短信渠道发送的消息对应的消息模板的内容和设置

以下 get-sms-template 示例检索短信模板的内容和设置。

```
aws pinpoint get-sms-template \
    --template-name TestTemplate \
    --region us-east-1
```

输出：

```
{
    "SMSTemplateResponse": {
        "Arn": "arn:aws:mobiletargeting:us-east-1:AIDACKCEVSQ6C2EXAMPLE:templates/
TestTemplate/SMS",
        "Body": "hello\n how are you?\n food is good",
        "CreationDate": "2023-06-20T21:37:30.124Z",
        "LastModifiedDate": "2023-06-20T21:37:30.124Z",
        "tags": {},
        "TemplateDescription": "Test SMS Template",
        "TemplateName": "TestTemplate",
        "TemplateType": "SMS",
        "Version": "1"
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint 消息模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSmsTemplate。

get-voice-channel

以下代码示例演示了如何使用 get-voice-channel。

Amazon CLI

检索有关应用程序的语音渠道的状态和设置的信息

以下 get-voice-channel 示例检索应用程序的语音渠道的状态和设置。

```
aws pinpoint get-voice-channel \
    --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
    --region us-east-1
```

输出：

```
{
    "VoiceChannelResponse": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2022-04-28T00:17:03.836Z",
        "Enabled": true,
        "Id": "voice",
        "IsArchived": false,
        "LastModifiedDate": "2022-04-28T00:17:03.836Z",
        "Platform": "VOICE",
        "Version": 1
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVoiceChannel。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

检索资源的标签列表

以下 list-tags-for-resource 示例检索与指定资源关联的所有标签（键名称和值）。

```
aws pinpoint list-tags-for-resource \
    --resource-arn arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example
```

输出：

```
{
    "TagsModel": {
```

```
        "tags": {
            "Year": "2019",
            "Stack": "Production"
        }
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 开发人员指南》中的“为 Amazon Pinpoint 资源添加标签<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

phone-number-validate

以下代码示例演示了如何使用 phone-number-validate。

Amazon CLI

检索有关电话号码的信息

以下 phone-number-validate 示例检索有关电话号码的信息。

```
aws pinpoint phone-number-validate \
    --number-validate-request PhoneNumber="+12065550142" \
    --region us-east-1
```

输出：

```
{
    "NumberValidateResponse": {
        "Carrier": "ExampleCorp Mobile",
        "City": "Seattle",
        "CleansedPhoneNumberE164": "+12065550142",
        "CleansedPhoneNumberNational": "2065550142",
        "Country": "United States",
        "CountryCodeIso2": "US",
        "CountryCodeNumeric": "1",
        "OriginalPhoneNumber": "+12065550142",
        "PhoneType": "MOBILE",
        "PhoneTypeCode": 0,
        "Timezone": "America/Los_Angeles",
        "ZipCode": "98101"
```

```
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint SMS 渠道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PhoneNumberValidate。

send-messages

以下代码示例演示了如何使用 send-messages。

Amazon CLI

使用应用程序的端点发送短信

以下 send-messages 示例通过端点为应用程序发送直接消息。

```
aws pinpoint send-messages \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --message-request file://myfile.json \
    --region us-west-2
```

的内容myfile.json：

```
{
    "MessageConfiguration": {
        "SMSMessage": {
            "Body": "hello, how are you?"
        }
    },
    "Endpoints": {
        "testendpoint": {}
    }
}
```

输出：

```
{
    "MessageResponse": {
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "EndpointResult": {
```

```
            "testendpoint": {
                "Address": "+12345678900",
                "DeliveryStatus": "SUCCESSFUL",
                "MessageId": "itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0",
                "StatusCode": 200,
                "StatusMessage": "MessageId:
 itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0"
            }
        },
        "RequestId": "c7e23264-04b2-4a46-b800-d24923f74753"
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint SMS 渠道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendMessages。

send-users-messages

以下代码示例演示了如何使用 send-users-messages。

Amazon CLI

向应用程序的用户发送短信

以下 send-users-messages 示例向应用程序用户发送私信。

```
aws pinpoint send-users-messages \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --send-users-message-request file://myfile.json \
    --region us-west-2
```

的内容myfile.json：

```
{
    "MessageConfiguration": {
        "SMSMessage": {
            "Body": "hello, how are you?"
        }
    },
    "Users": {
        "testuser": {}
```

```
    }
}
```

输出：

```
{
    "SendUsersMessageResponse": {
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "RequestId": "e0b12cf5-2359-11e9-bb0b-d5fb91876b25",
        "Result": {
            "testuser": {
                "testuserendpoint": {
                    "DeliveryStatus": "SUCCESSFUL",
                    "MessageId": "7qu4hk5bqhda3i7i2n4pjf98qcuh8b7p45ifsmo0",
                    "StatusCode": 200,
                    "StatusMessage": "MessageId:
 7qu4hk5bqhda3i7i2n4pjf98qcuh8b7p45ifsmo0",
                    "Address": "+12345678900"
                }
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint SMS 渠道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendUsersMessages。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源添加标签

以下示例向资源添加两个标签（键名和值）。

```
aws pinpoint list-tags-for-resource \
    --resource-arn arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \
    --tags-model tags={Stack=Production,Year=2019}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Pinpoint 开发人员指南》中的“为 Amazon Pinpoint 资源添加标签<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

示例 1：从资源中删除标签

以下 untag-resource 示例从资源中删除指定的标签（键名和值）。

```
aws pinpoint untag-resource \
    --resource-arn arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \
    --tag-keys Year
```

此命令不生成任何输出。

示例 2：从资源中删除多个标签

以下 untag-resource 示例从资源中删除指定的标签（键名和值）。

```
aws pinpoint untag-resource \
    --resource-arn arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \
    --tag-keys Year Stack
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Pinpoint 开发人员指南》中的“为 Amazon Pinpoint 资源添加标签<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-sms-channel

以下代码示例演示了如何使用 update-sms-channel。

Amazon CLI

启用短信渠道或更新应用程序短信通道的状态和设置。

以下 update-sms-channel 示例启用了应用程序的短信渠道。

```
aws pinpoint update-sms-channel \
    --application-id 611e3e3cdd47474c9c1399a505665b91 \
    --sms-channel-request Enabled=true \
    --region us-west-2
```

输出：

```
{
    "SMSChannelResponse": {
        "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
        "CreationDate": "2019-01-28T23:25:25.224Z",
        "Enabled": true,
        "Id": "sms",
        "IsArchived": false,
        "LastModifiedDate": "2023-05-18T23:22:50.977Z",
        "Platform": "SMS",
        "PromotionalMessagesPerSecond": 20,
        "TransactionalMessagesPerSecond": 20,
        "Version": 3
    }
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 Amazon Pinpoint SMS 渠道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSmsChannel。

使用 Amazon CLI 的 Amazon Polly 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Polly 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

delete-lexicon

以下代码示例演示了如何使用 delete-lexicon。

Amazon CLI

删除词典

以下 delete-lexicon 示例删除指定的词典。

```
aws polly delete-lexicon \
    --name w3c
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的使用 DeleteLexicon 操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLexicon。

get-lexicon

以下代码示例演示了如何使用 get-lexicon。

Amazon CLI

检索词典的内容

以下 get-lexicon 示例检索指定的发音词典内容。

```
aws polly get-lexicon \
    --name w3c
```

输出：

```
{
```

```
    "Lexicon": {
        "Content": "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<lexicon version=
\"1.0\" \n      xmlns=    \"http://www.w3.org/2005/01/pronunciation-lexicon
\"\n      xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n
 xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon \n
 http://www.w3.org/TR/2007/CR-pronunciation-    lexicon-20071212/pls.xsd\"\n
   alphabet=\"ipa\" \n      xml:lang=\"en-US\">\n  <lexeme>\n    <grapheme>W3C</
grapheme>\n        <alias>World Wide Web Consortium</alias>\n  </lexeme>\n</lexicon>
\n",
        "Name": "w3c"
    },
    "LexiconAttributes": {
        "Alphabet": "ipa",
        "LanguageCode": "en-US",
        "LastModified": 1603908910.99,
        "LexiconArn": "arn:aws:polly:us-west-2:880185128111:lexicon/w3c",
        "LexemesCount": 1,
        "Size": 492
    }
}
```

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的使用 GetLexicon 操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLexicon。

get-speech-synthesis-task

以下代码示例演示了如何使用 get-speech-synthesis-task。

Amazon CLI

获取有关语音合成任务的信息

以下 get-speech-synthesis-task 示例检索有关指定语音合成任务的信息。

```
aws polly get-speech-synthesis-task \
    --task-id 70b61c0f-57ce-4715-a247-cae8729dcce9
```

输出：

```
{
    "SynthesisTask": {
        "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
```

```
        "TaskStatus": "completed",
        "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}
```

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的创建长音频文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSpeechSynthesisTask。

list-lexicons

以下代码示例演示了如何使用 list-lexicons。

Amazon CLI

列出您的词典

以下 list-lexicons 示例列出了您的发音词典。

```
aws polly list-lexicons
```

输出：

```
{
    "Lexicons": [
        {
            "Name": "w3c",
            "Attributes": {
                "Alphabet": "ipa",
                "LanguageCode": "en-US",
                "LastModified": 1603908910.99,
                "LexiconArn": "arn:aws:polly:us-east-2:123456789012:lexicon/w3c",
                "LexemesCount": 1,
                "Size": 492
            }
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的使用 ListLexicons 操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLexicons。

`list-speech-synthesis-tasks`

以下代码示例演示了如何使用 `list-speech-synthesis-tasks`。

Amazon CLI

列出您的语音合成任务

以下 `list-speech-synthesis-tasks` 示例列出了您的语音合成任务。

```
aws polly list-speech-synthesis-tasks
```

输出：

```
{
    "SynthesisTasks": [
        {
            "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
            "TaskStatus": "completed",
            "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
            "CreationTime": 1603911042.689,
            "RequestCharacters": 1311,
            "OutputFormat": "mp3",
            "TextType": "text",
            "VoiceId": "Joanna"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的创建长音频文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSpeechSynthesisTasks。

`put-lexicon`

以下代码示例演示了如何使用 `put-lexicon`。

Amazon CLI

存储词典

以下 put-lexicon 示例存储指定的发音词典。example.pls 文件指定符合 W3C PLS 标准的词典。

```
aws polly put-lexicon \
    --name w3c \
    --content file://example.pls
```

example.pls 的内容

```
{
    <?xml version="1.0" encoding="UTF-8"?>
    <lexicon version="1.0"
        xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
            http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
        alphabet="ipa"
        xml:lang="en-US">
        <lexeme>
            <grapheme>W3C</grapheme>
            <alias>World Wide Web Consortium</alias>
        </lexeme>
    </lexicon>
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的使用 PutLexicon 操作。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLexicon。

start-speech-synthesis-task

以下代码示例演示了如何使用 start-speech-synthesis-task。

Amazon CLI

合成文本

以下 start-speech-synthesis-task 示例合成了 text_file.txt 中的文本，并将生成的 MP3 文件存储在指定的存储桶中。

```
aws polly start-speech-synthesis-task \
    --output-format mp3 \
    --output-s3-bucket-name amzn-s3-demo-bucket \
    --text  file://text_file.txt \
    --voice-id Joanna
```

输出：

```
{
    "SynthesisTask": {
        "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
        "TaskStatus": "scheduled",
        "OutputUri": "https://s3.us-east-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}
```

有关更多信息，请参阅《Amazon Polly 开发人员指南》中的创建长音频文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSpeechSynthesisTask。

Amazon 价目表使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon 价目表 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-services`

以下代码示例演示了如何使用 `describe-services`。

Amazon CLI

检索服务元数据

此示例检索 Amazon EC2 服务代码的元数据。

命令:

```
aws pricing describe-services --service-code AmazonEC2 --format-version aws_v1 --
max-items 1
```

输出：

```
{
  "Services": [
      {
          "ServiceCode": "AmazonEC2",
          "AttributeNames": [
              "volumeType",
              "maxIopsvolume",
              "instance",
              "instanceCapacity10xlarge",
              "locationType",
              "instanceFamily",
              "operatingSystem",
              "clockSpeed",
              "LeaseContractLength",
              "ecu",
              "networkPerformance",
              "instanceCapacity8xlarge",
              "group",
              "maxThroughputvolume",
              "gpuMemory",
              "ebsOptimized",
              "elasticGpuType",
```

```
                "maxVolumeSize",
                "gpu",
                "processorFeatures",
                "intelAvxAvailable",
                "instanceCapacity4xlarge",
                "servicecode",
                "groupDescription",
                "processorArchitecture",
                "physicalCores",
                "productFamily",
                "enhancedNetworkingSupported",
                "intelTurboAvailable",
                "memory",
                "dedicatedEbsThroughput",
                "vcpu",
                "OfferingClass",
                "instanceCapacityLarge",
                "capacitystatus",
                "termType",
                "storage",
                "intelAvx2Available",
                "storageMedia",
                "physicalProcessor",
                "provisioned",
                "servicename",
                "PurchaseOption",
                "instanceCapacity18xlarge",
                "instanceType",
                "tenancy",
                "usagetype",
                "normalizationSizeFactor",
                "instanceCapacity2xlarge",
                "instanceCapacity16xlarge",
                "maxIopsBurstPerformance",
                "instanceCapacity12xlarge",
                "instanceCapacity32xlarge",
                "instanceCapacityXlarge",
                "licenseModel",
                "currentGeneration",
                "preInstalledSw",
                "location",
                "instanceCapacity24xlarge",
                "instanceCapacity9xlarge",
                "instanceCapacityMedium",
```

```
                "operation"
            ]
        }
    ],
    "FormatVersion": "aws_v1"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeServices。

get-attribute-values

以下代码示例演示了如何使用 get-attribute-values。

Amazon CLI

检索属性值列表

以下 get-attribute-values 示例检索给定属性的可用值列表。

```
aws pricing get-attribute-values \
    --service-code AmazonEC2 \
    --attribute-name volumeType \
    --max-items 2
```

输出：

```
{
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ==",
    "AttributeValues": [
        {
            "Value": "Cold HDD"
        },
        {
            "Value": "General Purpose"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAttributeValues。

get-products

以下代码示例演示了如何使用 get-products。

Amazon CLI

检索产品列表

此示例检索与给定条件匹配的产品的列表。

命令:

```
aws pricing get-products --filters file://filters.json --format-version aws_v1 --
max-results 1 --service-code AmazonEC2
```

filters.json：

```
      [
  {
    "Type": "TERM_MATCH",
    "Field": "ServiceCode",
    "Value": "AmazonEC2"
  },
  {
    "Type": "TERM_MATCH",
    "Field": "volumeType",
    "Value": "Provisioned IOPS"
  }
]
```

输出：

```
{
  "FormatVersion": "aws_v1",
  "NextToken": "WGDY7ko8fQXdlaUZVdasFQ==:RVSagyIFn770XQOzdUIcO9BY6ucBG9itXAZGZF/
zioUzOsUKh6PCcPWaOyPZRiMePb986TeoKYB9l55fw/
CyoMq5ymnGmT1Vj39TljbbAlhcqnVfTmPIilx8Uy5bdDaBYy/e/2Ofw9EdzsykbS8LTBuNbiDQ
+BBds5yeI9AQkUepruKk3aEahFPxJ55kx/zk",
  "PriceList": [
      "{\"product\":{\"productFamily\":\"Storage\",\"attributes\":{\"storageMedia\":
\"SSD-backed\",\"maxThroughputvolume\":\"320 MB/sec\",\"volumeType\":\"Provisioned
 IOPS\",\"maxIopsvolume\":\"20000\",\"servicecode\":\"AmazonEC2\",\"usagetype
```

```
\":\"APS1-EBS:VolumeUsage.piops\",\"locationType\":\"AWS Region\",\"location\":
\"Asia Pacific (Singapore)\",\"servicename\":\"Amazon Elastic Compute Cloud\",
\"maxVolumeSize\":\"16 TiB\",\"operation\":\"\"},\"sku\":\"3MKHN58N7RDDVGKJ\"},
\"serviceCode\":\"AmazonEC2\",\"terms\":{\"OnDemand\":{\"3MKHN58N7RDDVGKJ.JRTCKXETXF
\":{\"priceDimensions\":{\"3MKHN58N7RDDVGKJ.JRTCKXETXF.6YS6EN2CT7\":{\"unit\":\"GB-
Mo\",\"endRange\":\"Inf\",\"description\":\"$0.138 per GB-month of Provisioned
 IOPS SSD (io1)  provisioned storage - Asia Pacific (Singapore)\",\"appliesTo
\":[],\"rateCode\":\"3MKHN58N7RDDVGKJ.JRTCKXETXF.6YS6EN2CT7\",\"beginRange\":
\"0\",\"pricePerUnit\":{\"USD\":\"0.1380000000\"}}},\"sku\":\"3MKHN58N7RDDVGKJ
\",\"effectiveDate\":\"2018-08-01T00:00:00Z\",\"offerTermCode\":\"JRTCKXETXF
\",\"termAttributes\":{}}}},\"version\":\"20180808005701\",\"publicationDate\":
\"2018-08-08T00:57:01Z\"}"
  ]
}
```


- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProducts。

使用 Amazon CLI 的 Amazon 私有 CA 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon 私有 CA 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-certificate-authority-audit-report`

以下代码示例演示了如何使用 `create-certificate-authority-audit-report`。

Amazon CLI

创建证书颁发机构审计报告

以下 create-certificate-authority-audit-report 命令为由 ARN 标识的私有 CA 创建审计报告。

```
aws acm-pca create-certificate-authority-audit-report --certificate-
authority-arn arn:aws:acm-pca:us-east-1:accountid:certificate-
authority/12345678-1234-1234-1234-123456789012 --s3-bucket-name your-bucket-name --
audit-report-response-format JSON
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCertificateAuthorityAuditReport。

create-certificate-authority

以下代码示例演示了如何使用 create-certificate-authority。

Amazon CLI

创建私有证书颁发机构

以下 create-certificate-authority 命令在您的 Amazon 账户中创建私有证书颁发机构。

```
aws acm-pca create-certificate-authority --certificate-authority-configuration
 file://C:\ca_config.txt --revocation-configuration file://C:\revoke_config.txt --
certificate-authority-type "SUBORDINATE" --idempotency-token 98256344
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCertificateAuthority。

delete-certificate-authority

以下代码示例演示了如何使用 delete-certificate-authority。

Amazon CLI

删除私有证书颁发机构

以下 delete-certificate-authority 命令删除由 ARN 标识的证书颁发机构。

```
aws acm-pca delete-certificate-authority --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCertificateAuthority。

describe-certificate-authority-audit-report

以下代码示例演示了如何使用 describe-certificate-authority-audit-report。

Amazon CLI

描述证书颁发机构的审计报告

以下 describe-certificate-authority-audit-report 命令列出有关由 ARN 标识的 CA 的指定审计报告的信息。

```
aws acm-pca describe-certificate-authority-audit-report --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/99999999-8888-7777-6666-555555555555 --audit-report-
id 11111111-2222-3333-4444-555555555555
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificateAuthorityAuditReport。

describe-certificate-authority

以下代码示例演示了如何使用 describe-certificate-authority。

Amazon CLI

描述私有证书颁发机构

以下 describe-certificate-authority 命令列出有关由 ARN 标识的私有 CA 的信息。

```
aws acm-pca describe-certificate-authority --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificateAuthority。

get-certificate-authority-certificate

以下代码示例演示了如何使用 get-certificate-authority-certificate。

Amazon CLI

检索证书颁发机构（CA）证书

以下 get-certificate-authority-certificate 命令检索由 ARN 指定的私有 CA 的证书和证书链。

```
aws acm-pca get-certificate-authority-certificate --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --output text
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCertificateAuthorityCertificate。

get-certificate-authority-csr

以下代码示例演示了如何使用 get-certificate-authority-csr。

Amazon CLI

检索证书颁发机构的证书签名请求

以下 get-certificate-authority-csr 命令检索由 ARN 指定的私有 CA 的 CSR。

```
aws acm-pca get-certificate-authority-csr --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --output text
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCertificateAuthorityCsr。

get-certificate

以下代码示例演示了如何使用 get-certificate。

Amazon CLI

检索已签发的证书

以下 get-certificate 示例检索指定私有 CA 的证书。

```
aws acm-pca get-certificate \
    --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 \
```

```
    --certificate-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012/
certificate/6707447683a9b7f4055627ffd55cebcc \
    --output text
```

输出：

```
-----BEGIN CERTIFICATE-----
MIIEDzCCAvegAwIBAgIRAJuJ8f6ZVYL7gG/rS3qvrZMwDQYJKoZIhvcNAQELBQAw
cTELMAkGA1UEBhMCVVMxEzARBgNVBAgMCldhc2hpbmd0b24xEDAOBgNVBAcMB1Nl
    ....certificate body truncated for brevity....
tKCSglgZZrd4FdLw1EkGm+UVXnodwMtJEQyy3oTfZjURPIyyaqskTu/KSS7YDjK0
KQNy73D6LtmdOEbAyq10XiDxqY41lvKHJ1eZrPaBmYNABxU=
-----END CERTIFICATE---- -----BEGIN CERTIFICATE-----
MIIDrzCCApegAwIBAgIRAOskdzLvcj1eShkoyEE693AwDQYJKoZIhvcNAQELBQAw
cTELMAkGA1UEBhMCVVMxEzARBgNVBAgMCldhc2hpbmd0b24xEDAOBgNVBAcMB1Nl
    ...certificate body truncated for brevity....
kdRGB6P2hpxstDOUIwAoCbhoaWwfA4ybJznf+jOQhAziNlRdKQRR8nODWpKt7H9w
dJ5nxsTk/fniJz86Ddtp6n8s82wYdkN3cVffeK72A9aTCOU=
-----END CERTIFICATE-----
```

输出的第一部分是证书本身。第二部分是链接到根 CA 证书的证书链。请注意，使用 `--output text` 选项时，会在两个证书片段之间插入一个 TAB 字符（由于文本缩进）。如果您打算使用此输出并使用其他工具解析证书，则可能需要移除 TAB 字符，以便正确处理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCertificate。

import-certificate-authority-certificate

以下代码示例演示了如何使用 `import-certificate-authority-certificate`。

Amazon CLI

将您的证书颁发机构证书导入 ACM PCA

以下 `import-certificate-authority-certificate` 命令将由 ARN 指定的 CA 的已签名私有 CA 证书导入 ACM PCA。

```
aws acm-pca import-certificate-authority-certificate --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --certificate file://C:\ca_cert.pem
 --certificate-chain file://C:\ca_cert_chain.pem
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportCertificateAuthorityCertificate。

issue-certificate

以下代码示例演示了如何使用 issue-certificate。

Amazon CLI

签发私有证书

以下 issue-certificate 命令使用由 ARN 指定的私有 CA 来签发私有证书。

```
aws acm-pca issue-certificate --certificate-authority-arn arn:aws:acm-pca:us-
west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012
 --csr file://C:\cert_1.csr --signing-algorithm "SHA256WITHRSA" --validity
 Value=365,Type="DAYS" --idempotency-token 1234
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IssueCertificate。

list-certificate-authorities

以下代码示例演示了如何使用 list-certificate-authorities。

Amazon CLI

列出您的私有证书颁发机构

以下 list-certificate-authorities 命令列出有关您账户中所有私有 CA 的信息。

```
aws acm-pca list-certificate-authorities --max-results 10
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCertificateAuthorities。

list-tags

以下代码示例演示了如何使用 list-tags。

Amazon CLI

列出您证书颁发机构的标签

以下 list-tags 命令列出与由 ARN 指定的私人 CA 关联的标签。

```
aws acm-pca list-tags --certificate-authority-arn arn:aws:acm-pca:us-
west-2:123456789012:certificate-authority/123455678-1234-1234-1234-123456789012 --
max-results 10
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTags。

revoke-certificate

以下代码示例演示了如何使用 revoke-certificate。

Amazon CLI

撤消私有证书

以下 revoke-certificate 命令撤销由 ARN 标识的 CA 的私有证书。

```
aws acm-pca revoke-certificate --certificate-authority-arn arn:aws:acm-pca:us-
west-2:1234567890:certificate-authority/12345678-1234-1234-1234-123456789012 --
certificate-serial 67:07:44:76:83:a9:b7:f4:05:56:27:ff:d5:5c:eb:cc --revocation-
reason "KEY_COMPROMISE"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeCertificate。

tag-certificate-authority

以下代码示例演示了如何使用 tag-certificate-authority。

Amazon CLI

向私有证书颁发机构添加标签

以下 tag-certificate-authority 命令将一个或多个标签附加到您的私有 CA。

```
aws acm-pca tag-certificate-authority --certificate-authority-
arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --tags Key=Admin,Value=Alice
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagCertificateAuthority。

untag-certificate-authority

以下代码示例演示了如何使用 untag-certificate-authority。

Amazon CLI

移除您私有证书颁发机构中的一个或多个标签

以下 untag-certificate-authority 命令移除由 ARN 标识的私有 CA 中的标签。

```
aws acm-pca untag-certificate-authority --certificate-authority-
arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --tags Key=Purpose,Value=Website
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagCertificateAuthority。

update-certificate-authority

以下代码示例演示了如何使用 update-certificate-authority。

Amazon CLI

更新您私有证书颁发机构的配置

以下 update-certificate-authority 命令更新由 ARN 标识的私有 CA 的状态和配置。

```
aws acm-pca update-certificate-authority --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-1232456789012 --revocation-configuration file://C:
\revoke_config.txt --status "DISABLED"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateCertificateAuthority。

Amazon Proton使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Proton 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

cancel-service-instance-deployment

以下代码示例演示了如何使用 cancel-service-instance-deployment。

Amazon CLI

取消服务实例部署

以下 cancel-service-instance-deployment 示例取消了服务实例部署。

```
aws proton cancel-service-instance-deployment \
    --service-instance-name "instance-one" \
    --service-name "simple-svc"
```

输出：

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "CANCELLING",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2021-04-02T21:45:15.406000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:38:00.823000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_optional_input: abc\n  my_sample_pipeline_required_input:
 '123'\ninstances:\n- name: my-instance\n  environment: MySimpleEnv
\n  spec:\n    my_sample_service_instance_optional_input: def\n
 my_sample_service_instance_required_input: '456'\n- name: my-other-instance\n
 environment: MySimpleEnv\n  spec:\n    my_sample_service_instance_required_input:
 '789'\n",
```

```
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的更新服务实例或《Amazon Proton 用户指南》中的更新服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelServiceInstanceDeployment。

cancel-service-pipeline-deployment

以下代码示例演示了如何使用 cancel-service-pipeline-deployment。

Amazon CLI

取消服务管道部署

以下 cancel-service-pipeline-deployment 示例取消了服务管道部署。

```
aws proton cancel-service-pipeline-deployment \
    --service-name "simple-svc"
```

输出：

```
{
    "pipeline": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "CANCELLING",
        "lastDeploymentAttemptedAt": "2021-04-02T22:02:45.095000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:39:28.991000+00:00",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的更新服务管道或《Amazon Proton 用户指南》中的更新服务管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelServicePipelineDeployment。

create-service

以下代码示例演示了如何使用 create-service。

Amazon CLI

创建服务

以下 create-service 示例创建了一个带有服务管道的服务。

```
aws proton create-service \
    --name "MySimpleService" \
    --template-name "fargate-service" \
    --template-major-version "1" \
    --branch-name "mainline" \
    --repository-connection-arn "arn:aws:codestar-connections:region-id:account-id:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
    --repository-id "myorg/myapp" \
    --spec file://spec.yaml
```

的内容spec.yaml：

```
proton: ServiceSpec

pipeline:
    my_sample_pipeline_required_input: "hello"
    my_sample_pipeline_optional_input: "bye"

instances:
    - name: "acme-network-dev"
        environment: "ENV_NAME"
        spec:
            my_sample_service_instance_required_input: "hi"
            my_sample_service_instance_optional_input: "ho"
```

输出：

```
{
    "service": {
```

```
        "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
        "createdAt": "2020-11-18T19:50:27.460000+00:00",
        "lastModifiedAt": "2020-11-18T19:50:27.460000+00:00",
        "name": "MySimpleService",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "myorg/myapp",
        "status": "CREATE_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的创建服务或《Amazon Proton 用户指南》中的创建服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateService。

delete-service

以下代码示例演示了如何使用 delete-service。

Amazon CLI

删除服务

以下 delete-service 示例删除一项服务。

```
aws proton delete-service \
    --name "simple-svc"
```

输出：

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "mainline",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "description": "Edit by updating description",
        "lastModifiedAt": "2020-11-29T00:30:39.248000+00:00",
        "name": "simple-svc",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "myorg/myapp",
```

```
        "status": "DELETE_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的删除服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteService。

get-service-instance

以下代码示例演示了如何使用 get-service-instance。

Amazon CLI

获取服务实例详细信息

以下 get-service-instance 示例获取服务实例的详细数据。

```
aws proton get-service-instance \
    --name "instance-one" \
    --service-name "simple-svc"
```

输出：

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
        "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_optional_input: hello world\n
 my_sample_pipeline_required_input: pipeline up\ninstances:\n- name: instance-one\n
 environment: my-simple-env\n  spec:\n    my_sample_service_instance_optional_input:
 Ola\n    my_sample_service_instance_required_input: Ciao\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
```

```
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的查看服务数据或《Amazon Proton 用户指南》中的查看服务数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceInstance。

get-service

以下代码示例演示了如何使用 get-service。

Amazon CLI

获取服务详细信息

以下 get-service 示例获取服务的详细数据。

```
aws proton get-service \
    --name "simple-svc"
```

输出：

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "mainline",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "lastModifiedAt": "2020-11-28T22:44:51.207000+00:00",
        "name": "simple-svc",
        "pipeline": {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
pipeline/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "createdAt": "2020-11-28T22:40:50.512000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
            "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
            "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_required_input: hello\n  my_sample_pipeline_optional_input:
 bye\ninstances:\n- name: instance-svc-simple\n  environment: my-simple-
env\n  spec:\n    my_sample_service_instance_required_input: hi\n
 my_sample_service_instance_optional_input: ho\n",
```

```
            "templateMajorVersion": "1",
            "templateMinorVersion": "1",
            "templateName": "svc-simple"
        },
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "repositoryId": "myorg/myapp",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_required_input: hello\n  my_sample_pipeline_optional_input:
 bye\ninstances:\n- name: instance-svc-simple\n  environment: my-simple-
env\n  spec:\n    my_sample_service_instance_required_input: hi\n
 my_sample_service_instance_optional_input: ho\n",
        "status": "ACTIVE",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的查看服务数据或《Amazon Proton 用户指南》中的查看服务数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetService。

list-service-instances

以下代码示例演示了如何使用 list-service-instances。

Amazon CLI

示例 1：列出所有服务实例

以下 list-service-instances 示例列出了服务实例。

```
aws proton list-service-instances
```

输出：

```
{
    "serviceInstances": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
service-instance/instance-one",
            "createdAt": "2020-11-28T22:40:50.512000+00:00",
```

```
            "deploymentStatus": "SUCCEEDED",
            "environmentArn": "arn:aws:proton:region-id:123456789012:environment/
simple-env",
            "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
            "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
            "name": "instance-one",
            "serviceName": "simple-svc",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的查看服务实例数据或《Amazon Proton 用户指南》中的查看服务实例数据。

示例 2：列出指定的服务实例

以下 `get-service-instance` 示例获取服务实例。

```
aws proton get-service-instance \
    --name "instance-one" \
    --service-name "simple-svc"
```

输出：

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
        "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_optional_input: hello world\n
 my_sample_pipeline_required_input: pipeline up\ninstances:\n- name: instance-one\n
 environment: my-simple-env\n  spec:\n    my_sample_service_instance_optional_input:
 Ola\n    my_sample_service_instance_required_input: Ciao\n",
```

```
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的查看服务实例数据或《Amazon Proton 用户指南》中的查看服务实例数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceInstances。

update-service-instance

以下代码示例演示了如何使用 update-service-instance。

Amazon CLI

将服务实例更新到新的次要版本

以下 update-service-instance 示例将服务实例更新为其服务模板的新次要版本，该模板添加了一个名为“my-other-instance”的新实例，其中包含新的必填输入。

```
aws proton update-service-instance \
    --service-name "simple-svc" \
    --spec "file://service-spec.yaml " \
    --template-major-version "1" \
    --template-minor-version "1" \
    --deployment-type "MINOR_VERSION" \
    --name "instance-one"
```

的内容service-spec.yaml：

```
proton: ServiceSpec
pipeline:
    my_sample_pipeline_optional_input: "abc"
    my_sample_pipeline_required_input: "123"
instances:
    - name: "instance-one"
        environment: "simple-env"
        spec:
            my_sample_service_instance_optional_input: "def"
            my_sample_service_instance_required_input: "456"
```

```
    - name: "my-other-instance"
        environment: "simple-env"
        spec:
            my_sample_service_instance_required_input: "789"
```

输出：

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "environmentName": "arn:aws:proton:region-id:123456789012:environment/
simple-env",
        "lastDeploymentAttemptedAt": "2021-04-02T21:38:00.823000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的更新服务实例或《Amazon Proton 用户指南》中的更新服务实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServiceInstance。

update-service-pipeline

以下代码示例演示了如何使用 update-service-pipeline。

Amazon CLI

更新服务管道

以下 update-service-pipeline 示例将服务管道更新为其服务模板的新次要版本。

```
aws proton update-service-pipeline \
    --service-name "simple-svc" \
```

```
    --spec "file://service-spec.yaml" \
    --template-major-version "1" \
    --template-minor-version "1" \
    --deployment-type "MINOR_VERSION"
```

输出：

```
{
    "pipeline": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2021-04-02T21:39:28.991000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"my-instance\"\n    environment: \"MySimpleEnv
\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"MySimpleEnv\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的更新服务管道或《Amazon Proton 用户指南》中的更新服务管道。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServicePipeline。

update-service

以下代码示例演示了如何使用 update-service。

Amazon CLI

更新服务

以下 update-service 示例编辑服务描述。

```
aws proton update-service \
    --name "MySimpleService" \
    --description "Edit by updating description"
```

输出：

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
        "branchName": "mainline",
        "createdAt": "2021-03-12T22:39:42.318000+00:00",
        "description": "Edit by updating description",
        "lastModifiedAt": "2021-03-12T22:44:21.975000+00:00",
        "name": "MySimpleService",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "myorg/myapp",
        "status": "ACTIVE",
        "templateName": "fargate-service"
    }
}
```

有关更多信息，请参阅《Amazon Proton 管理员指南》中的编辑服务或《Amazon Proton 用户指南》中的编辑服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateService。

使用Amazon CLI的 Amazon RDS 示例

以下代码示例展示了如何通过将 Amazon Command Line Interface与 Amazon RDS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

add-option-to-option-group

以下代码示例演示了如何使用 add-option-to-option-group。

Amazon CLI

向选项组添加选项

以下 add-option-to-option-group 示例向选项组添加了选项。

```
aws rds add-option-to-option-group \
    --option-group-name myoptiongroup \
    --options OptionName=OEM,Port=5500,DBSecurityGroupMemberships=default \
    --apply-immediately
```

输出：

```
{
    "OptionGroup": {
        "OptionGroupName": "myoptiongroup",
        "OptionGroupDescription": "Test Option Group",
        "EngineName": "oracle-ee",
        "MajorEngineVersion": "12.1",
        "Options": [
            {
                "OptionName": "Timezone",
                "OptionDescription": "Change time zone",
                "Persistent": true,
                "Permanent": false,
                "OptionSettings": [
                    {
                        "Name": "TIME_ZONE",
                        "Value": "Australia/Sydney",
                        "DefaultValue": "UTC",
                        "Description": "Specifies the timezone the user wants to
 change the system time to",
                        "ApplyType": "DYNAMIC",
                        "DataType": "STRING",
                        "AllowedValues": "Africa/Cairo,Africa/Casablanca,Africa/
Harare,Africa/Lagos,Africa/Luanda,Africa/Monrovia,Africa/Nairobi,Africa/
Tripoli,Africa/Windhoek,America/Araguaina,America/Argentina/Buenos_Aires,America/
Asuncion,America/Bogota,America/Caracas,America/Chicago,America/Chihuahua,America/
```

```
Cuiaba,America/Denver,America/Detroit,America/Fortaleza,America/Godthab,America/
Guatemala,America/Halifax,America/Lima,America/Los_Angeles,America/Manaus,America/
Matamoros,America/Mexico_City,America/Monterrey,America/Montevideo,America/
New_York,America/Phoenix,America/Santiago,America/Sao_Paulo,America/Tijuana,America/
Toronto,Asia/Amman,Asia/Ashgabat,Asia/Baghdad,Asia/Baku,Asia/Bangkok,Asia/
Beirut,Asia/Calcutta,Asia/Damascus,Asia/Dhaka,Asia/Hong_Kong,Asia/Irkutsk,Asia/
Jakarta,Asia/Jerusalem,Asia/Kabul,Asia/Karachi,Asia/Kathmandu,Asia/Kolkata,Asia/
Krasnoyarsk,Asia/Magadan,Asia/Manila,Asia/Muscat,Asia/Novosibirsk,Asia/Rangoon,Asia/
Riyadh,Asia/Seoul,Asia/Shanghai,Asia/Singapore,Asia/Taipei,Asia/Tehran,Asia/
Tokyo,Asia/Ulaanbaatar,Asia/Vladivostok,Asia/Yakutsk,Asia/Yerevan,Atlantic/
Azores,Atlantic/Cape_Verde,Australia/Adelaide,Australia/Brisbane,Australia/
Darwin,Australia/Eucla,Australia/Hobart,Australia/Lord_Howe,Australia/
Perth,Australia/Sydney,Brazil/DeNoronha,Brazil/East,Canada/Newfoundland,Canada/
Saskatchewan,Etc/GMT-3,Europe/Amsterdam,Europe/Athens,Europe/Berlin,Europe/
Dublin,Europe/Helsinki,Europe/Kaliningrad,Europe/London,Europe/Madrid,Europe/
Moscow,Europe/Paris,Europe/Prague,Europe/Rome,Europe/Sarajevo,Pacific/Apia,Pacific/
Auckland,Pacific/Chatham,Pacific/Fiji,Pacific/Guam,Pacific/Honolulu,Pacific/
Kiritimati,Pacific/Marquesas,Pacific/Samoa,Pacific/Tongatapu,Pacific/Wake,US/
Alaska,US/Central,US/East-Indiana,US/Eastern,US/Pacific,UTC",
                        "IsModifiable": true,
                        "IsCollection": false
                    }
                ],
                "DBSecurityGroupMemberships": [],
                "VpcSecurityGroupMemberships": []
            },
            {
                "OptionName": "OEM",
                "OptionDescription": "Oracle 12c EM Express",
                "Persistent": false,
                "Permanent": false,
                "Port": 5500,
                "OptionSettings": [],
                "DBSecurityGroupMemberships": [
                    {
                        "DBSecurityGroupName": "default",
                        "Status": "authorized"
                    }
                ],
                "VpcSecurityGroupMemberships": []
            }
        ],
        "AllowsVpcAndNonVpcInstanceMemberships": false,
        "OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:myoptiongroup"
```

```
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的向选项组添加选项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddOptionToOptionGroup。

add-role-to-db-cluster

以下代码示例演示了如何使用 add-role-to-db-cluster。

Amazon CLI

将 Amazon Identity and Access Management（IAM）角色与数据库集群相关联

以下 add-role-to-db-cluster 示例将角色与数据库集群相关联。

```
aws rds add-role-to-db-cluster \
    --db-cluster-identifier mydbcluster \
    --role-arn arn:aws:iam::123456789012:role/RDSLoadFromS3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Aurora 用户指南》中的将 IAM 角色与 Amazon Aurora MySQL 数据库集群相关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddRoleToDbCluster。

add-role-to-db-instance

以下代码示例演示了如何使用 add-role-to-db-instance。

Amazon CLI

将 Amazon Identity and Access Management（IAM）角色与数据库实例相关联

以下 add-role-to-db-instance 示例向名为 test-instance 的 Oracle 数据库实例添加了角色。

```
aws rds add-role-to-db-instance \
    --db-instance-identifier test-instance \
    --feature-name S3_INTEGRATION \
```

```
    --role-arn arn:aws:iam::111122223333:role/rds-s3-integration-role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的 Amazon RDS Oracle 与 Amazon S3 集成的先决条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddRoleToDbInstance。

add-source-identifier-to-subscription

以下代码示例演示了如何使用 add-source-identifier-to-subscription。

Amazon CLI

向订阅添加源标识符

以下 add-source-identifier 示例向现有订阅添加了另一个源标识符。

```
aws rds add-source-identifier-to-subscription \
    --subscription-name my-instance-events \
    --source-identifier test-instance-repl
```

输出：

```
{
    "EventSubscription": {
        "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",
        "CustSubscriptionId": "my-instance-events",
        "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
        "Enabled": false,
        "Status": "modifying",
        "EventCategoriesList": [
            "backup",
            "recovery"
        ],
        "CustomerAwsId": "123456789012",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
        "SourceType": "db-instance",
        "SourceIdsList": [
            "test-instance",
```

```
            "test-instance-repl"
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddSourceIdentifierToSubscription。

add-tags-to-resource

以下代码示例演示了如何使用 add-tags-to-resource。

Amazon CLI

为资源添加标签

以下 add-tags-to-resource 示例向 RDS 数据库添加标签。

```
aws rds add-tags-to-resource \
    --resource-name arn:aws:rds:us-east-1:123456789012:db:database-mysql \
    --tags "[{\"Key\": \"Name\",\"Value\": \"MyDatabase\"},{\"Key\": \"Environment
\",\"Value\": \"test\"}]"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的为 Amazon RDS 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToResource。

apply-pending-maintenance-action

以下代码示例演示了如何使用 apply-pending-maintenance-action。

Amazon CLI

应用待执行的维护操作

以下 apply-pending-maintenance-action 示例应用数据库集群的待处理维护操作。

```
aws rds apply-pending-maintenance-action \
    --resource-identifier arn:aws:rds:us-east-1:123456789012:cluster:my-db-cluster \
```

```
    --apply-action system-update \
    --opt-in-type immediate
```

输出：

```
{
    "ResourcePendingMaintenanceActions": {
        "ResourceIdentifier": "arn:aws:rds:us-east-1:123456789012:cluster:my-db-
cluster",
        "PendingMaintenanceActionDetails": [
            {
                "Action": "system-update",
                "OptInStatus": "immediate",
                "CurrentApplyDate": "2021-01-23T01:07:36.100Z",
                "Description": "Upgrade to Aurora PostgreSQL 3.3.2"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的维护数据库实例和《Amazon Aurora 用户指南》中的维护 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ApplyPendingMaintenanceAction。

authorize-db-security-group-ingress

以下代码示例演示了如何使用 authorize-db-security-group-ingress。

Amazon CLI

将 Amazon Identity and Access Management（IAM）角色与数据库实例相关联

以下 authorize-db-security-group-ingress 示例使用 CIDR IP 范围 192.0.2.0/24 的入口规则配置默认安全组。

```
aws rds authorize-db-security-group-ingress \
    --db-security-group-name default \
    --cidrip 192.0.2.0/24
```

输出：

```
{
    "DBSecurityGroup": {
        "OwnerId": "123456789012",
        "DBSecurityGroupName": "default",
        "DBSecurityGroupDescription": "default",
        "EC2SecurityGroups": [],
        "IPRanges": [
            {
                "Status": "authorizing",
                "CIDRIP": "192.0.2.0/24"
            }
        ],
        "DBSecurityGroupArn": "arn:aws:rds:us-east-1:111122223333:secgrp:default"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的从 IP 范围向数据库安全组授予网络访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeDbSecurityGroupIngress。

backtrack-db-cluster

以下代码示例演示了如何使用 backtrack-db-cluster。

Amazon CLI

回溯 Aurora 数据库集群

以下 backtrack-db-cluster 示例将指定的数据库集群 sample-cluster 回溯到 2018 年 3 月 19 日上午 10 点。

```
aws rds backtrack-db-cluster --db-cluster-identifier sample-cluster --backtrack-
to 2018-03-19T10:00:00+00:00
```

此命令输出一个 JSON 块，用于确认对 RDS 资源的更改。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BacktrackDbCluster。

cancel-export-task

以下代码示例演示了如何使用 cancel-export-task。

Amazon CLI

取消将快照导出到 Amazon S3

以下 cancel-export-task 示例取消了正在进行的将快照导出到 Amazon S3 的导出任务。

```
aws rds cancel-export-task \
    --export-task-identifier my-s3-export-1
```

输出：

```
{
    "ExportTaskIdentifier": "my-s3-export-1",
    "SourceArn": "arn:aws:rds:us-east-1:123456789012:snapshot:publisher-final-
snapshot",
    "SnapshotTime": "2019-03-24T20:01:09.815Z",
    "S3Bucket": "amzn-s3-demo-bucket",
    "S3Prefix": "",
    "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/export-snap-S3-role",
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/abcd0000-7bfd-4594-af38-
aabbccddeeff",
    "Status": "CANCELING",
    "PercentProgress": 0,
    "TotalExtractedDataInGB": 0
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的取消快照导出任务或《Amazon Aurora 用户指南》中的取消快照导出任务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelExportTask。

copy-db-cluster-parameter-group

以下代码示例演示了如何使用 copy-db-cluster-parameter-group。

Amazon CLI

复制数据库集群参数组

以下 copy-db-cluster-parameter-group 示例复制了数据库集群参数组。

```
aws rds copy-db-cluster-parameter-group \
```

```
    --source-db-cluster-parameter-group-identifier mydbclusterpg \
    --target-db-cluster-parameter-group-identifier mydbclusterpgcopy \
    --target-db-cluster-parameter-group-description "Copy of mydbclusterpg parameter
 group"
```

输出：

```
{
    "DBClusterParameterGroup": {
        "DBClusterParameterGroupName": "mydbclusterpgcopy",
        "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-
pg:mydbclusterpgcopy",
        "DBParameterGroupFamily": "aurora-mysql5.7",
        "Description": "Copy of mydbclusterpg parameter group"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的复制数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbClusterParameterGroup。

copy-db-cluster-snapshot

以下代码示例演示了如何使用 copy-db-cluster-snapshot。

Amazon CLI

复制数据库集群快照

以下 copy-db-cluster-snapshot 示例创建数据库集群快照的副本，包括其标签。

```
aws rds copy-db-cluster-snapshot \
    --source-db-cluster-snapshot-identifier arn:aws:rds:us-
east-1:123456789012:cluster-snapshot:rds:myaurora-2019-06-04-09-16
    --target-db-cluster-snapshot-identifier myclustersnapshotcopy \
    --copy-tags
```

输出：

```
{
    "DBClusterSnapshot": {
```

```
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1e"
        ],
        "DBClusterSnapshotIdentifier": "myclustersnapshotcopy",
        "DBClusterIdentifier": "myaurora",
        "SnapshotCreateTime": "2019-06-04T09:16:42.649Z",
        "Engine": "aurora-mysql",
        "AllocatedStorage": 0,
        "Status": "available",
        "Port": 0,
        "VpcId": "vpc-6594f31c",
        "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
        "MasterUsername": "myadmin",
        "EngineVersion": "5.7.mysql_aurora.2.04.2",
        "LicenseModel": "aurora-mysql",
        "SnapshotType": "manual",
        "PercentProgress": 100,
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:myclustersnapshotcopy",
        "IAMDatabaseAuthenticationEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的复制快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbClusterSnapshot。

copy-db-parameter-group

以下代码示例演示了如何使用 copy-db-parameter-group。

Amazon CLI

复制数据库集群参数组

以下 copy-db-parameter-group 示例创建了数据库参数组的副本。

```
aws rds copy-db-parameter-group \
    --source-db-parameter-group-identifier mydbpg \
```

```
    --target-db-parameter-group-identifier mydbpgcopy \
    --target-db-parameter-group-description "Copy of mydbpg parameter group"
```

输出：

```
{
    "DBParameterGroup": {
        "DBParameterGroupName": "mydbpgcopy",
        "DBParameterGroupArn": "arn:aws:rds:us-east-1:814387698303:pg:mydbpgcopy",
        "DBParameterGroupFamily": "mysql5.7",
        "Description": "Copy of mydbpg parameter group"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的复制数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbParameterGroup。

copy-db-snapshot

以下代码示例演示了如何使用 copy-db-snapshot。

Amazon CLI

复制数据库快照

以下 copy-db-snapshot 示例创建了数据库快照副本。

```
aws rds copy-db-snapshot \
    --source-db-snapshot-identifier rds:database-mysql-2019-06-06-08-38
    --target-db-snapshot-identifier mydbsnapshotcopy
```

输出：

```
{
    "DBSnapshot": {
        "VpcId": "vpc-6594f31c",
        "Status": "creating",
        "Encrypted": true,
        "SourceDBSnapshotIdentifier": "arn:aws:rds:us-
east-1:123456789012:snapshot:rds:database-mysql-2019-06-06-08-38",
```

```
        "MasterUsername": "admin",
        "Iops": 1000,
        "Port": 3306,
        "LicenseModel": "general-public-license",
        "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mydbsnapshotcopy",
        "EngineVersion": "5.6.40",
        "OptionGroupName": "default:mysql-5-6",
        "ProcessorFeatures": [],
        "Engine": "mysql",
        "StorageType": "io1",
        "DbiResourceId": "db-ZI7UJ5BLKMBYFGX7FDENCKADC4",
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "SnapshotType": "manual",
        "IAMDatabaseAuthenticationEnabled": false,
        "SourceRegion": "us-east-1",
        "DBInstanceIdentifier": "database-mysql",
        "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
        "AvailabilityZone": "us-east-1f",
        "PercentProgress": 0,
        "AllocatedStorage": 100,
        "DBSnapshotIdentifier": "mydbsnapshotcopy"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的复制快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyDbSnapshot。

copy-option-group

以下代码示例演示了如何使用 copy-option-group。

Amazon CLI

复制选项组

以下 copy-option-group 示例创建了选项组的副本。

```
aws rds copy-option-group \
    --source-option-group-identifier myoptiongroup \
    --target-option-group-identifier new-option-group \
    --target-option-group-description "My option group copy"
```

输出：

```
{
    "OptionGroup": {
        "Options": [],
        "OptionGroupName": "new-option-group",
        "MajorEngineVersion": "11.2",
        "OptionGroupDescription": "My option group copy",
        "AllowsVpcAndNonVpcInstanceMemberships": true,
        "EngineName": "oracle-ee",
        "OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:new-option-group"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的为选项组创建副本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyOptionGroup。

create-blue-green-deployment

以下代码示例演示了如何使用 create-blue-green-deployment。

Amazon CLI

示例 1：为 RDS for MySQL 数据库实例创建蓝绿部署

以下 create-blue-green-deployment 示例为 MySQL 数据库实例创建了蓝绿部署。

```
aws rds create-blue-green-deployment \
    --blue-green-deployment-name bgd-cli-test-instance \
    --source arn:aws:rds:us-east-1:123456789012:db:my-db-instance \
    --target-engine-version 8.0 \
    --target-db-parameter-group-name mysql-80-group
```

输出：

```
{
    "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",
        "BlueGreenDeploymentName": "bgd-cli-test-instance",
        "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
```

```
        "SwitchoverDetails": [
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3"
            }
        ],
        "Tasks": [
            {
                "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                "Status": "PENDING"
            },
            {
                "Name": "DB_ENGINE_VERSION_UPGRADE",
                "Status": "PENDING"
            },
            {
                "Name": "CONFIGURE_BACKUPS",
                "Status": "PENDING"
            },
            {
                "Name": "CREATING_TOPOLOGY_OF_SOURCE",
                "Status": "PENDING"
            }
        ],
        "Status": "PROVISIONING",
        "CreateTime": "2022-02-25T21:18:51.183000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建蓝绿部署。

示例 2：为 Aurora MySQL 数据库集群创建蓝绿部署

以下 create-blue-green-deployment 示例为 Aurora MySQL 数据库集群创建了蓝绿部署。

```
aws rds create-blue-green-deployment \
    --blue-green-deployment-name my-blue-green-deployment \
    --source arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster \
    --target-engine-version 8.0 \
    --target-db-cluster-parameter-group-name ams-80-binlog-enabled \
    --target-db-parameter-group-name mysql-80-cluster-group
```

输出：

```
{
      "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
        "BlueGreenDeploymentName": "my-blue-green-deployment",
        "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
        "SwitchoverDetails": [
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-
mysql-cluster",
            "Status": "PROVISIONING"
          },
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-
cluster-1",
            "Status": "PROVISIONING"
          },
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-
cluster-2",
            "Status": "PROVISIONING"
          },
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-
cluster-3",
            "Status": "PROVISIONING"
          },
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-endpoint:my-
excluded-member-endpoint",
            "Status": "PROVISIONING"
          },
```

```
          {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-endpoint:my-
reader-endpoint",
            "Status": "PROVISIONING"
          }
        ],
        "Tasks": [
          {
            "Name": "CREATING_READ_REPLICA_OF_SOURCE",
            "Status": "PENDING"
          },
          {
            "Name": "DB_ENGINE_VERSION_UPGRADE",
            "Status": "PENDING"
          },
          {
            "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
            "Status": "PENDING"
          },
          {
            "Name": "CREATE_CUSTOM_ENDPOINTS",
            "Status": "PENDING"
          }
        ],
        "Status": "PROVISIONING",
        "CreateTime": "2022-02-25T21:12:00.288000+00:00"
      }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建蓝绿部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBlueGreenDeployment。

create-db-cluster-endpoint

以下代码示例演示了如何使用 create-db-cluster-endpoint。

Amazon CLI

创建自定义数据库集群端点

以下 create-db-cluster-endpoint 示例创建了自定义数据库集群端点并将其与指定的 Aurora 数据库集群相关联。

```
aws rds create-db-cluster-endpoint \
    --db-cluster-endpoint-identifier mycustomendpoint \
    --endpoint-type reader \
    --db-cluster-identifier mydbcluster \
    --static-members dbinstance1 dbinstance2
```

输出：

```
{
    "DBClusterEndpointIdentifier": "mycustomendpoint",
    "DBClusterIdentifier": "mydbcluster",
    "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",
    "Endpoint": "mycustomendpoint.cluster-custom-cnpexample.us-
east-1.rds.amazonaws.com",
    "Status": "creating",
    "EndpointType": "CUSTOM",
    "CustomEndpointType": "READER",
    "StaticMembers": [
        "dbinstance1",
        "dbinstance2"
    ],
    "ExcludedMembers": [],
    "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:mycustomendpoint"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 连接管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbClusterEndpoint。

create-db-cluster-parameter-group

以下代码示例演示了如何使用 create-db-cluster-parameter-group。

Amazon CLI

创建数据库集群参数组

以下 create-db-cluster-parameter-group 示例创建了一个数据库集群参数组。

```
aws rds create-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterparametergroup \
```

```
    --db-parameter-group-family aurora5.6 \
    --description "My new cluster parameter group"
```

输出：

```
{
    "DBClusterParameterGroup": {
        "DBClusterParameterGroupName": "mydbclusterparametergroup",
        "DBParameterGroupFamily": "aurora5.6",
        "Description": "My new cluster parameter group",
        "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-
pg:mydbclusterparametergroup"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbClusterParameterGroup。

create-db-cluster-snapshot

以下代码示例演示了如何使用 create-db-cluster-snapshot。

Amazon CLI

创建数据库集群快照

以下 create-db-cluster-snapshot 示例创建了数据库集群快照。

```
aws rds create-db-cluster-snapshot \
    --db-cluster-identifier mydbcluster \
    --db-cluster-snapshot-identifier mydbclustersnapshot
```

输出：

```
{
    "DBClusterSnapshot": {
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1e"
        ],
```

```
        "DBClusterSnapshotIdentifier": "mydbclustersnapshot",
        "DBClusterIdentifier": "mydbcluster",
        "SnapshotCreateTime": "2019-06-18T21:21:00.469Z",
        "Engine": "aurora-mysql",
        "AllocatedStorage": 1,
        "Status": "creating",
        "Port": 0,
        "VpcId": "vpc-6594f31c",
        "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
        "MasterUsername": "myadmin",
        "EngineVersion": "5.7.mysql_aurora.2.04.2",
        "LicenseModel": "aurora-mysql",
        "SnapshotType": "manual",
        "PercentProgress": 0,
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:mydbclustersnapshot",
        "IAMDatabaseAuthenticationEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建数据库集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbClusterSnapshot。

create-db-cluster

以下代码示例演示了如何使用 create-db-cluster。

Amazon CLI

示例 1：创建与 MySQL 5.7 兼容的数据库集群

以下 create-db-cluster 示例使用默认引擎版本创建与 MySQL 5.7 兼容的数据库集群。然后，将示例密码 secret99 替换为安全密码。在使用控制台创建数据库集群时，Amazon RDS 会为您的数据库集群自动创建写入器数据库实例。但是，使用 Amazon CLI 创建数据库集群时，必须使用 create-db-instance Amazon CLI 命令显式为数据库集群创建写入器数据库实例。

```
aws rds create-db-cluster \
    --db-cluster-identifier sample-cluster \
    --engine aurora-mysql \
    --engine-version 5.7 \
```

```
    --master-username admin \
    --master-user-password secret99 \
    --db-subnet-group-name default \
    --vpc-security-group-ids sg-0b9130572daf3dc16
```

输出：

```
{
    "DBCluster": {
        "DBSubnetGroup": "default",
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-0b9130572daf3dc16",
                "Status": "active"
            }
        ],
        "AllocatedStorage": 1,
        "AssociatedRoles": [],
        "PreferredBackupWindow": "09:12-09:42",
        "ClusterCreateTime": "2023-02-27T23:21:33.048Z",
        "DeletionProtection": false,
        "IAMDatabaseAuthenticationEnabled": false,
        "ReadReplicaIdentifiers": [],
        "EngineMode": "provisioned",
        "Engine": "aurora-mysql",
        "StorageEncrypted": false,
        "MultiAZ": false,
        "PreferredMaintenanceWindow": "mon:04:31-mon:05:01",
        "HttpEndpointEnabled": false,
        "BackupRetentionPeriod": 1,
        "DbClusterResourceId": "cluster-ANPAJ4AE5446DAEXAMPLE",
        "DBClusterIdentifier": "sample-cluster",
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1e"
        ],
        "MasterUsername": "master",
        "EngineVersion": "5.7.mysql_aurora.2.11.1",
        "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-cluster",
        "DBClusterMembers": [],
        "Port": 3306,
        "Status": "creating",
```

```
        "Endpoint": "sample-cluster.cluster-cnpexample.us-east-1.rds.amazonaws.com",
        "DBClusterParameterGroup": "default.aurora-mysql5.7",
        "HostedZoneId": "Z2R2ITUGPM61AM",
        "ReaderEndpoint": "sample-cluster.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
        "CopyTagsToSnapshot": false
    }
}
```

示例 2：创建与 PostgreSQL 兼容的数据库集群

以下 `create-db-cluster` 示例使用默认引擎版本创建与 PostgreSQL 兼容的数据库集群。然后，将示例密码 `secret99` 替换为安全密码。在使用控制台创建数据库集群时，Amazon RDS 会为您的数据库集群自动创建写入器数据库实例。但是，使用 Amazon CLI 创建数据库集群时，必须使用 `create-db-instance` Amazon CLI 命令显式为数据库集群创建写入器数据库实例。

```
aws rds create-db-cluster \
    --db-cluster-identifier sample-pg-cluster \
    --engine aurora-postgresql \
    --master-username master \
    --master-user-password secret99 \
    --db-subnet-group-name default \
    --vpc-security-group-ids sg-0b9130572daf3dc16
```

输出：

```
{
    "DBCluster": {
        "Endpoint": "sample-pg-cluster.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
        "HttpEndpointEnabled": false,
        "DBClusterMembers": [],
        "EngineMode": "provisioned",
        "CopyTagsToSnapshot": false,
        "HostedZoneId": "Z2R2ITUGPM61AM",
        "IAMDatabaseAuthenticationEnabled": false,
        "AllocatedStorage": 1,
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-0b9130572daf3dc16",
                "Status": "active"
            }
```

```
        ],
        "DeletionProtection": false,
        "StorageEncrypted": false,
        "BackupRetentionPeriod": 1,
        "PreferredBackupWindow": "09:56-10:26",
        "ClusterCreateTime": "2023-02-27T23:26:08.371Z",
        "DBClusterParameterGroup": "default.aurora-postgresql13",
        "EngineVersion": "13.7",
        "Engine": "aurora-postgresql",
        "Status": "creating",
        "DBClusterIdentifier": "sample-pg-cluster",
        "MultiAZ": false,
        "Port": 5432,
        "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-pg-
cluster",
        "AssociatedRoles": [],
        "DbClusterResourceId": "cluster-ANPAJ4AE5446DAEXAMPLE",
        "PreferredMaintenanceWindow": "wed:03:33-wed:04:03",
        "ReaderEndpoint": "sample-pg-cluster.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
        "MasterUsername": "master",
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1c"
        ],
        "ReadReplicaIdentifiers": [],
        "DBSubnetGroup": "default"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbCluster。

`create-db-instance-read-replica`

以下代码示例演示了如何使用 `create-db-instance-read-replica`。

Amazon CLI

创建数据库实例只读副本

此示例为名为 test-instance 的现有数据库实例创建只读副本。只读副本名为 test-instance-repl。

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier test-instance-repl \
    --source-db-instance-identifier test-instance
```

输出：

```
{
    "DBInstance": {
        "IAMDatabaseAuthenticationEnabled": false,
        "MonitoringInterval": 0,
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance-repl",
        "ReadReplicaSourceDBInstanceIdentifier": "test-instance",
        "DBInstanceIdentifier": "test-instance-repl",
        ...some output truncated...
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbInstanceReadReplica。

create-db-instance

以下代码示例演示了如何使用 create-db-instance。

Amazon CLI

创建数据库实例

以下 create-db-instance 示例使用所需的选项启动新的数据库实例。

```
aws rds create-db-instance \
    --db-instance-identifier test-mysql-instance \
    --db-instance-class db.t3.micro \
    --engine mysql \
    --master-username admin \
    --master-user-password secret99 \
    --allocated-storage 20
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-mysql-instance",
        "DBInstanceClass": "db.t3.micro",
        "Engine": "mysql",
        "DBInstanceStatus": "creating",
        "MasterUsername": "admin",
        "AllocatedStorage": 20,
        "PreferredBackupWindow": "12:55-13:25",
        "BackupRetentionPeriod": 1,
        "DBSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-12345abc",
                "Status": "active"
            }
        ],
        "DBParameterGroups": [
            {
                "DBParameterGroupName": "default.mysql5.7",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "DBSubnetGroup": {
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "VpcId": "vpc-2ff2ff2f",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    },
                    "SubnetStatus": "Active"
                },
```

```
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetStatus": "Active"
                }
            ]
        },
        "PreferredMaintenanceWindow": "sun:08:07-sun:08:37",
        "PendingModifiedValues": {
            "MasterUserPassword": "****"
        },
        "MultiAZ": false,
        "EngineVersion": "5.7.22",
        "AutoMinorVersionUpgrade": true,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "general-public-license",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "default:mysql-5-7",
                "Status": "in-sync"
            }
        ],
        "PubliclyAccessible": true,
        "StorageType": "gp2",
        "DbInstancePort": 0,
        "StorageEncrypted": false,
        "DbiResourceId": "db-5555EXAMPLE44444444EXAMPLE",
        "CACertificateIdentifier": "rds-ca-2019",
        "DomainMemberships": [],
        "CopyTagsToSnapshot": false,
        "MonitoringInterval": 0,
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:test-mysql-
instance",
        "IAMDatabaseAuthenticationEnabled": false,
        "PerformanceInsightsEnabled": false,
```

```
        "DeletionProtection": false,
        "AssociatedRoles": []
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建 Amazon RDS 数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBInstance。

create-db-parameter-group

以下代码示例演示了如何使用 create-db-parameter-group。

Amazon CLI

创建数据库参数组

以下 create-db-parameter-group 示例创建一个数据库参数组。

```
aws rds create-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --db-parameter-group-family MySQL5.6 \
    --description "My new parameter group"
```

输出：

```
{
    "DBParameterGroup": {
        "DBParameterGroupName": "mydbparametergroup",
        "DBParameterGroupFamily": "mysql5.6",
        "Description": "My new parameter group",
        "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:mydbparametergroup"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBParameterGroup。

create-db-proxy-endpoint

以下代码示例演示了如何使用 create-db-proxy-endpoint。

Amazon CLI

为 RDS 数据库创建数据库代理端点

以下 create-db-proxy-endpoint 示例创建了一个数据库代理端点。

```
aws rds create-db-proxy-endpoint \
    --db-proxy-name proxyExample \
    --db-proxy-endpoint-name "proxyep1" \
    --vpc-subnet-ids subnetgroup1 subnetgroup2
```

输出：

```
{
"DBProxyEndpoint": {
        "DBProxyEndpointName": "proxyep1",
        "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-0123a01b12345c0ab",
        "DBProxyName": "proxyExample",
        "Status": "creating",
        "VpcId": "vpc-1234567",
        "VpcSecurityGroupIds": [
            "sg-1234",
            "sg-5678"
        ],
        "VpcSubnetIds": [
            "subnetgroup1",
            "subnetgroup2"
        ],
        "Endpoint": "proxyep1.endpoint.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
        "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
        "TargetRole": "READ_WRITE",
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建代理端点和《Amazon Aurora 用户指南》中的创建代理端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbProxyEndpoint。

create-db-proxy

以下代码示例演示了如何使用 create-db-proxy。

Amazon CLI

为 RDS 数据库创建数据库代理

以下 create-db-proxy 示例创建了一个数据库代理。

```
aws rds create-db-proxy \
    --db-proxy-name proxyExample \
    --engine-family MYSQL \
    --auth
 Description="proxydescription1",AuthScheme="SECRETS",SecretArn="arn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",IAMAuth="DISABLED",ClientPasswordAuthType="MYSQ
 \
    --role-arn arn:aws:iam::123456789123:role/ProxyRole \
    --vpc-subnet-ids subnetgroup1 subnetgroup2
```

输出：

```
{
"DBProxy": {
        "DBProxyName": "proxyExample",
        "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
        "EngineFamily": "MYSQL",
        "VpcId": "vpc-1234567",
        "VpcSecuritytGroupIds": [
            "sg-1234",
            "sg-5678",
            "sg-9101"
        ],
        "VpcSubnetIds": [
            "subnetgroup1",
            "subnetgroup2"
        ],
        "Auth": "[
            {
                "Description": "proxydescription1",
                "AuthScheme": "SECRETS",
                "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:proxysecret1-Abcd1e",
```

```
                "IAMAuth": "DISABLED"
            }
        ]",
        "RoleArn": "arn:aws:iam::12345678912:role/ProxyRole",
        "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-east-1.rds.amazonaws.com",
        "RequireTLS": false,
        "IdleClientTimeout": 1800,
        "DebuggingLogging": false,
        "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
        "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建 RDS 代理和《Amazon Aurora 用户指南》中的创建 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbProxy。

create-db-security-group

以下代码示例演示了如何使用 create-db-security-group。

Amazon CLI

创建 Amazon RDS 数据库安全组

以下 create-db-security-group 命令创建了一个新的 Amazon RDS 数据库安全组：

```
aws rds create-db-security-group --db-security-group-name mysecgroup --db-security-
group-description "My Test Security Group"
```

在此示例中，新的数据库安全组被命名为 mysecgroup 并带有描述。

输出：

```
{
    "DBSecurityGroup": {
        "OwnerId": "123456789012",
        "DBSecurityGroupName": "mysecgroup",
        "DBSecurityGroupDescription": "My Test Security Group",
        "VpcId": "vpc-a1b2c3d4",
        "EC2SecurityGroups": [],
        "IPRanges": [],
```

```
        "DBSecurityGroupArn": "arn:aws:rds:us-west-2:123456789012:secgrp:mysecgroup"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbSecurityGroup。

`create-db-shard-group`

以下代码示例演示了如何使用 `create-db-shard-group`。

Amazon CLI

示例 1：创建 Aurora PostgreSQL 主数据库集群

以下 `create-db-cluster` 示例创建了一个与 Aurora Serverless v2 和 Aurora Limitless 数据库兼容的 Aurora PostgreSQL SQL 主数据库集群。

```
aws rds create-db-cluster \
    --db-cluster-identifier my-sv2-cluster \
    --engine aurora-postgresql \
    --engine-version 15.2-limitless \
    --storage-type aurora-iopt1 \
    --serverless-v2-scaling-configuration MinCapacity=2,MaxCapacity=16 \
    --enable-limitless-database \
    --master-username myuser \
    --master-user-password mypassword \
    --enable-cloudwatch-logs-exports postgresql
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-east-2b",
            "us-east-2c",
            "us-east-2a"
        ],
        "BackupRetentionPeriod": 1,
        "DBClusterIdentifier": "my-sv2-cluster",
        "DBClusterParameterGroup": "default.aurora-postgresql15",
        "DBSubnetGroup": "default",
```

```
        "Status": "creating",
        "Endpoint": "my-sv2-cluster.cluster-cekycexample.us-
east-2.rds.amazonaws.com",
        "ReaderEndpoint": "my-sv2-cluster.cluster-ro-cekycexample.us-
east-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-postgresql",
        "EngineVersion": "15.2-limitless",
        "Port": 5432,
        "MasterUsername": "myuser",
        "PreferredBackupWindow": "06:05-06:35",
        "PreferredMaintenanceWindow": "mon:08:25-mon:08:55",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-########",
                "Status": "active"
            }
        ],
        "HostedZoneId": "Z2XHWR1EXAMPLE",
        "StorageEncrypted": false,
        "DbClusterResourceId": "cluster-XYEDT6ML6FHIXH4Q2J1EXAMPLE",
        "DBClusterArn": "arn:aws:rds:us-east-2:123456789012:cluster:my-sv2-cluster",
        "AssociatedRoles": [],
        "IAMDatabaseAuthenticationEnabled": false,
        "ClusterCreateTime": "2024-02-19T16:24:07.771000+00:00",
        "EnabledCloudwatchLogsExports": [
            "postgresql"
        ],
        "EngineMode": "provisioned",
        "DeletionProtection": false,
        "HttpEndpointEnabled": false,
        "CopyTagsToSnapshot": false,
        "CrossAccountClone": false,
        "DomainMemberships": [],
        "TagList": [],
        "StorageType": "aurora-iopt1",
        "AutoMinorVersionUpgrade": true,
        "ServerlessV2ScalingConfiguration": {
            "MinCapacity": 2.0,
            "MaxCapacity": 16.0
        },
        "NetworkType": "IPV4",
```

```
        "IOOptimizedNextAllowedModificationTime":
 "2024-03-21T16:24:07.781000+00:00",
        "LimitlessDatabase": {
            "Status": "not-in-use",
            "MinRequiredACU": 96.0
        }
    }
}
```

示例 2：创建主（写入器）数据库实例

以下 `create-db-instance` 示例创建了一个 Aurora Serverless v2 主（写入器）数据库实例。在使用控制台创建数据库集群时，Amazon RDS 会为您的数据库集群自动创建写入器数据库实例。但是，使用 Amazon CLI 创建数据库集群时，必须使用 `create-db-instance` Amazon CLI 命令显式为数据库集群创建写入器数据库实例。

```
aws rds create-db-instance \
    --db-instance-identifier my-sv2-instance \
    --db-cluster-identifier my-sv2-cluster \
    --engine aurora-postgresql \
    --db-instance-class db.serverless
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "my-sv2-instance",
        "DBInstanceClass": "db.serverless",
        "Engine": "aurora-postgresql",
        "DBInstanceStatus": "creating",
        "MasterUsername": "myuser",
        "AllocatedStorage": 1,
        "PreferredBackupWindow": "06:05-06:35",
        "BackupRetentionPeriod": 1,
        "DBSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-########",
                "Status": "active"
            }
        ],
        "DBParameterGroups": [
```

```
            {
                "DBParameterGroupName": "default.aurora-postgresql15",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "DBSubnetGroup": {
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "VpcId": "vpc-########",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-2c"
                    },
                    "SubnetOutpost": {},
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-2a"
                    },
                    "SubnetOutpost": {},
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-2b"
                    },
                    "SubnetOutpost": {},
                    "SubnetStatus": "Active"
                }
            ]
        },
        "PreferredMaintenanceWindow": "fri:09:01-fri:09:31",
        "PendingModifiedValues": {
            "PendingCloudwatchLogsExports": {
                "LogTypesToEnable": [
                    "postgresql"
                ]
            }
```

```
        },
        "MultiAZ": false,
        "EngineVersion": "15.2-limitless",
        "AutoMinorVersionUpgrade": true,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "postgresql-license",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "default:aurora-postgresql-15",
                "Status": "in-sync"
            }
        ],
        "PubliclyAccessible": false,
        "StorageType": "aurora-iopt1",
        "DbInstancePort": 0,
        "DBClusterIdentifier": "my-sv2-cluster",
        "StorageEncrypted": false,
        "DbiResourceId": "db-BIQTE3B3K3RM7M74SK5EXAMPLE",
        "CACertificateIdentifier": "rds-ca-rsa2048-g1",
        "DomainMemberships": [],
        "CopyTagsToSnapshot": false,
        "MonitoringInterval": 0,
        "PromotionTier": 1,
        "DBInstanceArn": "arn:aws:rds:us-east-2:123456789012:db:my-sv2-instance",
        "IAMDatabaseAuthenticationEnabled": false,
        "PerformanceInsightsEnabled": false,
        "DeletionProtection": false,
        "AssociatedRoles": [],
        "TagList": [],
        "CustomerOwnedIpEnabled": false,
        "BackupTarget": "region",
        "NetworkType": "IPV4",
        "StorageThroughput": 0,
        "CertificateDetails": {
            "CAIdentifier": "rds-ca-rsa2048-g1"
        },
        "DedicatedLogVolume": false
    }
}
```

示例 3：创建数据库分片组

以下 create-db-shard-group 示例在您的 Aurora PostgreSQL 主数据库集群中创建了一个数据库分片组。

```
aws rds create-db-shard-group \
    --db-shard-group-identifier my-db-shard-group \
    --db-cluster-identifier my-sv2-cluster \
    --max-acu 768
```

输出：

```
{
    "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
    "DBShardGroupIdentifier": "my-db-shard-group",
    "DBClusterIdentifier": "my-sv2-cluster",
    "MaxACU": 768.0,
    "ComputeRedundancy": 0,
    "Status": "creating",
    "PubliclyAccessible": false,
    "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-east-2.rds.amazonaws.com"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用 Aurora Serverless v2。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbShardGroup。

create-db-snapshot

以下代码示例演示了如何使用 create-db-snapshot。

Amazon CLI

创建数据库快照

以下 create-db-snapshot 示例创建数据库快照。

```
aws rds create-db-snapshot \
    --db-instance-identifier database-mysql \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
```

```
    "DBSnapshot": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBInstanceIdentifier": "database-mysql",
        "Engine": "mysql",
        "AllocatedStorage": 100,
        "Status": "creating",
        "Port": 3306,
        "AvailabilityZone": "us-east-1b",
        "VpcId": "vpc-6594f31c",
        "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
        "MasterUsername": "admin",
        "EngineVersion": "5.6.40",
        "LicenseModel": "general-public-license",
        "SnapshotType": "manual",
        "Iops": 1000,
        "OptionGroupName": "default:mysql-5-6",
        "PercentProgress": 0,
        "StorageType": "io1",
        "Encrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mydbsnapshot",
        "IAMDatabaseAuthenticationEnabled": false,
        "ProcessorFeatures": [],
        "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBSnapshot。

create-db-subnet-group

以下代码示例演示了如何使用 create-db-subnet-group。

Amazon CLI

创建数据库子网组

以下 create-db-subnet-group 示例使用现有子网创建了名为 mysubnetgroup 的数据库子网组。

```
aws rds create-db-subnet-group \
```

```
    --db-subnet-group-name mysubnetgroup \
    --db-subnet-group-description "test DB subnet group" \
    --subnet-ids
 '["subnet-0a1dc4e1a6f123456","subnet-070dd7ecb3aaaaaaa","subnet-00f5b198bc0abcdef"]'
```

输出：

```
{
    "DBSubnetGroup": {
        "DBSubnetGroupName": "mysubnetgroup",
        "DBSubnetGroupDescription": "test DB subnet group",
        "VpcId": "vpc-0f08e7610a1b2c3d4",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-070dd7ecb3aaaaaaa",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-00f5b198bc0abcdef",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2d"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-0a1dc4e1a6f123456",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                },
                "SubnetStatus": "Active"
            }
        ],
        "DBSubnetGroupArn": "arn:aws:rds:us-
west-2:0123456789012:subgrp:mysubnetgroup"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的在 VPC 中创建数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDbSubnetGroup。

create-event-subscription

以下代码示例演示了如何使用 create-event-subscription。

Amazon CLI

创建事件订阅

以下 create-event-subscription 示例为当前 Amazon 账户中的数据库实例创建了备份和恢复事件的订阅。您也可以将通知发送到 --sns-topic-arn 指定的 Amazon Simple Notification Service 主题。

```
aws rds create-event-subscription \
    --subscription-name my-instance-events \
    --source-type db-instance \
    --event-categories '["backup","recovery"]' \
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:interesting-events
```

输出：

```
{
    "EventSubscription": {
        "Status": "creating",
        "CustSubscriptionId": "my-instance-events",
        "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",
        "EventCategoriesList": [
            "backup",
            "recovery"
        ],
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
        "CustomerAwsId": "123456789012",
        "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
        "SourceType": "db-instance",
        "Enabled": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEventSubscription。

create-global-cluster

以下代码示例演示了如何使用 create-global-cluster。

Amazon CLI

创建全局数据库集群

以下 create-global-cluster 示例新建一个与 Aurora MySQL 兼容的全局数据库集群。

```
aws rds create-global-cluster \
    --global-cluster-identifier myglobalcluster \
    --engine aurora-mysql
```

输出：

```
{
    "GlobalCluster": {
        "GlobalClusterIdentifier": "myglobalcluster",
        "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",
        "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
        "Status": "available",
        "Engine": "aurora-mysql",
        "EngineVersion": "5.7.mysql_aurora.2.07.2",
        "StorageEncrypted": false,
        "DeletionProtection": false,
        "GlobalClusterMembers": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建 Aurora 全局数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGlobalCluster。

create-option-group

以下代码示例演示了如何使用 create-option-group。

Amazon CLI

创建 Amazon RDS 选项组

以下 create-option-group 命令为 Oracle Enterprise Edition 版本 11.2`, is named ``MyOptionGroup 创建新的 Amazon RDS 选项组并包含了描述。

```
aws rds create-option-group \
    --option-group-name MyOptionGroup \
    --engine-name oracle-ee \
    --major-engine-version 11.2 \
    --option-group-description "Oracle Database Manager Database Control"
```

输出：

```
{
    "OptionGroup": {
        "OptionGroupName": "myoptiongroup",
        "OptionGroupDescription": "Oracle Database Manager Database Control",
        "EngineName": "oracle-ee",
        "MajorEngineVersion": "11.2",
        "Options": [],
        "AllowsVpcAndNonVpcInstanceMemberships": true,
        "OptionGroupArn": "arn:aws:rds:us-west-2:123456789012:og:myoptiongroup"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOptionGroup。

delete-blue-green-deployment

以下代码示例演示了如何使用 delete-blue-green-deployment。

Amazon CLI

示例 1：在绿色环境中删除 RDS for MySQL 数据库实例的资源

以下 delete-blue-green-deployment 示例在绿色环境中删除了 RDS for MySQL 数据库实例的资源。

```
aws rds delete-blue-green-deployment \
    --blue-green-deployment-identifier bgd-v53303651eexfake \
    --delete-target
```

输出：

```
{
    "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",
        "BlueGreenDeploymentName": "bgd-cli-test-instance",
        "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
        "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-
rkfbpe",
        "SwitchoverDetails": [
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-green-rkfbpe",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1-green-j382ha",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-green-ejv4ao",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-vlpz3t",
                "Status": "AVAILABLE"
            }
        ],
        "Tasks": [
            {
                "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                "Status": "COMPLETED"
            },
            {
```

```
                "Name": "DB_ENGINE_VERSION_UPGRADE",
                "Status": "COMPLETED"
            },
            {
                "Name": "CONFIGURE_BACKUPS",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATING_TOPOLOGY_OF_SOURCE",
                "Status": "COMPLETED"
            }
        ],
        "Status": "DELETING",
        "CreateTime": "2022-02-25T21:18:51.183000+00:00",
        "DeleteTime": "2022-02-25T22:25:31.331000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除蓝绿部署。

示例 2：在绿色环境中删除 Aurora MySQL 数据库集群的资源

以下 `delete-blue-green-deployment` 示例在绿色环境中删除了 Aurora MySQL 数据库集群的资源。

```
aws rds delete-blue-green-deployment \
    --blue-green-deployment-identifier bgd-wi89nwzglccsfake \
    --delete-target
```

输出：

```
{
    "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
        "BlueGreenDeploymentName": "my-blue-green-deployment",
        "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
        "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3rnukl",
        "SwitchoverDetails": [
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
```

```
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3rnukl",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1-green-gpmaxf",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2-green-j2oajq",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3-green-mkxies",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwwzlg",
                "Status": "AVAILABLE"
            }
        ],
        "Tasks": [
            {
                "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                "Status": "COMPLETED"
```

```
            },
            {
                "Name": "DB_ENGINE_VERSION_UPGRADE",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATE_CUSTOM_ENDPOINTS",
                "Status": "COMPLETED"
            }
        ],
        "Status": "DELETING",
        "CreateTime": "2022-02-25T21:12:00.288000+00:00",
        "DeleteTime": "2022-02-25T22:29:11.336000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的删除蓝绿部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBlueGreenDeployment。

delete-db-cluster-endpoint

以下代码示例演示了如何使用 delete-db-cluster-endpoint。

Amazon CLI

删除自定义数据库集群端点

以下 delete-db-cluster-endpoint 示例删除指定的自定义数据库集群端点。

```
aws rds delete-db-cluster-endpoint \
    --db-cluster-endpoint-identifier mycustomendpoint
```

输出：

```
{
    "DBClusterEndpointIdentifier": "mycustomendpoint",
    "DBClusterIdentifier": "mydbcluster",
```

```
    "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",
    "Endpoint": "mycustomendpoint.cluster-custom-cnpexample.us-
east-1.rds.amazonaws.com",
    "Status": "deleting",
    "EndpointType": "CUSTOM",
    "CustomEndpointType": "READER",
    "StaticMembers": [
        "dbinstance1",
        "dbinstance2",
        "dbinstance3"
    ],
    "ExcludedMembers": [],
    "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:mycustomendpoint"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 连接管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbClusterEndpoint。

delete-db-cluster-parameter-group

以下代码示例演示了如何使用 delete-db-cluster-parameter-group。

Amazon CLI

删除数据库集群参数组

以下 delete-db-cluster-parameter-group 示例删除指定的数据库集群参数组。

```
aws rds delete-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterparametergroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbClusterParameterGroup。

delete-db-cluster-snapshot

以下代码示例演示了如何使用 delete-db-cluster-snapshot。

Amazon CLI

删除数据库集群快照

以下 delete-db-cluster-snapshot 示例删除了指定的数据库集群快照。

```
aws rds delete-db-cluster-snapshot \
    --db-cluster-snapshot-identifier mydbclustersnapshot
```

输出：

```
{
    "DBClusterSnapshot": {
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1e"
        ],
        "DBClusterSnapshotIdentifier": "mydbclustersnapshot",
        "DBClusterIdentifier": "mydbcluster",
        "SnapshotCreateTime": "2019-06-18T21:21:00.469Z",
        "Engine": "aurora-mysql",
        "AllocatedStorage": 0,
        "Status": "available",
        "Port": 0,
        "VpcId": "vpc-6594f31c",
        "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
        "MasterUsername": "myadmin",
        "EngineVersion": "5.7.mysql_aurora.2.04.2",
        "LicenseModel": "aurora-mysql",
        "SnapshotType": "manual",
        "PercentProgress": 100,
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:mydbclustersnapshot",
        "IAMDatabaseAuthenticationEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的删除快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbClusterSnapshot。

delete-db-cluster

以下代码示例演示了如何使用 delete-db-cluster。

Amazon CLI

示例 1：删除数据库集群中的数据库实例

以下 delete-db-instance 示例删除了数据库集群中的最后一个数据库实例。如果数据库集群包含不处于正在删除状态的数据库实例，则无法将其删除。删除数据库集群中的数据库实例时，无法拍摄最终快照。

```
aws rds delete-db-instance \
    --db-instance-identifier database-3
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "database-3",
        "DBInstanceClass": "db.r4.large",
        "Engine": "aurora-postgresql",
        "DBInstanceStatus": "deleting",

    ...output omitted...

    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的删除 Aurora 数据库集群中的数据库实例。

示例 2：删除数据库集群

以下 delete-db-cluster 示例删除了名为 mycluster 的数据库集群并拍摄了名为 mycluster-final-snapshot 的最终快照。拍摄快照时，数据库集群的状态为可用。要跟踪删除进度，请使用 describe-db-clusters CLI 命令。

```
aws rds delete-db-cluster \
    --db-cluster-identifier mycluster \
    --no-skip-final-snapshot \
    --final-db-snapshot-identifier mycluster-final-snapshot
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 20,
        "AvailabilityZones": [
            "eu-central-1b",
            "eu-central-1c",
            "eu-central-1a"
        ],
        "BackupRetentionPeriod": 7,
        "DBClusterIdentifier": "mycluster",
        "DBClusterParameterGroup": "default.aurora-postgresql10",
        "DBSubnetGroup": "default-vpc-aa11bb22",
        "Status": "available",

    ...output omitted...

    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的具有单个数据库实例的 Aurora 集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbCluster。

delete-db-instance-automated-backup

以下代码示例演示了如何使用 delete-db-instance-automated-backup。

Amazon CLI

从区域中删除已复制的自动备份

以下 delete-db-instance-automated-backup 示例删除了具有指定 Amazon 资源名称（ARN）的自动备份。

```
aws rds delete-db-instance-automated-backup \
    --db-instance-automated-backups-arn "arn:aws:rds:us-west-2:123456789012:auto-backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"
```

输出：

```
{
```

```
    "DBInstanceAutomatedBackup": {
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
        "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
        "Region": "us-east-1",
        "DBInstanceIdentifier": "new-orcl-db",
        "RestoreWindow": {},
        "AllocatedStorage": 20,
        "Status": "deleting",
        "Port": 1521,
        "AvailabilityZone": "us-east-1b",
        "VpcId": "vpc-########",
        "InstanceCreateTime": "2020-12-04T15:28:31Z",
        "MasterUsername": "admin",
        "Engine": "oracle-se2",
        "EngineVersion": "12.1.0.2.v21",
        "LicenseModel": "bring-your-own-license",
        "OptionGroupName": "default:oracle-se2-12-1",
        "Encrypted": false,
        "StorageType": "gp2",
        "IAMDatabaseAuthenticationEnabled": false,
        "BackupRetentionPeriod": 7,
        "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-
backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除复制备份。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbInstanceAutomatedBackup。

delete-db-instance

以下代码示例演示了如何使用 delete-db-instance。

Amazon CLI

删除数据库实例

以下 delete-db-instance 示例在创建名为 test-instance-final-snap 的最终数据库快照后删除指定的数据库实例。

```
aws rds delete-db-instance \
```

```
    --db-instance-identifier test-instance \
    --final-db-snapshot-identifier test-instance-final-snap
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-instance",
        "DBInstanceStatus": "deleting",
        ...some output truncated...
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDBInstance。

delete-db-parameter-group

以下代码示例演示了如何使用 delete-db-parameter-group。

Amazon CLI

删除数据库参数组

以下 command 示例删除一个数据库参数组。

```
aws rds delete-db-parameter-group \
    --db-parameter-group-name mydbparametergroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDBParameterGroup。

delete-db-proxy-endpoint

以下代码示例演示了如何使用 delete-db-proxy-endpoint。

Amazon CLI

删除 RDS 数据库的数据库代理端点

以下 delete-db-proxy-endpoint 示例删除了目标数据库的数据库代理端点。

```
aws rds delete-db-proxy-endpoint \
    --db-proxy-endpoint-name proxyEP1
```

输出：

```
{
"DBProxyEndpoint":
    {
        "DBProxyEndpointName": "proxyEP1",
        "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-0123a01b12345c0ab",
        "DBProxyName": "proxyExample",
        "Status": "deleting",
        "VpcId": "vpc-1234567",
        "VpcSecurityGroupIds": [
            "sg-1234",
            "sg-5678"
        ],
        "VpcSubnetIds": [
            "subnetgroup1",
            "subnetgroup2"
        ],
        "Endpoint": "proxyEP1.endpoint.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
        "CreatedDate": "2023-04-13T01:49:38.568000+00:00",
        "TargetRole": "READ_ONLY",
        "IsDefault": false
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除代理端点和《Amazon Aurora 用户指南》中的删除代理端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbProxyEndpoint。

`delete-db-proxy`

以下代码示例演示了如何使用 `delete-db-proxy`。

Amazon CLI

删除 RDS 数据库的数据库代理

以下 delete-db-proxy 示例删除了数据库代理。

```
aws rds delete-db-proxy \
    --db-proxy-name proxyExample
```

输出：

```
{
        "DBProxy":
        {
            "DBProxyName": "proxyExample",
            "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
            "Status": "deleting",
            "EngineFamily": "PostgreSQL",
            "VpcId": "vpc-1234567",
            "VpcSecurityGroupIds": [
                "sg-1234",
                "sg-5678"
            ],
            "VpcSubnetIds": [
                "subnetgroup1",
                "subnetgroup2"
            ],
            "Auth": "[
                {
                "Description": "proxydescription`"
                "AuthScheme": "SECRETS",
                "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:proxysecret1-Abcd1e",
                "IAMAuth": "DISABLED"
                } ],
            "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",
            "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
            "RequireTLS": false,
            "IdleClientTimeout": 1800,
            "DebuggingLogging": false,
        "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
        "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除 RDS 代理和《Amazon Aurora 用户指南》中的删除 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbProxy。

delete-db-security-group

以下代码示例演示了如何使用 delete-db-security-group。

Amazon CLI

删除数据库安全组

以下 delete-db-security-group 示例删除了名为 mysecuritygroup 的数据库安全组。

```
aws rds delete-db-security-group \
    --db-security-group-name mysecuritygroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库安全组（EC2-Classic 平台）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbSecurityGroup。

delete-db-shard-group

以下代码示例演示了如何使用 delete-db-shard-group。

Amazon CLI

示例 1：删除数据库分片组失败

以下 delete-db-shard-group 示例显示了在删除所有数据库和架构之前尝试删除数据库分片组时发生的错误。

```
aws rds delete-db-shard-group \
    --db-shard-group-identifier limitless-test-shard-grp
```

输出：

```
An error occurred (InvalidDBShardGroupState) when calling the DeleteDBShardGroup
 operation: Unable to delete the DB shard group limitless-test-db-shard-group.
```

```
Delete all of your Limitless Database databases and schemas, then try again.
```

示例 2：成功删除数据库分片组

以下 delete-db-shard-group 示例在您删除所有数据库和架构（包括 public 架构）后删除了数据库分片组。

```
aws rds delete-db-shard-group \
    --db-shard-group-identifier limitless-test-shard-grp
```

输出：

```
{
    "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
    "DBShardGroupIdentifier": "limitless-test-shard-grp",
    "DBClusterIdentifier": "limitless-test-cluster",
    "MaxACU": 768.0,
    "ComputeRedundancy": 0,
    "Status": "deleting",
    "PubliclyAccessible": true,
    "Endpoint": "limitless-test-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的删除 Aurora 数据库集群和数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbShardGroup。

delete-db-snapshot

以下代码示例演示了如何使用 delete-db-snapshot。

Amazon CLI

删除数据库快照

以下 delete-db-snapshot 示例删除了指定的数据库快照。

```
aws rds delete-db-snapshot \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
    "DBSnapshot": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBInstanceIdentifier": "database-mysql",
        "SnapshotCreateTime": "2019-06-18T22:08:40.702Z",
        "Engine": "mysql",
        "AllocatedStorage": 100,
        "Status": "deleted",
        "Port": 3306,
        "AvailabilityZone": "us-east-1b",
        "VpcId": "vpc-6594f31c",
        "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
        "MasterUsername": "admin",
        "EngineVersion": "5.6.40",
        "LicenseModel": "general-public-license",
        "SnapshotType": "manual",
        "Iops": 1000,
        "OptionGroupName": "default:mysql-5-6",
        "PercentProgress": 100,
        "StorageType": "io1",
        "Encrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
        "DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mydbsnapshot",
        "IAMDatabaseAuthenticationEnabled": false,
        "ProcessorFeatures": [],
        "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbSnapshot。

delete-db-subnet-group

以下代码示例演示了如何使用 delete-db-subnet-group。

Amazon CLI

删除数据库子网组

以下 delete-db-subnet-group 示例删除了名为 mysubnetgroup 的数据库子网组。

```
aws rds delete-db-subnet-group --db-subnet-group-name mysubnetgroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的在 VPC 中使用数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDbSubnetGroup。

delete-event-subscription

以下代码示例演示了如何使用 delete-event-subscription。

Amazon CLI

删除事件订阅

以下 delete-event-subscription 示例删除了指定的事件订阅。

```
aws rds delete-event-subscription --subscription-name my-instance-events
```

输出：

```
{
    "EventSubscription": {
        "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
        "CustomerAwsId": "123456789012",
        "Enabled": false,
        "SourceIdsList": [
            "test-instance"
        ],
        "SourceType": "db-instance",
        "EventCategoriesList": [
            "backup",
            "recovery"
        ],
        "SubscriptionCreationTime": "2018-07-31 23:22:01.893",
        "CustSubscriptionId": "my-instance-events",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
        "Status": "deleting"
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEventSubscription。

delete-global-cluster

以下代码示例演示了如何使用 delete-global-cluster。

Amazon CLI

删除全局数据库集群

以下 delete-global-cluster 示例删除了与 Aurora MySQL 兼容的全局数据库集群。输出显示了您要删除的集群，但后续 describe-global-clusters 命令并未列出该数据库集群。

```
aws rds delete-global-cluster \
    --global-cluster-identifier myglobalcluster
```

输出：

```
{
    "GlobalCluster": {
        "GlobalClusterIdentifier": "myglobalcluster",
        "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",
        "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
        "Status": "available",
        "Engine": "aurora-mysql",
        "EngineVersion": "5.7.mysql_aurora.2.07.2",
        "StorageEncrypted": false,
        "DeletionProtection": false,
        "GlobalClusterMembers": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的删除 Aurora 全局数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGlobalCluster。

delete-option-group

以下代码示例演示了如何使用 delete-option-group。

Amazon CLI

删除选项组

以下 delete-option-group 示例删除指定的选项组。

```
aws rds delete-option-group \
    --option-group-name myoptiongroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除选项组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteOptionGroup。

deregister-db-proxy-targets

以下代码示例演示了如何使用 deregister-db-proxy-targets。

Amazon CLI

从数据库目标组中取消注册数据库代理目标

以下 deregister-db-proxy-targets 示例删除了代理 proxyExample 与其目标之间的关联。

```
aws rds deregister-db-proxy-targets \
    --db-proxy-name proxyExample \
    --db-instance-identifiers database-1
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的删除 RDS 代理和《Amazon Aurora 用户指南》中的删除 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterDbProxyTargets。

describe-account-attributes

以下代码示例演示了如何使用 describe-account-attributes。

Amazon CLI

描述账户属性

以下 describe-account-attributes 示例将检索当前 Amazon 账户的属性。

```
aws rds describe-account-attributes
```

输出：

```
{
    "AccountQuotas": [
        {
            "Max": 40,
            "Used": 4,
            "AccountQuotaName": "DBInstances"
        },
        {
            "Max": 40,
            "Used": 0,
            "AccountQuotaName": "ReservedDBInstances"
        },
        {
            "Max": 100000,
            "Used": 40,
            "AccountQuotaName": "AllocatedStorage"
        },
        {
            "Max": 25,
            "Used": 0,
            "AccountQuotaName": "DBSecurityGroups"
        },
        {
            "Max": 20,
            "Used": 0,
            "AccountQuotaName": "AuthorizationsPerDBSecurityGroup"
        },
        {
            "Max": 50,
            "Used": 1,
            "AccountQuotaName": "DBParameterGroups"
        },
        {
```

```
            "Max": 100,
            "Used": 3,
            "AccountQuotaName": "ManualSnapshots"
        },
        {
            "Max": 20,
            "Used": 0,
            "AccountQuotaName": "EventSubscriptions"
        },
        {
            "Max": 50,
            "Used": 1,
            "AccountQuotaName": "DBSubnetGroups"
        },
        {
            "Max": 20,
            "Used": 1,
            "AccountQuotaName": "OptionGroups"
        },
        {
            "Max": 20,
            "Used": 6,
            "AccountQuotaName": "SubnetsPerDBSubnetGroup"
        },
        {
            "Max": 5,
            "Used": 0,
            "AccountQuotaName": "ReadReplicasPerMaster"
        },
        {
            "Max": 40,
            "Used": 1,
            "AccountQuotaName": "DBClusters"
        },
        {
            "Max": 50,
            "Used": 0,
            "AccountQuotaName": "DBClusterParameterGroups"
        },
        {
            "Max": 5,
            "Used": 0,
            "AccountQuotaName": "DBClusterRoles"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAttributes。

describe-blue-green-deployments

以下代码示例演示了如何使用 describe-blue-green-deployments。

Amazon CLI

示例 1：描述 RDS 数据库实例在创建完成后的蓝绿部署

以下 describe-blue-green-deployment 示例在创建完成后检索蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments \
    --blue-green-deployment-identifier bgd-v53303651eexfake
```

输出：

```
{
    "BlueGreenDeployments": [
        {
            "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",
            "BlueGreenDeploymentName": "bgd-cli-test-instance",
            "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
            "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-
rkfbpe",
            "SwitchoverDetails": [
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-green-rkfbpe",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1-green-j382ha",
                    "Status": "AVAILABLE"
                },
```

```
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-green-ejv4ao",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-vlpz3t",
                    "Status": "AVAILABLE"
                }
            ],
            "Tasks": [
                {
                    "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "DB_ENGINE_VERSION_UPGRADE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CONFIGURE_BACKUPS",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATING_TOPOLOGY_OF_SOURCE",
                    "Status": "COMPLETED"
                }
            ],
            "Status": "AVAILABLE",
            "CreateTime": "2022-02-25T21:18:51.183000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看蓝绿部署。

示例 2：描述 Aurora MySQL 数据库集群的蓝绿部署

以下 `describe-blue-green-deployment` 示例检索蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments \
    --blue-green-deployment-identifier bgd-wi89nwzglccsfake
```

输出：

```
{
    "BlueGreenDeployments": [
        {
            "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
            "BlueGreenDeploymentName": "my-blue-green-deployment",
            "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
            "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3rnukl",
            "SwitchoverDetails": [
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3rnukl",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1-green-gpmaxf",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2-green-j2oajq",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3-green-mkxies",
                    "Status": "AVAILABLE"
                },
```

```
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwwzlg",
                    "Status": "AVAILABLE"
                }
            ],
            "Tasks": [
                {
                    "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "DB_ENGINE_VERSION_UPGRADE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_CUSTOM_ENDPOINTS",
                    "Status": "COMPLETED"
                }
            ],
            "Status": "AVAILABLE",
            "CreateTime": "2022-02-25T21:12:00.288000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的查看蓝绿部署。

示例 3：描述 Aurora MySQL 集群在切换后的蓝绿部署

在绿色环境升级为生产环境之后，以下 describe-blue-green-deployment 示例检索有关蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments \
    --blue-green-deployment-identifier bgd-wi89nwzglccsfake
```

输出：

```
{
    "BlueGreenDeployments": [
        {
            "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
            "BlueGreenDeploymentName": "my-blue-green-deployment",
            "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-old1",
            "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
            "SwitchoverDetails": [
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3-old1",
```

```
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
                    "Status": "SWITCHOVER_COMPLETED"
                }
            ],
            "Tasks": [
                {
                    "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "DB_ENGINE_VERSION_UPGRADE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_CUSTOM_ENDPOINTS",
                    "Status": "COMPLETED"
                }
            ],
            "Status": "SWITCHOVER_COMPLETED",
            "CreateTime": "2022-02-25T22:38:49.522000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的查看蓝绿部署。

示例 4：描述组合蓝绿部署

以下 describe-blue-green-deployment 示例检索组合蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments
```

输出：

```
{
    "BlueGreenDeployments": [
        {
            "BlueGreenDeploymentIdentifier": "bgd-wi89nwzgfakelccs",
            "BlueGreenDeploymentName": "my-blue-green-deployment",
            "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
            "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3rnukl",
            "SwitchoverDetails": [
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3rnukl",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1-green-gpmaxf",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2-green-j2oajq",
                    "Status": "AVAILABLE"
                },
                {
```

```
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3-green-mkxies",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",
                    "Status": "AVAILABLE"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwwzlg",
                    "Status": "AVAILABLE"
                }
            ],
            "Tasks": [
                {
                    "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "DB_ENGINE_VERSION_UPGRADE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATE_CUSTOM_ENDPOINTS",
                    "Status": "COMPLETED"
                }
            ],
            "Status": "AVAILABLE",
            "CreateTime": "2022-02-25T21:12:00.288000+00:00"
        },
        {
            "BlueGreenDeploymentIdentifier": "bgd-v5330365fake1eex",
```

```
            "BlueGreenDeploymentName": "bgd-cli-test-instance",
            "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-old1",
            "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
            "SwitchoverDetails": [
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
                    "Status": "SWITCHOVER_COMPLETED"
                },
                {
                    "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-old1",
                    "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
                    "Status": "SWITCHOVER_COMPLETED"
                }
            ],
            "Tasks": [
                {
                    "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "DB_ENGINE_VERSION_UPGRADE",
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CONFIGURE_BACKUPS",
```

```
                    "Status": "COMPLETED"
                },
                {
                    "Name": "CREATING_TOPOLOGY_OF_SOURCE",
                    "Status": "COMPLETED"
                }
            ],
            "Status": "SWITCHOVER_COMPLETED",
            "CreateTime": "2022-02-25T22:33:22.225000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看蓝绿部署和《Amazon Aurora 用户指南》中的查看蓝绿部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeBlueGreenDeployments。

describe-certificates

以下代码示例演示了如何使用 describe-certificates。

Amazon CLI

描述证书

以下 describe-certificates 示例检索与用户默认区域关联的证书的详细信息。

```
aws rds describe-certificates
```

输出：

```
{
    "Certificates": [
        {
            "CertificateIdentifier": "rds-ca-ecc384-g1",
            "CertificateType": "CA",
            "Thumbprint": "2ee3dcc06e50192559b13929e73484354f23387d",
            "ValidFrom": "2021-05-24T22:06:59+00:00",
            "ValidTill": "2121-05-24T23:06:59+00:00",
            "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-ecc384-g1",
            "CustomerOverride": false
```

```
        },
        {
            "CertificateIdentifier": "rds-ca-rsa4096-g1",
            "CertificateType": "CA",
            "Thumbprint": "19da4f2af579a8ae1f6a0fa77aa5befd874b4cab",
            "ValidFrom": "2021-05-24T22:03:20+00:00",
            "ValidTill": "2121-05-24T23:03:20+00:00",
            "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-rsa4096-g1",
            "CustomerOverride": false
        },
        {
            "CertificateIdentifier": "rds-ca-rsa2048-g1",
            "CertificateType": "CA",
            "Thumbprint": "7c40cb42714b6fdb2b296f9bbd0e8bb364436a76",
            "ValidFrom": "2021-05-24T21:59:00+00:00",
            "ValidTill": "2061-05-24T22:59:00+00:00",
            "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-rsa2048-g1",
            "CustomerOverride": true,
            "CustomerOverrideValidTill": "2061-05-24T22:59:00+00:00"
        },
        {
            "CertificateIdentifier": "rds-ca-2019",
            "CertificateType": "CA",
            "Thumbprint": "d40ddb29e3750dffa671c3140bbf5f478d1c8096",
            "ValidFrom": "2019-08-22T17:08:50+00:00",
            "ValidTill": "2024-08-22T17:08:50+00:00",
            "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-2019",
            "CustomerOverride": false
        }
    ],
    "DefaultCertificateForNewLaunches": "rds-ca-rsa2048-g1"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 SSL/TLS 加密与数据库实例的连接和《Amazon Aurora 用户指南》中的使用 SSL/TLS 加密与数据库集群的连接。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCertificates。

describe-db-cluster-backtracks

以下代码示例演示了如何使用 describe-db-cluster-backtracks。

Amazon CLI

描述数据库集群的回溯

以下 describe-db-cluster-backtracks 示例将检索指定数据库集群的详细信息。

```
aws rds describe-db-cluster-backtracks \
    --db-cluster-identifier mydbcluster
```

输出：

```
{
    "DBClusterBacktracks": [
        {
            "DBClusterIdentifier": "mydbcluster",
            "BacktrackIdentifier": "2f5f5294-0dd2-44c9-9f50-EXAMPLE",
            "BacktrackTo": "2021-02-12T04:59:22Z",
            "BacktrackedFrom": "2021-02-12T14:37:31.640Z",
            "BacktrackRequestCreationTime": "2021-02-12T14:36:18.819Z",
            "Status": "COMPLETED"
        },
        {
            "DBClusterIdentifier": "mydbcluster",
            "BacktrackIdentifier": "3c7a6421-af2a-4ea3-ae95-EXAMPLE",
            "BacktrackTo": "2021-02-11T22:53:46Z",
            "BacktrackedFrom": "2021-02-12T00:09:27.006Z",
            "BacktrackRequestCreationTime": "2021-02-12T00:07:53.487Z",
            "Status": "COMPLETED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的回溯 Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterBacktracks。

describe-db-cluster-endpoints

以下代码示例演示了如何使用 describe-db-cluster-endpoints。

Amazon CLI

示例 1：描述数据库集群端点

以下 describe-db-cluster-endpoints 示例检索数据库集群端点的详细信息。最常见的 Aurora 集群有两个端点。一个端点为 WRITER 类型。您可以将此端点用于所有 SQL 语句。另一个端点为 READER 类型。此端点只能用于 SELECT 和其他只读 SQL 语句。

```
aws rds describe-db-cluster-endpoints
```

输出：

```
{
    "DBClusterEndpoints": [
        {
            "DBClusterIdentifier": "my-database-1",
            "Endpoint": "my-database-1.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
            "Status": "creating",
            "EndpointType": "WRITER"
        },
        {
            "DBClusterIdentifier": "my-database-1",
            "Endpoint": "my-database-1.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
            "Status": "creating",
            "EndpointType": "READER"
        },
        {
            "DBClusterIdentifier": "mydbcluster",
            "Endpoint": "mydbcluster.cluster-cnpexamle.us-east-1.rds.amazonaws.com",
            "Status": "available",
            "EndpointType": "WRITER"
        },
        {
            "DBClusterIdentifier": "mydbcluster",
            "Endpoint": "mydbcluster.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
            "Status": "available",
            "EndpointType": "READER"
        }
    ]
}
```

示例 2：描述单个数据库集群的数据库集群端点

以下 describe-db-cluster-endpoints 示例将检索单个指定数据库集群的数据库集群端点的详细信息。Aurora Serverless 集群只有一个类型为 WRITER 的端点。

```
aws rds describe-db-cluster-endpoints \
    --db-cluster-identifier serverless-cluster
```

输出：

```
{
    "DBClusterEndpoints": [
        {
            "Status": "available",
            "Endpoint": "serverless-cluster.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
            "DBClusterIdentifier": "serverless-cluster",
            "EndpointType": "WRITER"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 连接管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterEndpoints。

describe-db-cluster-parameter-groups

以下代码示例演示了如何使用 describe-db-cluster-parameter-groups。

Amazon CLI

描述数据库集群参数组

以下 describe-db-cluster-parameter-groups 示例将检索有关数据库集群参数组的详细信息。

```
aws rds describe-db-cluster-parameter-groups
```

输出：

```
{
```

```
    "DBClusterParameterGroups": [
        {
            "DBClusterParameterGroupName": "default.aurora-mysql5.7",
            "DBParameterGroupFamily": "aurora-mysql5.7",
            "Description": "Default cluster parameter group for aurora-mysql5.7",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora-mysql5.7"
        },
        {
            "DBClusterParameterGroupName": "default.aurora-postgresql9.6",
            "DBParameterGroupFamily": "aurora-postgresql9.6",
            "Description": "Default cluster parameter group for aurora-
postgresql9.6",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora-postgresql9.6"
        },
        {
            "DBClusterParameterGroupName": "default.aurora5.6",
            "DBParameterGroupFamily": "aurora5.6",
            "Description": "Default cluster parameter group for aurora5.6",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora5.6"
        },
        {
            "DBClusterParameterGroupName": "mydbclusterpg",
            "DBParameterGroupFamily": "aurora-mysql5.7",
            "Description": "My DB cluster parameter group",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:mydbclusterpg"
        },
        {
            "DBClusterParameterGroupName": "mydbclusterpgcopy",
            "DBParameterGroupFamily": "aurora-mysql5.7",
            "Description": "Copy of mydbclusterpg parameter group",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:mydbclusterpgcopy"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterParameterGroups。

describe-db-cluster-parameters

以下代码示例演示了如何使用 describe-db-cluster-parameters。

Amazon CLI

示例 1：描述数据库集群参数组中的参数

以下 describe-db-cluster-parameters 示例将检索有关数据库集群参数组中参数的详细信息。

```
aws rds describe-db-cluster-parameters \
    --db-cluster-parameter-group-name mydbclusterpg
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "allow-suspicious-udfs",
            "Description": "Controls whether user-defined functions that have only
 an xxx symbol for the main function can be loaded",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot",
            "SupportedEngineModes": [
                "provisioned"
            ]
        },
        {
            "ParameterName": "aurora_lab_mode",
            "ParameterValue": "0",
            "Description": "Enables new features in the Aurora engine.",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": true,
            "ApplyMethod": "pending-reboot",
            "SupportedEngineModes": [
```

```
                "provisioned"
            ]
        },
        ...some output truncated...
    ]
}
```

示例 2：仅列出数据库集群参数组中的参数名称

以下 describe-db-cluster-parameters 示例将仅检索数据库集群参数组中参数的名称。

```
aws rds describe-db-cluster-parameters \
    --db-cluster-parameter-group-name default.aurora-mysql5.7 \
    --query 'Parameters[].{ParameterName:ParameterName}'
```

输出：

```
[
    {
        "ParameterName": "allow-suspicious-udfs"
    },
    {
        "ParameterName": "aurora_binlog_read_buffer_size"
    },
    {
        "ParameterName": "aurora_binlog_replication_max_yield_seconds"
    },
    {
        "ParameterName": "aurora_binlog_use_large_read_buffer"
    },
    {
        "ParameterName": "aurora_lab_mode"
    },

    ...some output truncated...
    }
]
```

示例 3：仅描述数据库集群参数组中的可修改参数

以下 describe-db-cluster-parameters 示例将仅检索可在数据库集群参数组中修改的参数的名称。

```
aws rds describe-db-cluster-parameters \
    --db-cluster-parameter-group-name default.aurora-mysql5.7 \
    --query 'Parameters[].{ParameterName:ParameterName,IsModifiable:IsModifiable} |
 [?IsModifiable == `true`]'
```

输出：

```
[
    {
        "ParameterName": "aurora_binlog_read_buffer_size",
        "IsModifiable": true
    },
    {
        "ParameterName": "aurora_binlog_replication_max_yield_seconds",
        "IsModifiable": true
    },
    {
        "ParameterName": "aurora_binlog_use_large_read_buffer",
        "IsModifiable": true
    },
    {
        "ParameterName": "aurora_lab_mode",
        "IsModifiable": true
    },

    ...some output truncated...
    }
]
```

示例 4：仅描述数据库集群参数组中可修改的布尔参数

以下 describe-db-cluster-parameters 示例将仅检索可在数据库集群参数组中修改且包含布尔数据类型的参数的名称。

```
aws rds describe-db-cluster-parameters \
    --db-cluster-parameter-group-name default.aurora-mysql5.7 \
    --query 'Parameters[].
{ParameterName:ParameterName,DataType:DataType,IsModifiable:IsModifiable} | [?
DataType == `boolean`] | [?IsModifiable == `true`]'
```

输出：

```
[
    {
        "DataType": "boolean",
        "ParameterName": "aurora_binlog_use_large_read_buffer",
        "IsModifiable": true
    },
    {
        "DataType": "boolean",
        "ParameterName": "aurora_lab_mode",
        "IsModifiable": true
    },
    {
        "DataType": "boolean",
        "ParameterName": "autocommit",
        "IsModifiable": true
    },
    {
        "DataType": "boolean",
        "ParameterName": "automatic_sp_privileges",
        "IsModifiable": true
    },
    ...some output truncated...
    }
]
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterParameters。

describe-db-cluster-snapshot-attributes

以下代码示例演示了如何使用 describe-db-cluster-snapshot-attributes。

Amazon CLI

描述数据库集群快照的属性名称和值

以下 describe-db-cluster-snapshot-attributes 示例将检索指定数据库集群快照的属性名称和值的详细信息。

```
aws rds describe-db-cluster-snapshot-attributes \
    --db-cluster-snapshot-identifier myclustersnapshot
```

输出：

```
{
    "DBClusterSnapshotAttributesResult": {
        "DBClusterSnapshotIdentifier": "myclustersnapshot",
        "DBClusterSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "123456789012"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的共享数据库集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterSnapshotAttributes。

describe-db-cluster-snapshots

以下代码示例演示了如何使用 describe-db-cluster-snapshots。

Amazon CLI

描述数据库集群的数据库集群快照

以下 describe-db-cluster-snapshots 示例将检索指定数据库集群的数据库集群快照的详细信息。

```
aws rds describe-db-cluster-snapshots \
    --db-cluster-identifier mydbcluster
```

输出：

```
{
    "DBClusterSnapshots": [
        {
            "AvailabilityZones": [
                "us-east-1a",
```

```
                "us-east-1b",
                "us-east-1e"
            ],
            "DBClusterSnapshotIdentifier": "myclustersnapshotcopy",
            "DBClusterIdentifier": "mydbcluster",
            "SnapshotCreateTime": "2019-06-04T09:16:42.649Z",
            "Engine": "aurora-mysql",
            "AllocatedStorage": 0,
            "Status": "available",
            "Port": 0,
            "VpcId": "vpc-6594f31c",
            "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
            "MasterUsername": "myadmin",
            "EngineVersion": "5.7.mysql_aurora.2.04.2",
            "LicenseModel": "aurora-mysql",
            "SnapshotType": "manual",
            "PercentProgress": 100,
            "StorageEncrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
AKIAIOSFODNN7EXAMPLE",
            "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:814387698303:cluster-
snapshot:myclustersnapshotcopy",
            "IAMDatabaseAuthenticationEnabled": false
        },
        {
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1e"
            ],
            "DBClusterSnapshotIdentifier": "rds:mydbcluster-2019-06-20-09-16",
            "DBClusterIdentifier": "mydbcluster",
            "SnapshotCreateTime": "2019-06-20T09:16:26.569Z",
            "Engine": "aurora-mysql",
            "AllocatedStorage": 0,
            "Status": "available",
            "Port": 0,
            "VpcId": "vpc-6594f31c",
            "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
            "MasterUsername": "myadmin",
            "EngineVersion": "5.7.mysql_aurora.2.04.2",
            "LicenseModel": "aurora-mysql",
            "SnapshotType": "automated",
            "PercentProgress": 100,
```

```
            "StorageEncrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-1:814387698303:key/
AKIAIOSFODNN7EXAMPLE",
            "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:rds:mydbcluster-2019-06-20-09-16",
            "IAMDatabaseAuthenticationEnabled": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的创建数据库集群快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusterSnapshots。

describe-db-clusters

以下代码示例演示了如何使用 describe-db-clusters。

Amazon CLI

示例 1：描述数据库集群

以下 describe-db-clusters 示例将检索指定数据库集群的详细信息。

```
aws rds describe-db-clusters \
    --db-cluster-identifier mydbcluster
```

输出：

```
{
    "DBClusters": [
        {
            "AllocatedStorage": 1,
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1e"
            ],
            "BackupRetentionPeriod": 1,
            "DatabaseName": "mydbcluster",
            "DBClusterIdentifier": "mydbcluster",
            "DBClusterParameterGroup": "default.aurora-mysql5.7",
            "DBSubnetGroup": "default",
```

```
            "Status": "available",
            "EarliestRestorableTime": "2019-06-19T09:16:28.210Z",
            "Endpoint": "mydbcluster.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
            "ReaderEndpoint": "mydbcluster.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
            "MultiAZ": true,
            "Engine": "aurora-mysql",
            "EngineVersion": "5.7.mysql_aurora.2.04.2",
            "LatestRestorableTime": "2019-06-20T22:38:14.908Z",
            "Port": 3306,
            "MasterUsername": "myadmin",
            "PreferredBackupWindow": "09:09-09:39",
            "PreferredMaintenanceWindow": "sat:04:09-sat:04:39",
            "ReadReplicaIdentifiers": [],
            "DBClusterMembers": [
                {
                    "DBInstanceIdentifier": "dbinstance3",
                    "IsClusterWriter": false,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "PromotionTier": 1
                },
                {
                    "DBInstanceIdentifier": "dbinstance1",
                    "IsClusterWriter": false,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "PromotionTier": 1
                },
                {
                    "DBInstanceIdentifier": "dbinstance2",
                    "IsClusterWriter": false,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "PromotionTier": 1
                },
                {
                    "DBInstanceIdentifier": "mydbcluster",
                    "IsClusterWriter": false,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "PromotionTier": 1
                },
                {
                    "DBInstanceIdentifier": "mydbcluster-us-east-1b",
                    "IsClusterWriter": false,
                    "DBClusterParameterGroupStatus": "in-sync",
```

```
                    "PromotionTier": 1
                },
                {
                    "DBInstanceIdentifier": "mydbcluster",
                    "IsClusterWriter": true,
                    "DBClusterParameterGroupStatus": "in-sync",
                    "PromotionTier": 1
                }
            ],
            "VpcSecurityGroups": [
                {
                    "VpcSecurityGroupId": "sg-0b9130572daf3dc16",
                    "Status": "active"
                }
            ],
            "HostedZoneId": "Z2R2ITUGPM61AM",
            "StorageEncrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-1:814387698303:key/
AKIAIOSFODNN7EXAMPLE",
            "DbClusterResourceId": "cluster-AKIAIOSFODNN7EXAMPLE",
            "DBClusterArn": "arn:aws:rds:us-
east-1:123456789012:cluster:mydbcluster",
            "AssociatedRoles": [],
            "IAMDatabaseAuthenticationEnabled": false,
            "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
            "EngineMode": "provisioned",
            "DeletionProtection": false,
            "HttpEndpointEnabled": false
        }
    ]
}
```

示例 2：列出所有数据库集群的某些属性

以下 `describe-db-clusters` 示例仅检索当前 Amazon 区域中所有数据库集群的 `DBClusterIdentifier`、`Endpoint` 和 `ReaderEndpoint` 属性。

```
aws rds describe-db-clusters \
    --query 'DBClusters[].
{DBClusterIdentifier:DBClusterIdentifier,Endpoint:Endpoint,ReaderEndpoint:ReaderEndpoint}'
```

输出：

```
[
    {
        "Endpoint": "cluster-57-2020-05-01-2270.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
        "ReaderEndpoint": "cluster-57-2020-05-01-2270.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
        "DBClusterIdentifier": "cluster-57-2020-05-01-2270"
    },
    {
        "Endpoint": "cluster-57-2020-05-01-4615.cluster-cnpexample.us-
east-1.rds.amazonaws.com",
        "ReaderEndpoint": "cluster-57-2020-05-01-4615.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
        "DBClusterIdentifier": "cluster-57-2020-05-01-4615"
    },
    {
        "Endpoint": "pg2-cluster.cluster-cnpexample.us-east-1.rds.amazonaws.com",
        "ReaderEndpoint": "pg2-cluster.cluster-ro-cnpexample.us-
east-1.rds.amazonaws.com",
        "DBClusterIdentifier": "pg2-cluster"
    },
    ...output omitted...
    }
]
```

示例 3：列出具有特定属性的数据库集群

以下 describe-db-clusters 示例仅检索使用 aurora-postgresql 数据库引擎的数据库集群的 DBClusterIdentifier 和 Engine 属性。

```
aws rds describe-db-clusters \
    --query 'DBClusters[].{DBClusterIdentifier:DBClusterIdentifier,Engine:Engine} |
 [?Engine == `aurora-postgresql`]'
```

输出：

```
[
    {
        "Engine": "aurora-postgresql",
        "DBClusterIdentifier": "pg2-cluster"
    }
]
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbClusters。

describe-db-engine-versions

以下代码示例演示了如何使用 describe-db-engine-versions。

Amazon CLI

描述 MySQL 数据库引擎的数据库引擎版本

以下 describe-db-engine-versions 示例显示了有关指定数据库引擎的每个数据库引擎版本的详细信息。

```
aws rds describe-db-engine-versions \
    --engine mysql
```

输出：

```
{
    "DBEngineVersions": [
        {
            "Engine": "mysql",
            "EngineVersion": "5.5.46",
            "DBParameterGroupFamily": "mysql5.5",
            "DBEngineDescription": "MySQL Community Edition",
            "DBEngineVersionDescription": "MySQL 5.5.46",
            "ValidUpgradeTarget": [
                {
                    "Engine": "mysql",
                    "EngineVersion": "5.5.53",
                    "Description": "MySQL 5.5.53",
                    "AutoUpgrade": false,
                    "IsMajorVersionUpgrade": false
                },
                {
                    "Engine": "mysql",
                    "EngineVersion": "5.5.54",
                    "Description": "MySQL 5.5.54",
                    "AutoUpgrade": false,
                    "IsMajorVersionUpgrade": false
                },
```

```
            {
                "Engine": "mysql",
                "EngineVersion": "5.5.57",
                "Description": "MySQL 5.5.57",
                "AutoUpgrade": false,
                "IsMajorVersionUpgrade": false
            },
            ...some output truncated...
        ]
    }
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的什么是 Amazon Relational Database Service（Amazon RDS）？。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBEngineVersions。

describe-db-instance-automated-backups

以下代码示例演示了如何使用 describe-db-instance-automated-backups。

Amazon CLI

描述数据库实例的自动备份

以下 describe-db-instance-automated-backups 示例显示了指定数据库实例的自动备份的详细信息。详细信息包括在其他 Amazon 区域复制的自动备份。

```
aws rds describe-db-instance-automated-backups \
    --db-instance-identifier new-orcl-db
```

输出：

```
{
    "DBInstanceAutomatedBackups": [
        {
            "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
            "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
            "Region": "us-east-1",
            "DBInstanceIdentifier": "new-orcl-db",
            "RestoreWindow": {
                "EarliestTime": "2020-12-07T21:05:20.939Z",
                "LatestTime": "2020-12-07T21:05:20.939Z"
            },
```

```
            "AllocatedStorage": 20,
            "Status": "replicating",
            "Port": 1521,
            "InstanceCreateTime": "2020-12-04T15:28:31Z",
            "MasterUsername": "admin",
            "Engine": "oracle-se2",
            "EngineVersion": "12.1.0.2.v21",
            "LicenseModel": "bring-your-own-license",
            "OptionGroupName": "default:oracle-se2-12-1",
            "Encrypted": false,
            "StorageType": "gp2",
            "IAMDatabaseAuthenticationEnabled": false,
            "BackupRetentionPeriod": 14,
            "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-
west-2:123456789012:auto-backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查找有关复制备份的信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbInstanceAutomatedBackups。

describe-db-instances

以下代码示例演示了如何使用 describe-db-instances。

Amazon CLI

描述数据库实例

以下 describe-db-instances 示例将检索有关指定数据库实例的详细信息。

```
aws rds describe-db-instances \
    --db-instance-identifier mydbinstancecf
```

输出：

```
{
    "DBInstances": [
        {
            "DBInstanceIdentifier": "mydbinstancecf",
```

```
            "DBInstanceClass": "db.t3.small",
            "Engine": "mysql",
            "DBInstanceStatus": "available",
            "MasterUsername": "masterawsuser",
            "Endpoint": {
                "Address": "mydbinstancecf.abcexample.us-east-1.rds.amazonaws.com",
                "Port": 3306,
                "HostedZoneId": "Z2R2ITUGPM61AM"
            },
            ...some output truncated...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBInstances。

describe-db-log-files

以下代码示例演示了如何使用 describe-db-log-files。

Amazon CLI

描述数据库实例的日志文件

以下 describe-db-log-files 示例将检索有关指定数据库实例的日志文件的详细信息。

```
aws rds describe-db-log-files -\
    -db-instance-identifier test-instance
```

输出：

```
{
    "DescribeDBLogFiles": [
        {
            "Size": 0,
            "LastWritten": 1533060000000,
            "LogFileName": "error/mysql-error-running.log"
        },
        {
            "Size": 2683,
            "LastWritten": 1532994300000,
            "LogFileName": "error/mysql-error-running.log.0"
```

```
        },
        {
            "Size": 107,
            "LastWritten": 1533057300000,
            "LogFileName": "error/mysql-error-running.log.18"
        },
        {
            "Size": 13105,
            "LastWritten": 1532991000000,
            "LogFileName": "error/mysql-error-running.log.23"
        },
        {
            "Size": 0,
            "LastWritten": 1533061200000,
            "LogFileName": "error/mysql-error.log"
        },
        {
            "Size": 3519,
            "LastWritten": 1532989252000,
            "LogFileName": "mysqlUpgrade"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbLogFiles。

describe-db-parameter-groups

以下代码示例演示了如何使用 describe-db-parameter-groups。

Amazon CLI

描述数据库参数组

以下 describe-db-parameter-groups 示例将检索有关数据库参数组的详细信息。

```
aws rds describe-db-parameter-groups
```

输出：

```
{
    "DBParameterGroups": [
```

```
        {
            "DBParameterGroupName": "default.aurora-mysql5.7",
            "DBParameterGroupFamily": "aurora-mysql5.7",
            "Description": "Default parameter group for aurora-mysql5.7",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora-mysql5.7"
        },
        {
            "DBParameterGroupName": "default.aurora-postgresql9.6",
            "DBParameterGroupFamily": "aurora-postgresql9.6",
            "Description": "Default parameter group for aurora-postgresql9.6",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora-postgresql9.6"
        },
        {
            "DBParameterGroupName": "default.aurora5.6",
            "DBParameterGroupFamily": "aurora5.6",
            "Description": "Default parameter group for aurora5.6",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora5.6"
        },
        {
            "DBParameterGroupName": "default.mariadb10.1",
            "DBParameterGroupFamily": "mariadb10.1",
            "Description": "Default parameter group for mariadb10.1",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.mariadb10.1"
        },
        ...some output truncated...
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBParameterGroups。

`describe-db-parameters`

以下代码示例演示了如何使用 `describe-db-parameters`。

Amazon CLI

描述数据库参数组中的参数

以下 describe-db-parameters 示例将检索有关指定数据库参数组的详细信息。

```
aws rds describe-db-parameters \
    --db-parameter-group-name mydbpg
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "allow-suspicious-udfs",
            "Description": "Controls whether user-defined functions that have only
 an xxx symbol for the main function can be loaded",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot"
        },
        {
            "ParameterName": "auto_generate_certs",
            "Description": "Controls whether the server autogenerates SSL key and
 certificate files in the data directory, if they do not already exist.",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot"
        },
        ...some output truncated...
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBParameters。

describe-db-proxies

以下代码示例演示了如何使用 describe-db-proxies。

Amazon CLI

描述 RDS 数据库的数据库代理

以下 describe-db-proxies 示例返回了有关数据库代理的信息。

```
aws rds describe-db-proxies
```

输出：

```
{
    "DBProxies": [
        {
            "DBProxyName": "proxyExample1",
            "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
            "Status": "available",
            "EngineFamily": "PostgreSQL",
            "VpcId": "vpc-1234567",
            "VpcSecurityGroupIds": [
                "sg-1234"
            ],
            "VpcSubnetIds": [
                "subnetgroup1",
                "subnetgroup2"
            ],
            "Auth": "[
                {
                    "Description": "proxydescription1"
                    "AuthScheme": "SECRETS",
                    "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",
                    "IAMAuth": "DISABLED"
                }
            ]",
            "RoleArn": "arn:aws:iam::12345678912??:role/ProxyPostgreSQLRole",
            "Endpoint": "proxyExample1.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
            "RequireTLS": false,
            "IdleClientTimeout": 1800,
            "DebuggingLogging": false,
            "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
            "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
```

```
        },
        {
            "DBProxyName": "proxyExample2",
            "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-1234a12b23456c1ab",
            "Status": "available",
            "EngineFamily": "PostgreSQL",
            "VpcId": "sg-1234567",
            "VpcSecurityGroupIds": [
                "sg-1234"
            ],
            "VpcSubnetIds": [
                "subnetgroup1",
                "subnetgroup2"
            ],
            "Auth": "[
                {
                    "Description": "proxydescription2"
                    "AuthScheme": "SECRETS",
                    "SecretArn": "aarn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",
                    "IAMAuth": "DISABLED"
                }
            ]",
            "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",
            "Endpoint": "proxyExample2.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
            "RequireTLS": false,
            "IdleClientTimeout": 1800,
            "DebuggingLogging": false,
            "CreatedDate": "2022-01-05T16:19:33.452000+00:00",
            "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看 RDS 代理和《Amazon Aurora 用户指南》中的查看 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbProxies。

describe-db-proxy-endpoints

以下代码示例演示了如何使用 describe-db-proxy-endpoints。

Amazon CLI

描述数据库代理端点

以下 describe-db-proxy-endpoints 示例返回了有关数据库代理端点的信息。

```
aws rds describe-db-proxy-endpoints
```

输出：

```
{
    "DBProxyEndpoints": [
        {
            "DBProxyEndpointName": "proxyEndpoint1",
            "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-0123a01b12345c0ab",
            "DBProxyName": "proxyExample",
            "Status": "available",
            "VpcId": "vpc-1234567",
            "VpcSecurityGroupIds": [
                "sg-1234"
            ],
            "VpcSubnetIds": [
                "subnetgroup1",
                "subnetgroup2"
            ],
            "Endpoint": "proxyEndpoint1.endpoint.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
            "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
            "TargetRole": "READ_WRITE",
            "IsDefault": false
        },
        {
            "DBProxyEndpointName": "proxyEndpoint2",
            "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-4567a01b12345c0ab",
            "DBProxyName": "proxyExample2",
            "Status": "available",
            "VpcId": "vpc1234567",
            "VpcSecurityGroupIds": [
```

```
                "sg-5678"
            ],
            "VpcSubnetIds": [
                "subnetgroup1",
                "subnetgroup2"
            ],
            "Endpoint": "proxyEndpoint2.endpoint.proxy-cd1ef2klmnop.us-
east-1.rds.amazonaws.com",
            "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
            "TargetRole": "READ_WRITE",
            "IsDefault": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看代理端点和《Amazon Aurora 用户指南》中的创建代理端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbProxyEndpoints。

describe-db-proxy-target-groups

以下代码示例演示了如何使用 describe-db-proxy-target-groups。

Amazon CLI

描述数据库代理端点

以下 describe-db-proxy-target-groups 示例返回了有关数据库代理目标组的信息。

```
aws rds describe-db-proxy-target-groups \
    --db-proxy-name proxyExample
```

输出：

```
{
"TargetGroups":
    {
        "DBProxyName": "proxyExample",
        "TargetGroupName": "default",
        "TargetGroupArn": "arn:aws:rds:us-east-1:123456789012:target-group:prx-
tg-0123a01b12345c0ab",
        "IsDefault": true,
```

```
        "Status": "available",
        "ConnectionPoolConfig": {
            "MaxConnectionsPercent": 100,
            "MaxIdleConnectionsPercent": 50,
            "ConnectionBorrowTimeout": 120,
            "SessionPinningFilters": []
        },
        "CreatedDate": "2023-05-02T18:41:19.495000+00:00",
        "UpdatedDate": "2023-05-02T18:41:21.762000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看 RDS 代理和《Amazon Aurora 用户指南》中的查看 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbProxyTargetGroups。

`describe-db-proxy-targets`

以下代码示例演示了如何使用 `describe-db-proxy-targets`。

Amazon CLI

描述数据库代理目标

以下 `describe-db-proxy-targets` 示例返回了有关数据库代理目标的信息。

```
aws rds describe-db-proxy-targets \
    --db-proxy-name proxyExample
```

输出：

```
{
    "Targets": [
        {
            "Endpoint": "database1.ab0cd1efghij.us-east-1.rds.amazonaws.com",
            "TrackedClusterId": "database1",
            "RdsResourceId": "database1-instance-1",
            "Port": 3306,
            "Type": "RDS_INSTANCE",
            "Role": "READ_WRITE",
            "TargetHealth": {
                "State": "UNAVAILABLE",
```

```
                "Reason": "PENDING_PROXY_CAPACITY",
                "Description": "DBProxy Target is waiting for proxy to scale to
 desired capacity"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看 RDS 代理和《Amazon Aurora 用户指南》中的查看 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbProxyTargets。

`describe-db-recommendations`

以下代码示例演示了如何使用 `describe-db-recommendations`。

Amazon CLI

示例 1：列出所有数据库建议

以下 `describe-db-recommendations` 示例列出了您的 Amazon 账户中的所有数据库建议。

```
aws rds describe-db-recommendations
```

输出：

```
{
    "DBRecommendations": [
        {
            "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
            "TypeId": "config_recommendation::old_minor_version",
            "Severity": "informational",
            "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
            "Status": "active",
            "CreatedTime": "2024-02-21T23:14:19.292000+00:00",
            "UpdatedTime": "2024-02-21T23:14:19+00:00",
            "Detection": "**[resource-name]** is not running the latest minor DB
 engine version",
            "Recommendation": "Upgrade to latest engine version",
            "Description": "Your database resources aren't running the latest minor
 DB engine version. The latest minor version contains the latest security fixes and
 other improvements.",
```

```
            "RecommendedActions": [
                {
                    "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",
                    "Operation": "modifyDbInstance",
                    "Parameters": [
                        {
                            "Key": "EngineVersion",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "DBInstanceIdentifier",
                            "Value": "database-1"
                        }
                    ],
                    "ApplyModes": [
                        "immediately",
                        "next-maintenance-window"
                    ],
                    "Status": "ready",
                    "ContextAttributes": [
                        {
                            "Key": "Recommended value",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "Current engine version",
                            "Value": "5.7.42"
                        }
                    ]
                }
            ],
            "Category": "security",
            "Source": "RDS",
            "TypeDetection": "**[resource-count] resources** are not running the
latest minor DB engine version",
            "TypeRecommendation": "Upgrade to latest engine version",
            "Impact": "Reduced database performance and data security at risk",
            "AdditionalInfo": "We recommend that you maintain your database with the
latest DB engine minor version as this version includes the latest security and
functionality fixes. The DB engine minor version upgrades contain only the changes
which are backward-compatible with earlier minor versions of the same major version
of the DB engine.",
            "Links": [
                {
```

```
                    "Text": "Upgrading an RDS DB instance engine version",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
USER_UpgradeDBInstance.Upgrading.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon Aurora",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon RDS",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看和响应 Amazon RDS 建议和《Amazon Aurora 用户指南》中的查看和响应 Amazon RDS。

示例 2：列出高严重性数据库建议

以下 `describe-db-recommendations` 示例列出了您的 Amazon 账户中的高严重性数据库建议。

```
aws rds describe-db-recommendations \
    --filters Name=severity,Values=high
```

输出：

```
{
    "DBRecommendations": [
        {
            "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
            "TypeId": "config_recommendation::rds_extended_support",
            "Severity": "high",
            "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
            "Status": "active",
            "CreatedTime": "2024-02-21T23:14:19.392000+00:00",
```

```
            "UpdatedTime": "2024-02-21T23:14:19+00:00",
            "Detection": "Your databases will be auto-enrolled to RDS Extended
 Support on February 29",
            "Recommendation": "Upgrade your major version before February 29, 2024
 to avoid additional charges",
            "Description": "Your PostgreSQL 11 and MySQL 5.7 databases will be
 automatically enrolled into RDS Extended Support on February 29, 2024. To avoid
 the increase in charges due to RDS Extended Support, we recommend upgrading your
 databases to a newer major engine version before February 29, 2024.\nTo learn more
 about the RDS Extended Support pricing, refer to the pricing page.",
            "RecommendedActions": [
                {
                    "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",
                    "Parameters": [],
                    "ApplyModes": [
                        "manual"
                    ],
                    "Status": "ready",
                    "ContextAttributes": []
                }
            ],
            "Category": "cost optimization",
            "Source": "RDS",
            "TypeDetection": "Your database will be auto-enrolled to RDS Extended
 Support on February 29",
            "TypeRecommendation": "Upgrade your major version before February 29,
 2024 to avoid additional charges",
            "Impact": "Increase in charges due to RDS Extended Support",
            "AdditionalInfo": "With Amazon RDS Extended Support, you can continue
 running your database on a major engine version past the RDS end of standard
 support date for an additional cost. This paid feature gives you more time to
 upgrade to a supported major engine version.\nDuring Extended Support, Amazon RDS
 will supply critical CVE patches and bug fixes.",
            "Links": [
                {
                    "Text": "Amazon RDS Extended Support pricing for RDS for MySQL",
                    "Url": "https://aws.amazon.com/rds/mysql/pricing/"
                },
                {
                    "Text": "Amazon RDS Extended Support for RDS for MySQL and
 PostgreSQL databases",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
extended-support.html"
                },
```

```
                {
                    "Text": "Amazon RDS Extended Support pricing for Amazon Aurora
 PostgreSQL",
                    "Url": "https://aws.amazon.com/rds/aurora/pricing/"
                },
                {
                    "Text": "Amazon RDS Extended Support for Aurora PostgreSQL
 databases",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/extended-support.html"
                },
                {
                    "Text": "Amazon RDS Extended Support pricing for RDS for
 PostgreSQL",
                    "Url": "https://aws.amazon.com/rds/postgresql/pricing/"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看和响应 Amazon RDS 建议和《Amazon Aurora 用户指南》中的查看和响应 Amazon RDS。

示例 3：列出指定数据库实例的数据库建议

以下 `describe-db-recommendations` 示例列出了指定数据库实例的所有数据库建议。

```
aws rds describe-db-recommendations \
    --filters Name=dbi-resource-id,Values=database-1
```

输出：

```
{
    "DBRecommendations": [
        {
            "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
            "TypeId": "config_recommendation::old_minor_version",
            "Severity": "informational",
            "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
            "Status": "active",
            "CreatedTime": "2024-02-21T23:14:19.292000+00:00",
            "UpdatedTime": "2024-02-21T23:14:19+00:00",
```

```
            "Detection": "**[resource-name]** is not running the latest minor DB
engine version",
            "Recommendation": "Upgrade to latest engine version",
            "Description": "Your database resources aren't running the latest minor
DB engine version. The latest minor version contains the latest security fixes and
other improvements.",
            "RecommendedActions": [
                {
                    "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",
                    "Operation": "modifyDbInstance",
                    "Parameters": [
                        {
                            "Key": "EngineVersion",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "DBInstanceIdentifier",
                            "Value": "database-1"
                        }
                    ],
                    "ApplyModes": [
                        "immediately",
                        "next-maintenance-window"
                    ],
                    "Status": "ready",
                    "ContextAttributes": [
                        {
                            "Key": "Recommended value",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "Current engine version",
                            "Value": "5.7.42"
                        }
                    ]
                }
            ],
            "Category": "security",
            "Source": "RDS",
            "TypeDetection": "**[resource-count] resources** are not running the
latest minor DB engine version",
            "TypeRecommendation": "Upgrade to latest engine version",
            "Impact": "Reduced database performance and data security at risk",
```

```
            "AdditionalInfo": "We recommend that you maintain your database with the
 latest DB engine minor version as this version includes the latest security and
 functionality fixes. The DB engine minor version upgrades contain only the changes
 which are backward-compatible with earlier minor versions of the same major version
 of the DB engine.",
            "Links": [
                {
                    "Text": "Upgrading an RDS DB instance engine version",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
USER_UpgradeDBInstance.Upgrading.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon Aurora",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon RDS",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看和响应 Amazon RDS 建议和《Amazon Aurora 用户指南》中的查看和响应 Amazon RDS。

示例 4：列出所有有效的数据库建议

以下 `describe-db-recommendations` 示例列出了您的 Amazon 账户中的所有有效数据库建议。

```
aws rds describe-db-recommendations \
    --filters Name=status,Values=active
```

输出：

```
{
    "DBRecommendations": [
```

```
        {
            "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
            "TypeId": "config_recommendation::old_minor_version",
            "Severity": "informational",
            "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
            "Status": "active",
            "CreatedTime": "2024-02-21T23:14:19.292000+00:00",
            "UpdatedTime": "2024-02-21T23:14:19+00:00",
            "Detection": "**[resource-name]** is not running the latest minor DB
engine version",
            "Recommendation": "Upgrade to latest engine version",
            "Description": "Your database resources aren't running the latest minor
DB engine version. The latest minor version contains the latest security fixes and
other improvements.",
            "RecommendedActions": [
                {
                    "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",
                    "Operation": "modifyDbInstance",
                    "Parameters": [
                        {
                            "Key": "EngineVersion",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "DBInstanceIdentifier",
                            "Value": "database-1"
                        }
                    ],
                    "ApplyModes": [
                        "immediately",
                        "next-maintenance-window"
                    ],
                    "Status": "ready",
                    "ContextAttributes": [
                        {
                            "Key": "Recommended value",
                            "Value": "5.7.44"
                        },
                        {
                            "Key": "Current engine version",
                            "Value": "5.7.42"
                        }
                    ]
                }
```

```
            ],
            "Category": "security",
            "Source": "RDS",
            "TypeDetection": "**[resource-count] resources** are not running the
 latest minor DB engine version",
            "TypeRecommendation": "Upgrade to latest engine version",
            "Impact": "Reduced database performance and data security at risk",
            "AdditionalInfo": "We recommend that you maintain your database with the
 latest DB engine minor version as this version includes the latest security and
 functionality fixes. The DB engine minor version upgrades contain only the changes
 which are backward-compatible with earlier minor versions of the same major version
 of the DB engine.",
            "Links": [
                {
                    "Text": "Upgrading an RDS DB instance engine version",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
USER_UpgradeDBInstance.Upgrading.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon Aurora",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
                },
                {
                    "Text": "Using Amazon RDS Blue/Green Deployments for database
 updates for Amazon RDS",
                    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查看和响应 Amazon RDS 建议和《Amazon Aurora 用户指南》中的查看和响应 Amazon RDS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbRecommendations。

describe-db-security-groups

以下代码示例演示了如何使用 describe-db-security-groups。

Amazon CLI

列出数据库安全组

以下 describe-db-security-groups 示例列出了数据库安全组。

```
aws rds describe-db-security-groups
```

输出：

```
{
    "DBSecurityGroups": [
        {
            "OwnerId": "123456789012",
            "DBSecurityGroupName": "default",
            "DBSecurityGroupDescription": "default",
            "EC2SecurityGroups": [],
            "IPRanges": [],
            "DBSecurityGroupArn": "arn:aws:rds:us-
west-1:111122223333:secgrp:default"
        },
        {
            "OwnerId": "123456789012",
            "DBSecurityGroupName": "mysecgroup",
            "DBSecurityGroupDescription": "My Test Security Group",
            "VpcId": "vpc-1234567f",
            "EC2SecurityGroups": [],
            "IPRanges": [],
            "DBSecurityGroupArn": "arn:aws:rds:us-
west-1:111122223333:secgrp:mysecgroup"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的列出可用的数据库安全组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbSecurityGroups。

describe-db-shard-groups

以下代码示例演示了如何使用 describe-db-shard-groups。

Amazon CLI

示例 1：描述数据库分片组

以下 describe-db-shard-groups 示例将检索有关数据库分片组的详细信息。

```
aws rds describe-db-shard-groups
```

输出：

```
{
    "DBShardGroups": [
        {
            "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
            "DBShardGroupIdentifier": "limitless-test-shard-grp",
            "DBClusterIdentifier": "limitless-test-cluster",
            "MaxACU": 768.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": true,
            "Endpoint": "limitless-test-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        },
        {
            "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
            "DBShardGroupIdentifier": "my-db-shard-group",
            "DBClusterIdentifier": "my-sv2-cluster",
            "MaxACU": 768.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": false,
            "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbShardGroups。

describe-db-snapshot-attributes

以下代码示例演示了如何使用 describe-db-snapshot-attributes。

Amazon CLI

描述数据库快照的属性名称和值

以下 describe-db-snapshot-attributes 示例描述了数据库快照的属性名称和值。

```
aws rds describe-db-snapshot-attributes \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
    "DBSnapshotAttributesResult": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "123456789012",
                    "210987654321"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的共享数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbSnapshotAttributes。

describe-db-snapshots

以下代码示例演示了如何使用 describe-db-snapshots。

Amazon CLI

示例 1：描述数据库实例的数据库快照

以下 describe-db-snapshots 示例将检索有关数据库实例的数据库快照的详细信息。

```
aws rds describe-db-snapshots \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
    "DBSnapshots": [
        {
            "DBSnapshotIdentifier": "mydbsnapshot",
            "DBInstanceIdentifier": "mysqldb",
            "SnapshotCreateTime": "2018-02-08T22:28:08.598Z",
            "Engine": "mysql",
            "AllocatedStorage": 20,
            "Status": "available",
            "Port": 3306,
            "AvailabilityZone": "us-east-1f",
            "VpcId": "vpc-6594f31c",
            "InstanceCreateTime": "2018-02-08T22:24:55.973Z",
            "MasterUsername": "mysqladmin",
            "EngineVersion": "5.6.37",
            "LicenseModel": "general-public-license",
            "SnapshotType": "manual",
            "OptionGroupName": "default:mysql-5-6",
            "PercentProgress": 100,
            "StorageType": "gp2",
            "Encrypted": false,
            "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mydbsnapshot",
            "IAMDatabaseAuthenticationEnabled": false,
            "ProcessorFeatures": [],
            "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

示例 2：查找手动拍摄的快照数量

以下 describe-db-snapshots 示例使用 --query 选项中的 length 运算符来返回在特定 Amazon 区域拍摄的手动快照的数量。

```
aws rds describe-db-snapshots \
    --snapshot-type manual \
    --query "length(*[].{DBSnapshots:SnapshotType})" \
    --region eu-central-1
```

输出：

```
35
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBSnapshots。

describe-db-subnet-groups

以下代码示例演示了如何使用 describe-db-subnet-groups。

Amazon CLI

描述数据库子网组

以下 describe-db-subnet-groups 示例将检索有关指定数据库子网组的详细信息。

```
aws rds describe-db-subnet-groups
```

输出：

```
{
    "DBSubnetGroups": [
        {
            "DBSubnetGroupName": "mydbsubnetgroup",
            "DBSubnetGroupDescription": "My DB Subnet Group",
            "VpcId": "vpc-971c12ee",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-d8c8e7f4",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
```

```
                {
                    "SubnetIdentifier": "subnet-718fdc7d",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1f"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-cbc8e7e7",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-0ccde220",
                    "SubnetAvailabilityZone": {
                        "Name": "us-east-1a"
                    },
                    "SubnetStatus": "Active"
                }
            ],
            "DBSubnetGroupArn": "arn:aws:rds:us-
east-1:123456789012:subgrp:mydbsubnetgroup"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的 Amazon 虚拟私有云（VPC）和 Amazon RDS。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDbSubnetGroups。

`describe-engine-default-cluster-parameters`

以下代码示例演示了如何使用 `describe-engine-default-cluster-parameters`。

Amazon CLI

描述 Aurora 数据库引擎的默认引擎和系统参数信息

以下 `describe-engine-default-cluster-parameters` 示例检索与 MySQL 5.7 兼容的 Aurora 数据库集群的默认引擎和系统参数信息的详细信息。

```
aws rds describe-engine-default-cluster-parameters \
    --db-parameter-group-family aurora-mysql5.7
```

输出：

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "aurora_load_from_s3_role",
                "Description": "IAM role ARN used to load data from AWS S3",
                "Source": "engine-default",
                "ApplyType": "dynamic",
                "DataType": "string",
                "IsModifiable": true,
                "SupportedEngineModes": [
                    "provisioned"
                ]
            },
            ...some output truncated...
        ]
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngineDefaultClusterParameters。

describe-engine-default-parameters

以下代码示例演示了如何使用 describe-engine-default-parameters。

Amazon CLI

描述数据库引擎的默认引擎和系统参数信息

以下 describe-engine-default-parameters 示例检索 MySQL 5.7 数据库实例的默认引擎和系统参数信息的详细信息。

```
aws rds describe-engine-default-parameters \
    --db-parameter-group-family mysql5.7
```

输出：

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "allow-suspicious-udfs",
                "Description": "Controls whether user-defined functions that have
 only an xxx symbol for the main function can be loaded",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "boolean",
                "AllowedValues": "0,1",
                "IsModifiable": false
            },
            ...some output truncated...
        ]
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEngineDefaultParameters。

describe-event-categories

以下代码示例演示了如何使用 describe-event-categories。

Amazon CLI

描述事件类别

以下 describe-event-categories 示例检索有关所有可用事件源的事件类别的详细信息。

```
aws rds describe-event-categories
```

输出：

```
{
    "EventCategoriesMapList": [
        {
            "SourceType": "db-instance",
```

```
            "EventCategories": [
                "deletion",
                "read replica",
                "failover",
                "restoration",
                "maintenance",
                "low storage",
                "configuration change",
                "backup",
                "creation",
                "availability",
                "recovery",
                "failure",
                "backtrack",
                "notification"
            ]
        },
        {
            "SourceType": "db-security-group",
            "EventCategories": [
                "configuration change",
                "failure"
            ]
        },
        {
            "SourceType": "db-parameter-group",
            "EventCategories": [
                "configuration change"
            ]
        },
        {
            "SourceType": "db-snapshot",
            "EventCategories": [
                "deletion",
                "creation",
                "restoration",
                "notification"
            ]
        },
        {
            "SourceType": "db-cluster",
            "EventCategories": [
                "failover",
                "failure",
```

```
                "notification"
            ]
        },
        {
            "SourceType": "db-cluster-snapshot",
            "EventCategories": [
                "backup"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventCategories。

describe-event-subscriptions

以下代码示例演示了如何使用 describe-event-subscriptions。

Amazon CLI

描述事件订阅

此示例描述了当前 Amazon 账户的所有 Amazon RDS 事件订阅。

```
aws rds describe-event-subscriptions
```

输出：

```
{
    "EventSubscriptionsList": [
        {
            "EventCategoriesList": [
                "backup",
                "recovery"
            ],
            "Enabled": true,
            "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-
instance-events",
            "Status": "creating",
            "SourceType": "db-instance",
            "CustomerAwsId": "123456789012",
            "SubscriptionCreationTime": "2018-07-31 23:22:01.893",
            "CustSubscriptionId": "my-instance-events",
```

```
            "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events"
        },
        ...some output truncated...
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventSubscriptions。

describe-events

以下代码示例演示了如何使用 describe-events。

Amazon CLI

描述事件

以下 describe-events 示例检索指定数据库实例发生的事件的详细信息。

```
aws rds describe-events \
    --source-identifier test-instance \
    --source-type db-instance
```

输出：

```
{
    "Events": [
        {
            "SourceType": "db-instance",
            "SourceIdentifier": "test-instance",
            "EventCategories": [
                "backup"
            ],
            "Message": "Backing up DB instance",
            "Date": "2018-07-31T23:09:23.983Z",
            "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"
        },
        {
            "SourceType": "db-instance",
            "SourceIdentifier": "test-instance",
            "EventCategories": [
                "backup"
            ],
```

```
            "Message": "Finished DB Instance backup",
            "Date": "2018-07-31T23:15:13.049Z",
            "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-export-tasks

以下代码示例演示了如何使用 describe-export-tasks。

Amazon CLI

描述快照导出任务

以下 describe-export-tasks 示例返回有关将快照导出到 Amazon S3 的信息。

```
aws rds describe-export-tasks
```

输出：

```
{
    "ExportTasks": [
        {
            "ExportTaskIdentifier": "test-snapshot-export",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:test-
snapshot",
            "SnapshotTime": "2020-03-02T18:26:28.163Z",
            "TaskStartTime": "2020-03-02T18:57:56.896Z",
            "TaskEndTime": "2020-03-02T19:10:31.985Z",
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Prefix": "",
            "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",
            "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
abcd0000-7fca-4128-82f2-aabbccddeeff",
            "Status": "COMPLETE",
            "PercentProgress": 100,
            "TotalExtractedDataInGB": 0
        },
        {
```

```
            "ExportTaskIdentifier": "my-s3-export",
            "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-
test",
            "SnapshotTime": "2020-03-27T20:48:42.023Z",
            "S3Bucket": "amzn-s3-demo-bucket",
            "S3Prefix": "",
            "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",
            "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
abcd0000-7fca-4128-82f2-aabbccddeeff",
            "Status": "STARTING",
            "PercentProgress": 0,
            "TotalExtractedDataInGB": 0
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的监控快照导出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeExportTasks。

describe-global-clusters

以下代码示例演示了如何使用 describe-global-clusters。

Amazon CLI

描述全局数据库集群

以下 describe-global-clusters 示例列出了当前 Amazon 区域中的 Aurora 全局数据库集群。

```
aws rds describe-global-clusters
```

输出：

```
{
    "GlobalClusters": [
        {
            "GlobalClusterIdentifier": "myglobalcluster",
            "GlobalClusterResourceId": "cluster-f5982077e3b5aabb",
            "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
            "Status": "available",
```

```
            "Engine": "aurora-mysql",
            "EngineVersion": "5.7.mysql_aurora.2.07.2",
            "StorageEncrypted": false,
            "DeletionProtection": false,
            "GlobalClusterMembers": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的管理 Aurora 全局数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGlobalClusters。

describe-option-group-options

以下代码示例演示了如何使用 describe-option-group-options。

Amazon CLI

描述所有可用的选项

以下 describe-option-group-options 示例列出了 Oracle Database 19c 实例的两个选项。

```
aws rds describe-option-group-options \
    --engine-name oracle-ee \
    --major-engine-version 19 \
    --max-items 2
```

输出：

```
{
    "OptionGroupOptions": [
        {
            "Name": "APEX",
            "Description": "Oracle Application Express Runtime Environment",
            "EngineName": "oracle-ee",
            "MajorEngineVersion": "19",
            "MinimumRequiredMinorEngineVersion": "0.0.0.ru-2019-07.rur-2019-07.r1",
            "PortRequired": false,
            "OptionsDependedOn": [],
            "OptionsConflictsWith": [],
            "Persistent": false,
            "Permanent": false,
```

```
            "RequiresAutoMinorEngineVersionUpgrade": false,
            "VpcOnly": false,
            "SupportsOptionVersionDowngrade": false,
            "OptionGroupOptionSettings": [],
            "OptionGroupOptionVersions": [
                {
                    "Version": "19.1.v1",
                    "IsDefault": true
                },
                {
                    "Version": "19.2.v1",
                    "IsDefault": false
                }
            ]
        },
        {
            "Name": "APEX-DEV",
            "Description": "Oracle Application Express Development Environment",
            "EngineName": "oracle-ee",
            "MajorEngineVersion": "19",
            "MinimumRequiredMinorEngineVersion": "0.0.0.ru-2019-07.rur-2019-07.r1",
            "PortRequired": false,
            "OptionsDependedOn": [
                "APEX"
            ],
            "OptionsConflictsWith": [],
            "Persistent": false,
            "Permanent": false,
            "RequiresAutoMinorEngineVersionUpgrade": false,
            "VpcOnly": false,
            "OptionGroupOptionSettings": []
        }
    ],
    "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的列出选项组的选项和选项设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOptionGroupOptions。

describe-option-groups

以下代码示例演示了如何使用 describe-option-groups。

Amazon CLI

描述可用的选项组

以下 describe-option-groups 示例列出了 Oracle Database 19c 实例的选项组。

```
aws rds describe-option-groups \
    --engine-name oracle-ee \
    --major-engine-version 19
```

输出：

```
{
    "OptionGroupsList": [
        {
            "OptionGroupName": "default:oracle-ee-19",
            "OptionGroupDescription": "Default option group for oracle-ee 19",
            "EngineName": "oracle-ee",
            "MajorEngineVersion": "19",
            "Options": [],
            "AllowsVpcAndNonVpcInstanceMemberships": true,
            "OptionGroupArn": "arn:aws:rds:us-west-1:111122223333:og:default:oracle-
ee-19"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的列出选项组的选项和选项设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOptionGroups。

describe-orderable-db-instance-options

以下代码示例演示了如何使用 describe-orderable-db-instance-options。

Amazon CLI

描述可订购的数据库实例选项

以下 describe-orderable-db-instance-options 示例将检索有关运行 MySQL 数据库引擎的数据库实例的可订购选项的详细信息。

```
aws rds describe-orderable-db-instance-options \
    --engine mysql
```

输出：

```
{
    "OrderableDBInstanceOptions": [
        {
            "MinStorageSize": 5,
            "ReadReplicaCapable": true,
            "MaxStorageSize": 6144,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "SupportsIops": false,
            "AvailableProcessorFeatures": [],
            "MultiAZCapable": true,
            "DBInstanceClass": "db.m1.large",
            "Vpc": true,
            "StorageType": "gp2",
            "LicenseModel": "general-public-license",
            "EngineVersion": "5.5.46",
            "SupportsStorageEncryption": false,
            "SupportsEnhancedMonitoring": true,
            "Engine": "mysql",
            "SupportsIAMDatabaseAuthentication": false,
            "SupportsPerformanceInsights": false
        }
    ]
    ...some output truncated...
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrderableDBInstanceOptions。

describe-pending-maintenance-actions

以下代码示例演示了如何使用 describe-pending-maintenance-actions。

Amazon CLI

列出至少包含一项待处理维护操作的资源

以下 describe-pending-maintenance-actions 示例列出了数据库实例的待处理维护操作。

```
aws rds describe-pending-maintenance-actions
```

输出：

```
{
    "PendingMaintenanceActions": [
        {
            "ResourceIdentifier": "arn:aws:rds:us-
west-2:123456789012:cluster:global-db1-cl1",
            "PendingMaintenanceActionDetails": [
                {
                    "Action": "system-update",
                    "Description": "Upgrade to Aurora PostgreSQL 2.4.2"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的维护数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePendingMaintenanceActions。

describe-reserved-db-instances-offerings

以下代码示例演示了如何使用 describe-reserved-db-instances-offerings。

Amazon CLI

描述预留的数据库实例服务

以下 describe-reserved-db-instances-offerings 示例检索有关 oracle 的预留数据库实例选项的详细信息。

```
aws rds describe-reserved-db-instances-offerings \
    --product-description oracle
```

输出：

```
{
    "ReservedDBInstancesOfferings": [
        {
            "CurrencyCode": "USD",
            "UsagePrice": 0.0,
            "ProductDescription": "oracle-se2(li)",
            "ReservedDBInstancesOfferingId": "005bdee3-9ef4-4182-aa0c-58ef7cb6c2f8",
            "MultiAZ": true,
            "DBInstanceClass": "db.m4.xlarge",
            "OfferingType": "Partial Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.594,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "FixedPrice": 4089.0,
            "Duration": 31536000
        },
    ...some output truncated...
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedDbInstancesOfferings。

describe-reserved-db-instances

以下代码示例演示了如何使用 describe-reserved-db-instances。

Amazon CLI

描述预留的数据库实例

以下 `describe-reserved-db-instances` 示例检索有关当前 Amazon 账户中任意预留数据库实例的详细信息。

```
aws rds describe-reserved-db-instances
```

输出：

```
{
    "ReservedDBInstances": [
        {
            "ReservedDBInstanceId": "myreservedinstance",
            "ReservedDBInstancesOfferingId": "12ab34cd-59af-4b2c-a660-1abcdef23456",
            "DBInstanceClass": "db.t3.micro",
            "StartTime": "2020-06-01T13:44:21.436Z",
            "Duration": 31536000,
            "FixedPrice": 0.0,
            "UsagePrice": 0.0,
            "CurrencyCode": "USD",
            "DBInstanceCount": 1,
            "ProductDescription": "sqlserver-ex(li)",
            "OfferingType": "No Upfront",
            "MultiAZ": false,
            "State": "payment-pending",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.014,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "ReservedDBInstanceArn": "arn:aws:rds:us-
west-2:123456789012:ri:myreservedinstance",
            "LeaseId": "a1b2c3d4-6b69-4a59-be89-5e11aa446666"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的 Amazon RDS 的预留数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedDbInstances。

describe-source-regions

以下代码示例演示了如何使用 describe-source-regions。

Amazon CLI

描述源区域

以下 describe-source-regions 示例检索有关所有源 Amazon 区域的详细信息。它还显示，自动备份只能从美国西部（俄勒冈州）复制到目标 Amazon 区域，即美国东部（弗吉尼亚州北部）。

```
aws rds describe-source-regions \
    --region us-east-1
```

输出：

```
{
    "SourceRegions": [
        {
            "RegionName": "af-south-1",
            "Endpoint": "https://rds.af-south-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "ap-east-1",
            "Endpoint": "https://rds.ap-east-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "ap-northeast-1",
            "Endpoint": "https://rds.ap-northeast-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "ap-northeast-2",
            "Endpoint": "https://rds.ap-northeast-2.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
```

```
        {
            "RegionName": "ap-northeast-3",
            "Endpoint": "https://rds.ap-northeast-3.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "ap-south-1",
            "Endpoint": "https://rds.ap-south-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "ap-southeast-1",
            "Endpoint": "https://rds.ap-southeast-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "ap-southeast-2",
            "Endpoint": "https://rds.ap-southeast-2.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "ap-southeast-3",
            "Endpoint": "https://rds.ap-southeast-3.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "ca-central-1",
            "Endpoint": "https://rds.ca-central-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "eu-north-1",
            "Endpoint": "https://rds.eu-north-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "eu-south-1",
```

```
            "Endpoint": "https://rds.eu-south-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "eu-west-1",
            "Endpoint": "https://rds.eu-west-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "eu-west-2",
            "Endpoint": "https://rds.eu-west-2.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "eu-west-3",
            "Endpoint": "https://rds.eu-west-3.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "me-central-1",
             "Endpoint": "https://rds.me-central-1.amazonaws.com",
             "Status": "available",
             "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "me-south-1",
            "Endpoint": "https://rds.me-south-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
        {
            "RegionName": "sa-east-1",
            "Endpoint": "https://rds.sa-east-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "us-east-2",
            "Endpoint": "https://rds.us-east-2.amazonaws.com",
            "Status": "available",
```

```
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "us-west-1",
            "Endpoint": "https://rds.us-west-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
        {
            "RegionName": "us-west-2",
           "Endpoint": "https://rds.us-west-2.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的查找有关复制备份的信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSourceRegions。

describe-valid-db-instance-modifications

以下代码示例演示了如何使用 describe-valid-db-instance-modifications。

Amazon CLI

描述数据库实例的有效修改

以下 describe-valid-db-instance-modifications 示例将检索有关指定数据库实例的有效修改的详细信息。

```
aws rds describe-valid-db-instance-modifications \
    --db-instance-identifier test-instance
```

输出：

```
{
    "ValidDBInstanceModificationsMessage": {
        "ValidProcessorFeatures": [],
        "Storage": [
            {
                "StorageSize": [
```

```
                    {
                        "Step": 1,
                        "To": 20,
                        "From": 20
                    },
                    {
                        "Step": 1,
                        "To": 6144,
                        "From": 22
                    }
                ],
                "ProvisionedIops": [
                    {
                        "Step": 1,
                        "To": 0,
                        "From": 0
                    }
                ],
                "IopsToStorageRatio": [
                    {
                        "To": 0.0,
                        "From": 0.0
                    }
                ],
                "StorageType": "gp2"
            },
            {
                "StorageSize": [
                    {
                        "Step": 1,
                        "To": 6144,
                        "From": 100
                    }
                ],
                "ProvisionedIops": [
                    {
                        "Step": 1,
                        "To": 40000,
                        "From": 1000
                    }
                ],
                "IopsToStorageRatio": [
                    {
                        "To": 50.0,
```

```
                        "From": 1.0
                    }
                ],
                "StorageType": "io1"
            },
            {
                "StorageSize": [
                    {
                        "Step": 1,
                        "To": 20,
                        "From": 20
                    },
                    {
                        "Step": 1,
                        "To": 3072,
                        "From": 22
                    }
                ],
                "ProvisionedIops": [
                    {
                        "Step": 1,
                        "To": 0,
                        "From": 0
                    }
                ],
                "IopsToStorageRatio": [
                    {
                        "To": 0.0,
                        "From": 0.0
                    }
                ],
                "StorageType": "magnetic"
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeValidDbInstanceModifications。

download-db-log-file-portion

以下代码示例演示了如何使用 download-db-log-file-portion。

Amazon CLI

示例 1：下载数据库日志文件的最新部分

以下 download-db-log-file-portion 示例仅下载日志文件的最新部分，将其保存到名为 tail.txt 的本地文件中。

```
aws rds download-db-log-file-portion \
    --db-instance-identifier test-instance \
    --log-file-name log.txt \
    --output text > tail.txt
```

保存的文件可能包含空行。下载时，它们会出现在日志文件每个部分的末尾。

示例 2：下载完整的数据库日志文件

以下 download-db-log-file-portion 示例使用 --starting-token 0 参数下载完整日志文件，并将输出保存到名为 full.txt 的本地文件中。

```
aws rds download-db-log-file-portion \
    --db-instance-identifier test-instance \
    --log-file-name log.txt \
    --starting-token 0 \
    --output text > full.txt
```

保存的文件可能包含空行。下载时，它们会出现在日志文件每个部分的末尾。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DownloadDbLogFilePortion。

generate-auth-token

以下代码示例演示了如何使用 generate-auth-token。

Amazon CLI

生成身份验证令牌

以下 generate-db-auth-token 示例生成与 IAM 数据库身份验证一起使用的身份验证令牌。

```
aws rds generate-db-auth-token \
    --hostname aurmysql-test.cdgmuqiadpid.us-west-2.rds.amazonaws.com \
    --port 3306 \
```

```
    --region us-east-1 \
    --username jane_doe
```

输出：

```
aurmysql-test.cdgmuqiadpid.us-west-2.rds.amazonaws.com:3306/?
Action=connect&DBUser=jane_doe&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIESZCNJ3OEXAMPLE%2F20180731%2Fus-east-1%2Frds-db%2Faws4_request&X-
Amz-Date=20180731T235209Z&X-Amz-Expires=900&X-Amz-SignedHeaders=host&X-Amz-
Signature=5a8753ebEXAMPLEa2c724e5667797EXAMPLE9d6ec6e3f427191fa41aeEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateAuthToken。

generate-db-auth-token

以下代码示例演示了如何使用 generate-db-auth-token。

Amazon CLI

生成 IAM 身份验证令牌

以下 generate-db-auth-token 示例生成了用于连接到数据库的 IAM 身份验证令牌。

```
aws rds generate-db-auth-token \
    --hostname mydb.123456789012.us-east-1.rds.amazonaws.com \
    --port 3306 \
    --region us-east-1 \
    --username db_user
```

输出：

```
mydb.123456789012.us-east-1.rds.amazonaws.com:3306/?
Action=connect&DBUser=db_user&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIEXAMPLE%2Fus-east-1%2Frds-db%2Faws4_request&X-Amz-
Date=20210123T011543Z&X-Amz-Expires=900&X-Amz-SignedHeaders=host&X-Amz-
Signature=88987EXAMPLE1EXAMPLE2EXAMPLE3EXAMPLE4EXAMPLE5EXAMPLE6
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 IAM 身份验证连接到数据库实例和《Amazon Aurora 用户指南》中的使用 IAM 身份验证连接到数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GenerateDbAuthToken。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出 Amazon RDS 资源上的标签

以下 list-tags-for-resource 示例列出了数据库实例上的所有标签。

```
aws rds list-tags-for-resource \
    --resource-name arn:aws:rds:us-east-1:123456789012:db:orcl1
```

输出：

```
{
    "TagList": [
        {
            "Key": "Environment",
            "Value": "test"
        },
        {
            "Key": "Name",
            "Value": "MyDatabase"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的为 Amazon RDS 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

modify-certificates

以下代码示例演示了如何使用 modify-certificates。

Amazon CLI

临时覆盖新数据库实例的系统默认 SSL/TLS 证书

以下 modify-certificates 示例临时覆盖新数据库实例的系统默认 SSL/TLS 证书。

```
aws rds modify-certificates \
```

```
    --certificate-identifier rds-ca-2019
```

输出：

```
{
    "Certificate": {
        "CertificateIdentifier": "rds-ca-2019",
        "CertificateType": "CA",
        "Thumbprint": "EXAMPLE123456789012",
        "ValidFrom": "2019-09-19T18:16:53Z",
        "ValidTill": "2024-08-22T17:08:50Z",
        "CertificateArn": "arn:aws:rds:us-east-1::cert:rds-ca-2019",
        "CustomerOverride": true,
        "CustomerOverrideValidTill": "2024-08-22T17:08:50Z"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的轮换 SSL/TLS 证书和《Amazon Aurora 用户指南》中的轮换 SSL/TLS 证书。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCertificates。

modify-current-db-cluster-capacity

以下代码示例演示了如何使用 modify-current-db-cluster-capacity。

Amazon CLI

扩展 Aurora Serverless 数据库集群的容量

以下 modify-current-db-cluster-capacity 示例将 Aurora Serverless 数据库集群的容量扩展到 8。

```
aws rds modify-current-db-cluster-capacity \
    --db-cluster-identifier mydbcluster \
    --capacity 8
```

输出：

```
{
    "DBClusterIdentifier": "mydbcluster",
    "PendingCapacity": 8,
```

```
    "CurrentCapacity": 1,
    "SecondsBeforeTimeout": 300,
    "TimeoutAction": "ForceApplyCapacityChange"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的手动扩展 Aurora Serverless v1 数据库集群容量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCurrentDbClusterCapacity。

modify-db-cluster-endpoint

以下代码示例演示了如何使用 modify-db-cluster-endpoint。

Amazon CLI

修改自定义数据库集群端点

以下 modify-db-cluster-endpoint 示例修改了指定的自定义数据库集群端点。

```
aws rds modify-db-cluster-endpoint \
    --db-cluster-endpoint-identifier mycustomendpoint \
    --static-members dbinstance1 dbinstance2 dbinstance3
```

输出：

```
{
    "DBClusterEndpointIdentifier": "mycustomendpoint",
    "DBClusterIdentifier": "mydbcluster",
    "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",
    "Endpoint": "mycustomendpoint.cluster-custom-cnpexample.us-
east-1.rds.amazonaws.com",
    "Status": "modifying",
    "EndpointType": "CUSTOM",
    "CustomEndpointType": "READER",
    "StaticMembers": [
        "dbinstance1",
        "dbinstance2",
        "dbinstance3"
    ],
    "ExcludedMembers": [],
    "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:mycustomendpoint"
```

```
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 连接管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbClusterEndpoint。

modify-db-cluster-parameter-group

以下代码示例演示了如何使用 modify-db-cluster-parameter-group。

Amazon CLI

修改数据库集群参数组中的参数

以下 modify-db-cluster-parameter-group 示例修改了数据库集群参数组中的参数的值。

```
aws rds modify-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterpg \
    --
parameters "ParameterName=server_audit_logging,ParameterValue=1,ApplyMethod=immediate"
 \

  "ParameterName=server_audit_logs_upload,ParameterValue=1,ApplyMethod=immediate"
```

输出：

```
{
    "DBClusterParameterGroupName": "mydbclusterpg"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbClusterParameterGroup。

modify-db-cluster-snapshot-attribute

以下代码示例演示了如何使用 modify-db-cluster-snapshot-attribute。

Amazon CLI

修改数据库集群快照属性

以下 modify-db-cluster-snapshot-attribute 示例对指定的数据库集群快照属性进行了更改。

```
aws rds modify-db-cluster-snapshot-attribute \
    --db-cluster-snapshot-identifier myclustersnapshot \
    --attribute-name restore \
    --values-to-add 123456789012
```

输出：

```
{
    "DBClusterSnapshotAttributesResult": {
        "DBClusterSnapshotIdentifier": "myclustersnapshot",
        "DBClusterSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "123456789012"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的从数据库集群快照还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbClusterSnapshotAttribute。

modify-db-cluster

以下代码示例演示了如何使用 modify-db-cluster。

Amazon CLI

示例 1：修改数据库集群

以下 modify-db-cluster 示例更改了名为 cluster-2 的数据库集群的主用户密码，并将备份保留期设置为 14 天。--apply-immediately 参数使得可以立即进行更改，而不是等到下一个维护时段。

```
aws rds modify-db-cluster \
```

```
    --db-cluster-identifier cluster-2 \
    --backup-retention-period 14 \
    --master-user-password newpassword99 \
    --apply-immediately
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "eu-central-1b",
            "eu-central-1c",
            "eu-central-1a"
        ],
        "BackupRetentionPeriod": 14,
        "DatabaseName": "",
        "DBClusterIdentifier": "cluster-2",
        "DBClusterParameterGroup": "default.aurora5.6",
        "DBSubnetGroup": "default-vpc-2305ca49",
        "Status": "available",
        "EarliestRestorableTime": "2020-06-03T02:07:29.637Z",
        "Endpoint": "cluster-2.cluster-############.eu-central-1.rds.amazonaws.com",
        "ReaderEndpoint": "cluster-2.cluster-ro-############.eu-
central-1.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora",
        "EngineVersion": "5.6.10a",
        "LatestRestorableTime": "2020-06-04T15:11:25.748Z",
        "Port": 3306,
        "MasterUsername": "admin",
        "PreferredBackupWindow": "01:55-02:25",
        "PreferredMaintenanceWindow": "thu:21:14-thu:21:44",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [
            {
                "DBInstanceIdentifier": "cluster-2-instance-1",
                "IsClusterWriter": true,
                "DBClusterParameterGroupStatus": "in-sync",
                "PromotionTier": 1
            }
        ],
        "VpcSecurityGroups": [
```

```
            {
                "VpcSecurityGroupId": "sg-20a5c047",
                "Status": "active"
            }
        ],
        "HostedZoneId": "Z1RLNU0EXAMPLE",
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:eu-central-1:123456789012:key/
d1bd7c8f-5cdb-49ca-8a62-a1b2c3d4e5f6",
        "DbClusterResourceId": "cluster-AGJ7XI77XVIS6FUXHU1EXAMPLE",
        "DBClusterArn": "arn:aws:rds:eu-central-1:123456789012:cluster:cluster-2",
        "AssociatedRoles": [],
        "IAMDatabaseAuthenticationEnabled": false,
        "ClusterCreateTime": "2020-04-03T14:44:02.764Z",
        "EngineMode": "provisioned",
        "DeletionProtection": false,
        "HttpEndpointEnabled": false,
        "CopyTagsToSnapshot": true,
        "CrossAccountClone": false,
        "DomainMemberships": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的修改 Amazon Aurora 数据库集群。

示例 2：将 VPC 安全组与数据库集群关联

以下 `modify-db-instance` 示例关联特定 VPC 安全组并从数据库集群中移除数据库安全组。

```
aws rds modify-db-cluster \
    --db-cluster-identifier dbName \
    --vpc-security-group-ids sg-ID
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-west-2c",
            "us-west-2b",
            "us-west-2a"
        ],
```

```
        "BackupRetentionPeriod": 1,
        "DBClusterIdentifier": "dbName",
        "DBClusterParameterGroup": "default.aurora-mysql8.0",
        "DBSubnetGroup": "default",
        "Status": "available",
        "EarliestRestorableTime": "2024-02-15T01:12:13.966000+00:00",
        "Endpoint": "dbName.cluster-abcdefghji.us-west-2.rds.amazonaws.com",
        "ReaderEndpoint": "dbName.cluster-ro-abcdefghji.us-
west-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-mysql",
        "EngineVersion": "8.0.mysql_aurora.3.04.1",
        "LatestRestorableTime": "2024-02-15T02:25:33.696000+00:00",
        "Port": 3306,
        "MasterUsername": "admin",
        "PreferredBackupWindow": "10:59-11:29",
        "PreferredMaintenanceWindow": "thu:08:54-thu:09:24",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [
            {
                "DBInstanceIdentifier": "dbName-instance-1",
                "IsClusterWriter": true,
                "DBClusterParameterGroupStatus": "in-sync",
                "PromotionTier": 1
            }
        ],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-ID",
                "Status": "active"
            }
        ],
        ...output omitted...
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用安全组控制访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbCluster。

`modify-db-instance`

以下代码示例演示了如何使用 `modify-db-instance`。

Amazon CLI

示例 1：修改数据库实例

以下 modify-db-instance 示例将选项组和参数组与兼容的 Microsoft SQL Server 数据库实例相关联。--apply-immediately 参数使选项和参数组立即关联，而不是等到下一个维护时段。

```
aws rds modify-db-instance \
    --db-instance-identifier database-2 \
    --option-group-name test-se-2017 \
    --db-parameter-group-name test-sqlserver-se-2017 \
    --apply-immediately
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "database-2",
        "DBInstanceClass": "db.r4.large",
        "Engine": "sqlserver-se",
        "DBInstanceStatus": "available",

        ...output omitted...

        "DBParameterGroups": [
            {
                "DBParameterGroupName": "test-sqlserver-se-2017",
                "ParameterApplyStatus": "applying"
            }
        ],
        "AvailabilityZone": "us-west-2d",

        ...output omitted...

        "MultiAZ": true,
        "EngineVersion": "14.00.3281.6.v1",
        "AutoMinorVersionUpgrade": false,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "license-included",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "test-se-2017",
                "Status": "pending-apply"
```

```
            }
        ],
        "CharacterSetName": "SQL_Latin1_General_CP1_CI_AS",
        "SecondaryAvailabilityZone": "us-west-2c",
        "PubliclyAccessible": true,
        "StorageType": "gp2",

        ...output omitted...

        "DeletionProtection": false,
        "AssociatedRoles": [],
        "MaxAllocatedStorage": 1000
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改 Amazon RDS 数据库实例。

示例 2：将 VPC 安全组与数据库实例关联

以下 modify-db-instance 示例关联特定 VPC 安全组并从数据库实例中移除数据库安全组：

```
aws rds modify-db-instance \
    --db-instance-identifier dbName \
    --vpc-security-group-ids sg-ID
```

输出：

```
{
"DBInstance": {
    "DBInstanceIdentifier": "dbName",
    "DBInstanceClass": "db.t3.micro",
    "Engine": "mysql",
    "DBInstanceStatus": "available",
    "MasterUsername": "admin",
    "Endpoint": {
        "Address": "dbName.abcdefghijk.us-west-2.rds.amazonaws.com",
        "Port": 3306,
        "HostedZoneId": "ABCDEFGHIJK1234"
    },
    "AllocatedStorage": 20,
    "InstanceCreateTime": "2024-02-15T00:37:58.793000+00:00",
    "PreferredBackupWindow": "11:57-12:27",
    "BackupRetentionPeriod": 7,
```

```
    "DBSecurityGroups": [],
    "VpcSecurityGroups": [
        {
            "VpcSecurityGroupId": "sg-ID",
            "Status": "active"
        }
    ],
    ... output omitted ...
    "MultiAZ": false,
    "EngineVersion": "8.0.35",
    "AutoMinorVersionUpgrade": true,
    "ReadReplicaDBInstanceIdentifiers": [],
    "LicenseModel": "general-public-license",

    ... output ommited ...
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用安全组控制访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDBInstance。

modify-db-parameter-group

以下代码示例演示了如何使用 modify-db-parameter-group。

Amazon CLI

修改数据库参数组

以下 modify-db-parameter-group 示例将更改数据库参数组中 clr enabled 参数的值。--apply-immediately 参数使数据库参数组得到立即修改，而不是等到下一个维护时段。

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name test-sqlserver-se-2017 \
    --parameters "ParameterName='clr
 enabled',ParameterValue=1,ApplyMethod=immediate"
```

输出：

```
{
    "DBParameterGroupName": "test-sqlserver-se-2017"
```

```
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改数据库参数组中的参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDBParameterGroup。

modify-db-proxy-endpoint

以下代码示例演示了如何使用 modify-db-proxy-endpoint。

Amazon CLI

修改 RDS 数据库的数据库代理端点

以下 modify-db-proxy-endpoint 示例修改了数据库代理端点 proxyEndpoint，以将读取超时设置为 65 秒。

```
aws rds modify-db-proxy-endpoint \
    --db-proxy-endpoint-name proxyEndpoint \
    --cli-read-timeout 65
```

输出：

```
{
"DBProxyEndpoint":
    {
        "DBProxyEndpointName": "proxyEndpoint",
        "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-0123a01b12345c0ab",
        "DBProxyName": "proxyExample",
        "Status": "available",
        "VpcId": "vpc-1234567",
        "VpcSecurityGroupIds": [
            "sg-1234"
        ],
        "VpcSubnetIds": [
            "subnetgroup1",
            "subnetgroup2"
        ],
        "Endpoint": "proxyEndpoint.endpoint.proxyExample-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
        "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
        "TargetRole": "READ_WRITE",
```

```
        "IsDefault": "false"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改代理端点和《Amazon Aurora 用户指南》中的修改代理端点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbProxyEndpoint。

modify-db-proxy-target-group

以下代码示例演示了如何使用 modify-db-proxy-target-group。

Amazon CLI

修改数据库代理端点

以下 modify-db-proxy-target-group 示例修改了数据库代理目标组，将最大连接数设置为 80，并将最大空闲连接数设置为 10。

```
aws rds modify-db-proxy-target-group \
    --target-group-name default \
    --db-proxy-name proxyExample \
    --connection-pool-config MaxConnectionsPercent=80,MaxIdleConnectionsPercent=10
```

输出：

```
{
"DBProxyTargetGroup":
    {
        "DBProxyName": "proxyExample",
        "TargetGroupName": "default",
        "TargetGroupArn": "arn:aws:rds:us-east-1:123456789012:target-group:prx-
tg-0123a01b12345c0ab",
        "IsDefault": true,
        "Status": "available",
        "ConnectionPoolConfig": {
            "MaxConnectionsPercent": 80,
            "MaxIdleConnectionsPercent": 10,
            "ConnectionBorrowTimeout": 120,
            "SessionPinningFilters": []
        },
```

```
        "CreatedDate": "2023-05-02T18:41:19.495000+00:00",
        "UpdatedDate": "2023-05-02T18:41:21.762000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改 RDS 代理和《Amazon Aurora 用户指南》中的修改 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbProxyTargetGroup。

`modify-db-proxy`

以下代码示例演示了如何使用 `modify-db-proxy`。

Amazon CLI

修改 RDS 数据库的数据库代理

以下 `modify-db-proxy` 示例修改了名为 `proxyExample` 的数据库代理，使其连接需要 SSL。

```
aws rds modify-db-proxy \
    --db-proxy-name proxyExample \
    --require-tls
```

输出：

```
{
"DBProxy":
    {
        "DBProxyName": "proxyExample",
        "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
        "Status": "modifying"
        "EngineFamily": "PostgreSQL",
        "VpcId": "sg-1234567",
        "VpcSecurityGroupIds": [
            "sg-1234"
        ],
        "VpcSubnetIds": [
            "subnetgroup1",
            "subnetgroup2"
        ],
        "Auth": "[
```

```
            {
                "Description": "proxydescription1",
                "AuthScheme": "SECRETS",
                "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:proxysecret1-Abcd1e",
                "IAMAuth": "DISABLED"
            }
        ]",
        "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",
        "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-east-1.rds.amazonaws.com",
        "RequireTLS": true,
        "IdleClientTimeout": 1800,
        "DebuggingLogging": false,
        "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
        "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改 RDS 代理和《Amazon Aurora 用户指南》中的创建 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbProxy。

modify-db-shard-group

以下代码示例演示了如何使用 modify-db-shard-group。

Amazon CLI

示例 1：修改数据库分片组

以下 modify-db-shard-group 示例更改了数据库分片组的最大容量。

```
aws rds modify-db-shard-group \
    --db-shard-group-identifier my-db-shard-group \
    --max-acu 1000
```

输出：

```
{
    "DBShardGroups": [
        {
            "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
```

```
            "DBShardGroupIdentifier": "my-db-shard-group",
            "DBClusterIdentifier": "my-sv2-cluster",
            "MaxACU": 768.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": false,
            "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 数据库集群。

示例 2：描述您的数据库分片组

以下 `describe-db-shard-groups` 示例将在您运行 `modify-db-shard-group` 命令后检索数据库分片组的详细信息。现在，数据库分片组 `my-db-shard-group` 的最大容量为 1000 个 Aurora 容量单元（ACU）。

```
aws rds describe-db-shard-groups
```

输出：

```
{
    "DBShardGroups": [
        {
            "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
            "DBShardGroupIdentifier": "limitless-test-shard-grp",
            "DBClusterIdentifier": "limitless-test-cluster",
            "MaxACU": 768.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": true,
            "Endpoint": "limitless-test-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        },
        {
            "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
            "DBShardGroupIdentifier": "my-db-shard-group",
            "DBClusterIdentifier": "my-sv2-cluster",
            "MaxACU": 1000.0,
```

```
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": false,
            "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbShardGroup。

modify-db-snapshot-attribute

以下代码示例演示了如何使用 modify-db-snapshot-attribute。

Amazon CLI

示例 1：允许两个 Amazon 账户还原数据库快照

以下 modify-db-snapshot-attribute 示例将向两个 Amazon 账户（标识符为 111122223333 和 444455556666）授予还原名为 mydbsnapshot 的数据库快照的权限。

```
aws rds modify-db-snapshot-attribute \
    --db-snapshot-identifier mydbsnapshot \
    --attribute-name restore \
    --values-to-add {"111122223333","444455556666"}
```

输出：

```
{
    "DBSnapshotAttributesResult": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "111122223333",
                    "444455556666"
                ]
            }
        ]
```

```
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的共享快照。

示例 2：防止 Amazon 账户还原数据库快照

以下 modify-db-snapshot-attribute 示例删除了特定 Amazon 账户还原名为 mydbsnapshot 的数据库快照的权限。指定单个账户时，账户标识符不能用引号或大括号括起来。

```
aws rds modify-db-snapshot-attribute \
    --db-snapshot-identifier mydbsnapshot \
    --attribute-name restore \
    --values-to-remove 444455556666
```

输出：

```
{
    "DBSnapshotAttributesResult": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "111122223333"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的共享快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbSnapshotAttribute。

modify-db-snapshot-attributes

以下代码示例演示了如何使用 modify-db-snapshot-attributes。

Amazon CLI

修改数据库快照属性

以下 modify-db-snapshot-attribute 示例允许两个 Amazon 账户标识符（111122223333 和 444455556666）还原名为 mydbsnapshot 的数据库快照。

```
aws rds modify-db-snapshot-attribute \
    --db-snapshot-identifier mydbsnapshot \
    --attribute-name restore \
    --values-to-add '["111122223333","444455556666"]'
```

输出：

```
{
    "DBSnapshotAttributesResult": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBSnapshotAttributes": [
            {
                "AttributeName": "restore",
                "AttributeValues": [
                    "111122223333",
                    "444455556666"
                ]
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的共享快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbSnapshotAttributes。

modify-db-snapshot

以下代码示例演示了如何使用 modify-db-snapshot。

Amazon CLI

修改数据库快照

以下 modify-db-snapshot 示例将名为 db5-snapshot-upg-test 的 PostgeSQL 10.6 快照升级为 PostgreSQL 11.7。新的数据库引擎版本将在快照完成升级且其状态变为可用后显示。

```
aws rds modify-db-snapshot \
    --db-snapshot-identifier db5-snapshot-upg-test \
```

```
    --engine-version 11.7
```

输出：

```
{
    "DBSnapshot": {
        "DBSnapshotIdentifier": "db5-snapshot-upg-test",
        "DBInstanceIdentifier": "database-5",
        "SnapshotCreateTime": "2020-03-27T20:49:17.092Z",
        "Engine": "postgres",
        "AllocatedStorage": 20,
        "Status": "upgrading",
        "Port": 5432,
        "AvailabilityZone": "us-west-2a",
        "VpcId": "vpc-2ff27557",
        "InstanceCreateTime": "2020-03-27T19:59:04.735Z",
        "MasterUsername": "postgres",
        "EngineVersion": "10.6",
        "LicenseModel": "postgresql-license",
        "SnapshotType": "manual",
        "OptionGroupName": "default:postgres-11",
        "PercentProgress": 100,
        "StorageType": "gp2",
        "Encrypted": false,
        "DBSnapshotArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-
upg-test",
        "IAMDatabaseAuthenticationEnabled": false,
        "ProcessorFeatures": [],
        "DbiResourceId": "db-GJMF75LM42IL6BTFRE4UZJ5YM4"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的升级 PostgreSQL 数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbSnapshot。

modify-db-subnet-group

以下代码示例演示了如何使用 modify-db-subnet-group。

Amazon CLI

修改数据库子网组

以下 modify-db-subnet-group 示例将 ID 为 subnet-08e41f9e230222222 的子网添加到名为 mysubnetgroup 的数据库子网组。要保留现有子网组中的子网，请在 --subnet-ids 选项中包含其 ID 作为值。确保数据库子网组中的子网至少位于两个不同的可用区中。

```
aws rds modify-db-subnet-group \
    --db-subnet-group-name mysubnetgroup \
    --subnet-ids
 '["subnet-0a1dc4e1a6f123456","subnet-070dd7ecb3aaaaaaa","subnet-00f5b198bc0abcdef","subnet-
```

输出：

```
{
    "DBSubnetGroup": {
        "DBSubnetGroupName": "mysubnetgroup",
        "DBSubnetGroupDescription": "test DB subnet group",
        "VpcId": "vpc-0f08e7610a1b2c3d4",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-08e41f9e230222222",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-070dd7ecb3aaaaaaa",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-00f5b198bc0abcdef",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2d"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-0a1dc4e1a6f123456",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
```

```
                },
                "SubnetStatus": "Active"
            }
        ],
        "DBSubnetGroupArn": "arn:aws:rds:us-
west-2:534026745191:subgrp:mysubnetgroup"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的步骤 3：创建数据库子网组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDbSubnetGroup。

modify-event-subscription

以下代码示例演示了如何使用 modify-event-subscription。

Amazon CLI

修改事件订阅

以下 modify-event-subscription 示例禁用了指定的事件订阅，使其不再向指定的 Amazon Simple Notification Service 主题发布通知。

```
aws rds modify-event-subscription \
    --subscription-name my-instance-events \
    --no-enabled
```

输出：

```
{
    "EventSubscription": {
        "EventCategoriesList": [
            "backup",
            "recovery"
        ],
        "CustomerAwsId": "123456789012",
        "SourceType": "db-instance",
        "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",
        "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
        "CustSubscriptionId": "my-instance-events",
```

```
        "Status": "modifying",
        "Enabled": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyEventSubscription。

modify-global-cluster

以下代码示例演示了如何使用 modify-global-cluster。

Amazon CLI

修改全局数据库集群

以下 modify-global-cluster 示例启用了与 Aurora MySQL 兼容的全局数据库集群的删除保护。

```
aws rds modify-global-cluster \
    --global-cluster-identifier myglobalcluster \
    --deletion-protection
```

输出：

```
{
    "GlobalCluster": {
        "GlobalClusterIdentifier": "myglobalcluster",
        "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",
        "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
        "Status": "available",
        "Engine": "aurora-mysql",
        "EngineVersion": "5.7.mysql_aurora.2.07.2",
        "StorageEncrypted": false,
        "DeletionProtection": true,
        "GlobalClusterMembers": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的管理 Aurora 全局数据库。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyGlobalCluster。

promote-read-replica-db-cluster

以下代码示例演示了如何使用 promote-read-replica-db-cluster。

Amazon CLI

提升数据库集群只读副本

以下 promote-read-replica-db-cluster 示例将指定的只读副本提升为独立的数据库集群。

```
aws rds promote-read-replica-db-cluster \
    --db-cluster-identifier mydbcluster-1
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1b",
            "us-east-1c"
        ],
        "BackupRetentionPeriod": 1,
        "DatabaseName": "",
        "DBClusterIdentifier": "mydbcluster-1",
        ...some output truncated...
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的将只读副本提升为数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PromoteReadReplicaDbCluster。

promote-read-replica

以下代码示例演示了如何使用 promote-read-replica。

Amazon CLI

提升只读副本

以下 promote-read-replica 示例将指定的只读副本提升为独立的数据库实例。

```
aws rds promote-read-replica \
    --db-instance-identifier test-instance-repl
```

输出：

```
{
    "DBInstance": {
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance-repl",
        "StorageType": "standard",
        "ReadReplicaSourceDBInstanceIdentifier": "test-instance",
        "DBInstanceStatus": "modifying",
        ...some output truncated...
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PromoteReadReplica。

purchase-reserved-db-instance

以下代码示例演示了如何使用 purchase-reserved-db-instance。

Amazon CLI

购买预留数据库实例服务

以下 purchase-reserved-db-instances-offering 示例购买了预留数据库实例服务。reserved-db-instances-offering-id 必须是 describe-reserved-db-instances-offering 命令返回的有效产品编号。

aws rds purchase-reserved-db-instances-offering --reserved-db-instances-offering-id 438012d3-4a52-4cc7-b2e3-8dff72e0e706

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseReservedDbInstance。

purchase-reserved-db-instances-offerings

以下代码示例演示了如何使用 purchase-reserved-db-instances-offerings。

Amazon CLI

示例 1：查找要购买的预留数据库实例

以下 describe-reserved-db-instances-offerings 示例列出了实例类为 db.t2.micro 且持续时间为一年的可用预留 MySQL 数据库实例。购买预留数据库实例需要提供产品 ID。

```
aws rds describe-reserved-db-instances-offerings \
    --product-description mysql \
    --db-instance-class db.t2.micro \
    --duration 1
```

输出：

```
{
    "ReservedDBInstancesOfferings": [
        {
            "ReservedDBInstancesOfferingId": "8ba30be1-b9ec-447f-8f23-6114e3f4c7b4",
            "DBInstanceClass": "db.t2.micro",
            "Duration": 31536000,
            "FixedPrice": 51.0,
            "UsagePrice": 0.0,
            "CurrencyCode": "USD",
            "ProductDescription": "mysql",
            "OfferingType": "Partial Upfront",
            "MultiAZ": false,
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.006,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
        },
    ... some output truncated ...
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的 Amazon RDS 的预留数据库实例。

示例 2：购买预留数据库实例

以下 purchase-reserved-db-instances-offering 示例演示如何购买上一个示例中的预留数据库实例服务。

aws rds purchase-reserved-db-instances-offering --reserved-db-instances-offering-id 8ba30be1-b9ec-447f-8f23-6114e3f4c7b4

输出：

```
{
    "ReservedDBInstance": {
        "ReservedDBInstanceId": "ri-2020-06-29-16-54-57-670",
        "ReservedDBInstancesOfferingId": "8ba30be1-b9ec-447f-8f23-6114e3f4c7b4",
        "DBInstanceClass": "db.t2.micro",
        "StartTime": "2020-06-29T16:54:57.670Z",
        "Duration": 31536000,
        "FixedPrice": 51.0,
        "UsagePrice": 0.0,
        "CurrencyCode": "USD",
        "DBInstanceCount": 1,
        "ProductDescription": "mysql",
        "OfferingType": "Partial Upfront",
        "MultiAZ": false,
        "State": "payment-pending",
        "RecurringCharges": [
            {
                "RecurringChargeAmount": 0.006,
                "RecurringChargeFrequency": "Hourly"
            }
        ],
        "ReservedDBInstanceArn": "arn:aws:rds:us-
west-2:123456789012:ri:ri-2020-06-29-16-54-57-670"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的 Amazon RDS 的预留数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseReservedDbInstancesOfferings。

reboot-db-instance

以下代码示例演示了如何使用 reboot-db-instance。

Amazon CLI

重启数据库实例

以下 reboot-db-instance 示例开始重启指定的数据库实例。

```
aws rds reboot-db-instance \
    --db-instance-identifier test-mysql-instance
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-mysql-instance",
        "DBInstanceClass": "db.t3.micro",
        "Engine": "mysql",
        "DBInstanceStatus": "rebooting",
        "MasterUsername": "admin",
        "Endpoint": {
            "Address": "test-mysql-instance.############.us-
west-2.rds.amazonaws.com",
            "Port": 3306,
            "HostedZoneId": "Z1PVIF0EXAMPLE"
        },

    ... output omitted...

    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的重启数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootDBInstance。

reboot-db-shard-group

以下代码示例演示了如何使用 reboot-db-shard-group。

Amazon CLI

示例 1：重启数据库分片组

以下 reboot-db-shard-group 示例重新启动数据库分片组。

```
aws rds reboot-db-shard-group \
    --db-shard-group-identifier my-db-shard-group
```

输出：

```
{
    "DBShardGroups": [
        {
            "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
            "DBShardGroupIdentifier": "my-db-shard-group",
            "DBClusterIdentifier": "my-sv2-cluster",
            "MaxACU": 1000.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": false,
            "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的重启 Amazon Aurora 数据库实例。

示例 2：描述您的数据库分片组

以下 describe-db-shard-groups 示例将在您运行 reboot-db-shard-group 命令后检索数据库分片组的详细信息。数据库分片组 my-db-shard-group 现在正在重启。

```
aws rds describe-db-shard-groups
```

输出：

```
{
    "DBShardGroups": [
        {
            "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
            "DBShardGroupIdentifier": "limitless-test-shard-grp",
            "DBClusterIdentifier": "limitless-test-cluster",
            "MaxACU": 768.0,
            "ComputeRedundancy": 0,
            "Status": "available",
            "PubliclyAccessible": true,
            "Endpoint": "limitless-test-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        },
        {
            "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
```

```
            "DBShardGroupIdentifier": "my-db-shard-group",
            "DBClusterIdentifier": "my-sv2-cluster",
            "MaxACU": 1000.0,
            "ComputeRedundancy": 0,
            "Status": "rebooting",
            "PubliclyAccessible": false,
            "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-
east-2.rds.amazonaws.com"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的重启 Amazon Aurora 数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootDbShardGroup。

register-db-proxy-targets

以下代码示例演示了如何使用 register-db-proxy-targets。

Amazon CLI

向数据库注册数据库代理

以下 register-db-proxy-targets 示例创建了数据库与代理之间的关联。

```
aws rds register-db-proxy-targets \
    --db-proxy-name proxyExample \
    --db-cluster-identifiers database-5
```

输出：

```
{
    "DBProxyTargets": [
        {
            "RdsResourceId": "database-5",
            "Port": 3306,
            "Type": "TRACKED_CLUSTER",
            "TargetHealth": {
                "State": "REGISTERING"
            }
        },
```

```
        {
            "Endpoint": "database-5instance-1.ab0cd1efghij.us-
east-1.rds.amazonaws.com",
            "RdsResourceId": "database-5",
            "Port": 3306,
            "Type": "RDS_INSTANCE",
            "TargetHealth": {
                "State": "REGISTERING"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建 RDS 代理和《Amazon Aurora 用户指南》中的创建 RDS 代理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterDbProxyTargets。

`remove-from-global-cluster`

以下代码示例演示了如何使用 `remove-from-global-cluster`。

Amazon CLI

将 Aurora 辅助集群从 Aurora 全局数据库集群中分离

以下 `remove-from-global-cluster` 示例将 Aurora 辅助集群从 Aurora 全局数据库集群中分离。集群从只读变为具有读写能力的独立集群。

```
aws rds remove-from-global-cluster \
    --region us-west-2 \
    --global-cluster-identifier myglobalcluster \
    --db-cluster-identifier arn:aws:rds:us-west-2:123456789012:cluster:DB-1
```

输出：

```
{
    "GlobalCluster": {
        "GlobalClusterIdentifier": "myglobalcluster",
        "GlobalClusterResourceId": "cluster-abc123def456gh",
        "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
```

```
        "Status": "available",
        "Engine": "aurora-postgresql",
        "EngineVersion": "10.11",
        "StorageEncrypted": true,
        "DeletionProtection": false,
        "GlobalClusterMembers": [
            {
                "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:js-
global-cluster",
                "Readers": [
                    "arn:aws:rds:us-west-2:123456789012:cluster:DB-1"
                ],
                "IsWriter": true
            },
            {
                "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:DB-1",
                "Readers": [],
                "IsWriter": false,
                "GlobalWriteForwardingStatus": "disabled"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的从 Amazon Aurora Global Database 删除集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveFromGlobalCluster。

remove-option-from-option-group

以下代码示例演示了如何使用 remove-option-from-option-group。

Amazon CLI

从选项组中删除选项

以下 remove-option-from-option-group 示例从 myoptiongroup 中删除了 OEM 选项。

```
aws rds remove-option-from-option-group \
    --option-group-name myoptiongroup \
    --options OEM \
    --apply-immediately
```

输出：

```
{
    "OptionGroup": {
        "OptionGroupName": "myoptiongroup",
        "OptionGroupDescription": "Test",
        "EngineName": "oracle-ee",
        "MajorEngineVersion": "19",
        "Options": [],
        "AllowsVpcAndNonVpcInstanceMemberships": true,
        "OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:myoptiongroup"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的从选项组中删除选项。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveOptionFromOptionGroup。

`remove-role-from-db-cluster`

以下代码示例演示了如何使用 `remove-role-from-db-cluster`。

Amazon CLI

取消 Amazon Identity and Access Management（IAM）角色与数据库集群的关联

以下 `remove-role-from-db-cluster` 示例从数据库集群中删除角色。

```
aws rds remove-role-from-db-cluster \
    --db-cluster-identifier mydbcluster \
    --role-arn arn:aws:iam::123456789012:role/RDSLoadFromS3
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Aurora 用户指南》中的将 IAM 角色与 Amazon Aurora MySQL 数据库集群相关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveRoleFromDbCluster。

`remove-role-from-db-instance`

以下代码示例演示了如何使用 `remove-role-from-db-instance`。

Amazon CLI

取消 Amazon Identity and Access Management（IAM）角色与数据库实例的关联

以下 `remove-role-from-db-instance` 示例从名为 `test-instance` 的 Oracle 数据库实例中删除了名为 `rds-s3-integration-role` 的角色。

```
aws rds remove-role-from-db-instance \
    --db-instance-identifier test-instance \
    --feature-name S3_INTEGRATION \
    --role-arn arn:aws:iam::111122223333:role/rds-s3-integration-role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的禁用 RDS SQL Server 与 S3 的集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveRoleFromDbInstance。

`remove-source-identifier-from-subscription`

以下代码示例演示了如何使用 `remove-source-identifier-from-subscription`。

Amazon CLI

从订阅中删除源标识符

以下 `remove-source-identifier` 示例将从现有订阅中删除指定的源标识符。

```
aws rds remove-source-identifier-from-subscription \
    --subscription-name my-instance-events \
    --source-identifier test-instance-repl
```

输出：

```
{
    "EventSubscription": {
        "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
        "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",
        "EventCategoriesList": [
            "backup",
            "recovery"
        ],
```

```
        "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
        "Status": "modifying",
        "CustSubscriptionId": "my-instance-events",
        "CustomerAwsId": "123456789012",
        "SourceIdsList": [
            "test-instance"
        ],
        "SourceType": "db-instance",
        "Enabled": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveSourceIdentifierFromSubscription。

`remove-tags-from-resource`

以下代码示例演示了如何使用 `remove-tags-from-resource`。

Amazon CLI

从资源中删除标签

以下 `remove-tags-from-resource` 示例将从源中删除标签。

```
aws rds remove-tags-from-resource \
    --resource-name arn:aws:rds:us-east-1:123456789012:db:mydbinstance \
    --tag-keys Name Environment
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的为 Amazon RDS 资源添加标签和《Amazon Aurora 用户指南》中的为 Amazon RDS 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromResource。

`reset-db-cluster-parameter-group`

以下代码示例演示了如何使用 `reset-db-cluster-parameter-group`。

Amazon CLI

示例 1：将所有参数重置为其默认值

以下 reset-db-cluster-parameter-group 示例将客户创建的数据库集群参数组中的所有参数值重置为默认值。

```
aws rds reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclpg \
    --reset-all-parameters
```

输出：

```
{
    "DBClusterParameterGroupName": "mydbclpg"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

示例 2：将指定的参数重置为其默认值

以下 reset-db-cluster-parameter-group 示例将客户创建的数据库集群参数组中的特定参数的参数值重置为默认值。

```
aws rds reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclpgy \
    --parameters "ParameterName=max_connections,ApplyMethod=immediate" \
                 "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

输出：

```
{
    "DBClusterParameterGroupName": "mydbclpg"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetDbClusterParameterGroup。

reset-db-parameter-group

以下代码示例演示了如何使用 reset-db-parameter-group。

Amazon CLI

示例 1：将所有参数重置为其默认值

以下 reset-db-parameter-group 示例将客户创建的数据库参数组中的所有参数值重置为默认值。

```
aws rds reset-db-parameter-group \
    --db-parameter-group-name mypg \
    --reset-all-parameters
```

输出：

```
{
    "DBParameterGroupName": "mypg"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组和《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

示例 2：将指定的参数重置为其默认值

以下 reset-db-parameter-group 示例将客户创建的数据库参数组中的特定参数的参数值重置为默认值。

```
aws rds reset-db-parameter-group \
    --db-parameter-group-name mypg \
    --parameters "ParameterName=max_connections,ApplyMethod=immediate" \
                 "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

输出：

```
{
    "DBParameterGroupName": "mypg"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组和《Amazon Aurora 用户指南》中的使用数据库参数组和数据库集群参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetDbParameterGroup。

restore-db-cluster-from-s3

以下代码示例演示了如何使用 restore-db-cluster-from-s3。

Amazon CLI

从 Amazon S3 还原 Amazon Aurora 数据库集群

以下 `restore-db-cluster-from-s3` 示例从 Amazon S3 中的 MySQL 5.7 数据库备份文件中还原了与 Amazon Aurora MySQL 版本 5.7 兼容的数据库集群。

```
aws rds restore-db-cluster-from-s3 \
    --db-cluster-identifier cluster-s3-restore \
    --engine aurora-mysql \
    --master-username admin \
    --master-user-password mypassword \
    --s3-bucket-name amzn-s3-demo-bucket \
    --s3-prefix test-backup \
    --s3-ingestion-role-arn arn:aws:iam::123456789012:role/service-role/TestBackup \
    --source-engine mysql \
    --source-engine-version 5.7.28
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-west-2c",
            "us-west-2a",
            "us-west-2b"
        ],
        "BackupRetentionPeriod": 1,
        "DBClusterIdentifier": "cluster-s3-restore",
        "DBClusterParameterGroup": "default.aurora-mysql5.7",
        "DBSubnetGroup": "default",
        "Status": "creating",
        "Endpoint": "cluster-s3-restore.cluster-co3xyzabc123.us-
west-2.rds.amazonaws.com",
        "ReaderEndpoint": "cluster-s3-restore.cluster-ro-co3xyzabc123.us-
west-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-mysql",
        "EngineVersion": "5.7.12",
        "Port": 3306,
        "MasterUsername": "admin",
        "PreferredBackupWindow": "11:15-11:45",
```

```
        "PreferredMaintenanceWindow": "thu:12:19-thu:12:49",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-########",
                "Status": "active"
            }
        ],
        "HostedZoneId": "Z1PVIF0EXAMPLE",
        "StorageEncrypted": false,
        "DbClusterResourceId": "cluster-SU5THYQQHOWCXZZDGXREXAMPLE",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:cluster-s3-
restore",
        "AssociatedRoles": [],
        "IAMDatabaseAuthenticationEnabled": false,
        "ClusterCreateTime": "2020-07-27T14:22:08.095Z",
        "EngineMode": "provisioned",
        "DeletionProtection": false,
        "HttpEndpointEnabled": false,
        "CopyTagsToSnapshot": false,
        "CrossAccountClone": false,
        "DomainMemberships": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的通过使用 Amazon S3 存储桶从 MySQL 中迁移数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbClusterFromS3。

`restore-db-cluster-from-snapshot`

以下代码示例演示了如何使用 `restore-db-cluster-from-snapshot`。

Amazon CLI

从快照还原数据库集群

以下 `restore-db-cluster-from-snapshot` 示例从名为 `test-instance-snapshot` 的数据库集群快照还原与 PostgreSQL 版本 10.7 兼容的 Aurora PostgreSQL 数据库集群。

```
aws rds restore-db-cluster-from-snapshot \
```

```
    --db-cluster-identifier newdbcluster \
    --snapshot-identifier test-instance-snapshot \
    --engine aurora-postgresql \
    --engine-version 10.7
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-west-2c",
            "us-west-2a",
            "us-west-2b"
        ],
        "BackupRetentionPeriod": 7,
        "DatabaseName": "",
        "DBClusterIdentifier": "newdbcluster",
        "DBClusterParameterGroup": "default.aurora-postgresql10",
        "DBSubnetGroup": "default",
        "Status": "creating",
        "Endpoint": "newdbcluster.cluster-############.us-west-2.rds.amazonaws.com",
        "ReaderEndpoint": "newdbcluster.cluster-ro-############.us-
west-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-postgresql",
        "EngineVersion": "10.7",
        "Port": 5432,
        "MasterUsername": "postgres",
        "PreferredBackupWindow": "09:33-10:03",
        "PreferredMaintenanceWindow": "sun:12:22-sun:12:52",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-########",
                "Status": "active"
            }
        ],
        "HostedZoneId": "Z1PVIF0EXAMPLE",
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/287364e4-33e3-4755-a3b0-
a1b2c3d4e5f6",
```

```
        "DbClusterResourceId": "cluster-5DSB5IFQDDUVAWOUWM1EXAMPLE",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:newdbcluster",
        "AssociatedRoles": [],
        "IAMDatabaseAuthenticationEnabled": false,
        "ClusterCreateTime": "2020-06-05T15:06:58.634Z",
        "EngineMode": "provisioned",
        "DeletionProtection": false,
        "HttpEndpointEnabled": false,
        "CopyTagsToSnapshot": false,
        "CrossAccountClone": false,
        "DomainMemberships": []
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的从数据库集群快照还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbClusterFromSnapshot。

`restore-db-cluster-to-point-in-time`

以下代码示例演示了如何使用 `restore-db-cluster-to-point-in-time`。

Amazon CLI

将数据库集群还原到指定时间

以下 `restore-db-cluster-to-point-in-time` 示例将名为 `database-4` 的数据库集群还原到尽可能晚的时间。使用 `copy-on-write` 作为还原类型，将新数据库集群还原为源数据库集群的克隆。

```
aws rds restore-db-cluster-to-point-in-time \
    --source-db-cluster-identifier database-4 \
    --db-cluster-identifier sample-cluster-clone \
    --restore-type copy-on-write \
    --use-latest-restorable-time
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-west-2c",
```

```
            "us-west-2a",
            "us-west-2b"
        ],
        "BackupRetentionPeriod": 7,
        "DatabaseName": "",
        "DBClusterIdentifier": "sample-cluster-clone",
        "DBClusterParameterGroup": "default.aurora-postgresql10",
        "DBSubnetGroup": "default",
        "Status": "creating",
        "Endpoint": "sample-cluster-clone.cluster-############.us-
west-2.rds.amazonaws.com",
        "ReaderEndpoint": "sample-cluster-clone.cluster-ro-############.us-
west-2.rds.amazonaws.com",
        "MultiAZ": false,
        "Engine": "aurora-postgresql",
        "EngineVersion": "10.7",
        "Port": 5432,
        "MasterUsername": "postgres",
        "PreferredBackupWindow": "09:33-10:03",
        "PreferredMaintenanceWindow": "sun:12:22-sun:12:52",
        "ReadReplicaIdentifiers": [],
        "DBClusterMembers": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-########",
                "Status": "active"
            }
        ],
        "HostedZoneId": "Z1PVIF0EXAMPLE",
        "StorageEncrypted": true,
        "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/287364e4-33e3-4755-a3b0-
a1b2c3d4e5f6",
        "DbClusterResourceId": "cluster-BIZ77GDSA2XBSTNPFW1EXAMPLE",
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster-
clone",
        "AssociatedRoles": [],
        "IAMDatabaseAuthenticationEnabled": false,
        "CloneGroupId": "8d19331a-099a-45a4-b4aa-11aa22bb33cc44dd",
        "ClusterCreateTime": "2020-03-10T19:57:38.967Z",
        "EngineMode": "provisioned",
        "DeletionProtection": false,
        "HttpEndpointEnabled": false,
        "CopyTagsToSnapshot": false,
        "CrossAccountClone": false
```

```
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的将数据库集群还原到指定时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbClusterToPointInTime。

`restore-db-instance-from-db-snapshot`

以下代码示例演示了如何使用 `restore-db-instance-from-db-snapshot`。

Amazon CLI

从数据库快照还原数据库实例

以下 `restore-db-instance-from-db-snapshot` 示例从指定数据库快照创建数据库实例类为 `db.t3.small` 且名为 `db7-new-instance` 的新数据库实例。从中拍摄快照的源数据库实例使用已弃用的数据库实例类，因此您无法对其进行升级。

```
aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier db7-new-instance \
    --db-snapshot-identifier db7-test-snapshot \
    --db-instance-class db.t3.small
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "db7-new-instance",
        "DBInstanceClass": "db.t3.small",
        "Engine": "mysql",
        "DBInstanceStatus": "creating",

        ...output omitted...

        "PreferredMaintenanceWindow": "mon:07:37-mon:08:07",
        "PendingModifiedValues": {},
        "MultiAZ": false,
        "EngineVersion": "5.7.22",
        "AutoMinorVersionUpgrade": true,
        "ReadReplicaDBInstanceIdentifiers": [],
```

```
        "LicenseModel": "general-public-license",

        ...output omitted...

        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:db7-new-instance",
        "IAMDatabaseAuthenticationEnabled": false,
        "PerformanceInsightsEnabled": false,
        "DeletionProtection": false,
        "AssociatedRoles": []
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的从数据库快照还原。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbInstanceFromDbSnapshot。

`restore-db-instance-from-s3`

以下代码示例演示了如何使用 `restore-db-instance-from-s3`。

Amazon CLI

从 Amazon S3 中的备份还原数据库实例

以下 `restore-db-instance-from-s3` 示例根据 `my-backups` S3 存储桶中的现有备份新建一个名为 `restored-test-instance` 的数据库实例。

```
aws rds restore-db-instance-from-s3 \
    --db-instance-identifier restored-test-instance \
    --allocated-storage 250 --db-instance-class db.m4.large --engine mysql \
    --master-username master --master-user-password secret99 \
    --s3-bucket-name my-backups --s3-ingestion-role-
arn arn:aws:iam::123456789012:role/my-role \
    --source-engine mysql --source-engine-version 5.6.27
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbInstanceFromS3。

`restore-db-instance-to-point-in-time`

以下代码示例演示了如何使用 `restore-db-instance-to-point-in-time`。

Amazon CLI

示例 1：将数据库实例还原到某个时间点

截至指定的时间，以下 restore-db-instance-to-point-in-time 示例会将 test-instance 还原到名为 restored-test-instance 的新数据库实例。

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-identifier test-instance \
    --target-db-instance restored-test-instance \
    --restore-time 2018-07-30T23:45:00.000Z
```

输出：

```
{
    "DBInstance": {
        "AllocatedStorage": 20,
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:restored-test-
instance",
        "DBInstanceStatus": "creating",
        "DBInstanceIdentifier": "restored-test-instance",
        ...some output omitted...
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的将数据库实例还原到指定时间。

示例 2：将数据库实例从复制备份还原到指定时间

以下 restore-db-instance-to-point-in-time 示例从复制的自动备份将 Oracle 数据库实例恢复到指定时间。

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-automated-backups-arn "arn:aws:rds:us-
west-2:123456789012:auto-backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example" \
    --target-db-instance-identifier myorclinstance-from-replicated-backup \
    --restore-time 2020-12-08T18:45:00.000Z
```

输出：

```
{
```

```
    "DBInstance": {
        "DBInstanceIdentifier": "myorclinstance-from-replicated-backup",
        "DBInstanceClass": "db.t3.micro",
        "Engine": "oracle-se2",
        "DBInstanceStatus": "creating",
        "MasterUsername": "admin",
        "DBName": "ORCL",
        "AllocatedStorage": 20,
        "PreferredBackupWindow": "07:45-08:15",
        "BackupRetentionPeriod": 14,
        ... some output omitted ...
        "DbiResourceId": "db-KGLXG75BGVIWKQT7NQ4EXAMPLE",
        "CACertificateIdentifier": "rds-ca-2019",
        "DomainMemberships": [],
        "CopyTagsToSnapshot": false,
        "MonitoringInterval": 0,
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:myorclinstance-from-
replicated-backup",
        "IAMDatabaseAuthenticationEnabled": false,
        "PerformanceInsightsEnabled": false,
        "DeletionProtection": false,
        "AssociatedRoles": [],
        "TagList": []
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的从复制备份还原到指定时间。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreDbInstanceToPointInTime。

`start-activity-stream`

以下代码示例演示了如何使用 `start-activity-stream`。

Amazon CLI

启动数据库活动流

以下 `start-activity-stream` 示例启动异步活动流来监控名为 my-pg-cluster 的 Aurora 集群。

```
aws rds start-activity-stream \
    --region us-east-1 \
    --mode async \
```

```
    --kms-key-id arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-
h123-456i789jk0l1 \
    --resource-arn arn:aws:rds:us-east-1:1234567890123:cluster:my-pg-cluster \
    --apply-immediately
```

输出：

```
{
    "KmsKeyId": "arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-
h123-456i789jk0l1",
    "KinesisStreamName": "aws-rds-das-cluster-0ABCDEFGHI1JKLM2NOPQ3R4S",
    "Status": "starting",
    "Mode": "async",
    "ApplyImmediately": true
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的启动数据库活动流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartActivityStream。

start-db-cluster

以下代码示例演示了如何使用 start-db-cluster。

Amazon CLI

启动数据库集群

以下 start-db-cluster 示例启动数据库集群及其数据库实例。

```
aws rds start-db-cluster \
    --db-cluster-identifier mydbcluster
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1e",
            "us-east-1b"
```

```
        ],
        "BackupRetentionPeriod": 1,
        "DatabaseName": "mydb",
        "DBClusterIdentifier": "mydbcluster",
        ...some output truncated...
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的停止和启动 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDbCluster。

start-db-instance-automated-backups-replication

以下代码示例演示了如何使用 start-db-instance-automated-backups-replication。

Amazon CLI

启用跨区域自动备份

以下 start-db-instance-automated-backups-replication 示例将数据库实例的自动备份从美国东部（弗吉尼亚州北部）地区复制到美国西部（俄勒冈州）地区。备份保留期为 14 天。

```
aws rds start-db-instance-automated-backups-replication \
    --region us-west-2 \
    --source-db-instance-arn "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db" \
    --backup-retention-period 14
```

输出：

```
{
    "DBInstanceAutomatedBackup": {
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
        "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
        "Region": "us-east-1",
        "DBInstanceIdentifier": "new-orcl-db",
        "RestoreWindow": {},
        "AllocatedStorage": 20,
        "Status": "pending",
        "Port": 1521,
        "InstanceCreateTime": "2020-12-04T15:28:31Z",
```

```
        "MasterUsername": "admin",
        "Engine": "oracle-se2",
        "EngineVersion": "12.1.0.2.v21",
        "LicenseModel": "bring-your-own-license",
        "OptionGroupName": "default:oracle-se2-12-1",
        "Encrypted": false,
        "StorageType": "gp2",
        "IAMDatabaseAuthenticationEnabled": false,
        "BackupRetentionPeriod": 14,
        "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-
backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的启用跨区域自动备份。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDbInstanceAutomatedBackupsReplication。

start-db-instance

以下代码示例演示了如何使用 start-db-instance。

Amazon CLI

启动数据库实例

以下 start-db-instance 示例启动了指定的数据库实例。

```
aws rds start-db-instance \
    --db-instance-identifier test-instance
```

输出：

```
{
    "DBInstance": {
        "DBInstanceStatus": "starting",
        ...some output truncated...
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDbInstance。

start-export-task

以下代码示例演示了如何使用 start-export-task。

Amazon CLI

将快照导出到 Amazon S3

以下 start-export-task 示例将名为 db5-snapshot-test 的数据库快照导出到名为 amzn-s3-demo-bucket 的 Amazon S3 存储桶。

```
aws rds start-export-task \
    --export-task-identifier my-s3-export \
    --source-arn arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-test \
    --s3-bucket-name amzn-s3-demo-bucket \
    --iam-role-arn arn:aws:iam::123456789012:role/service-role/ExportRole \
    --kms-key-id arn:aws:kms:us-west-2:123456789012:key/abcd0000-7fca-4128-82f2-aabbccddeeff
```

输出：

```
{
    "ExportTaskIdentifier": "my-s3-export",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-test",
    "SnapshotTime": "2020-03-27T20:48:42.023Z",
    "S3Bucket": "amzn-s3-demo-bucket",
    "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/abcd0000-7fca-4128-82f2-aabbccddeeff",
    "Status": "STARTING",
    "PercentProgress": 0,
    "TotalExtractedDataInGB": 0
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的将快照导出到 Amazon S3 存储桶。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartExportTask。

stop-activity-stream

以下代码示例演示了如何使用 stop-activity-stream。

Amazon CLI

停止数据库活动流

以下 stop-activity-stream 示例停止了名为 my-pg-cluster 的 Aurora 集群中的活动流。

```
aws rds stop-activity-stream \
    --region us-east-1 \
    --resource-arn arn:aws:rds:us-east-1:1234567890123:cluster:my-pg-cluster \
    --apply-immediately
```

输出：

```
{
    "KmsKeyId": "arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-
h123-456i789jk0l1",
    "KinesisStreamName": "aws-rds-das-cluster-0ABCDEFGHI1JKLM2NOPQ3R4S",
    "Status": "stopping"
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的停止活动流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopActivityStream。

stop-db-cluster

以下代码示例演示了如何使用 stop-db-cluster。

Amazon CLI

停止数据库集群

以下 stop-db-cluster 示例停止了数据库集群及其数据库实例。

```
aws rds stop-db-cluster \
    --db-cluster-identifier mydbcluster
```

输出：

```
{
    "DBCluster": {
        "AllocatedStorage": 1,
```

```
        "AvailabilityZones": [
            "us-east-1a",
            "us-east-1e",
            "us-east-1b"
        ],
        "BackupRetentionPeriod": 1,
        "DatabaseName": "mydb",
        "DBClusterIdentifier": "mydbcluster",
        ...some output truncated...
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的停止和启动 Amazon Aurora 数据库集群。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDbCluster。

stop-db-instance-automated-backups-replication

以下代码示例演示了如何使用 stop-db-instance-automated-backups-replication。

Amazon CLI

停止复制自动备份

以下 stop-db-instance-automated-backups-replication 结束了将自动备份复制到美国西部（俄勒冈州）区域。所复制备份的保留期为设定的备份保留期。

```
aws rds stop-db-instance-automated-backups-replication \
    --region us-west-2 \
    --source-db-instance-arn "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db"
```

输出：

```
{
    "DBInstanceAutomatedBackup": {
        "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
        "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
        "Region": "us-east-1",
        "DBInstanceIdentifier": "new-orcl-db",
        "RestoreWindow": {
            "EarliestTime": "2020-12-04T23:13:21.030Z",
            "LatestTime": "2020-12-07T19:59:57Z"
```

```
        },
        "AllocatedStorage": 20,
        "Status": "replicating",
        "Port": 1521,
        "InstanceCreateTime": "2020-12-04T15:28:31Z",
        "MasterUsername": "admin",
        "Engine": "oracle-se2",
        "EngineVersion": "12.1.0.2.v21",
        "LicenseModel": "bring-your-own-license",
        "OptionGroupName": "default:oracle-se2-12-1",
        "Encrypted": false,
        "StorageType": "gp2",
        "IAMDatabaseAuthenticationEnabled": false,
        "BackupRetentionPeriod": 7,
        "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-
backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的停止自动备份复制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDbInstanceAutomatedBackupsReplication。

stop-db-instance

以下代码示例演示了如何使用 stop-db-instance。

Amazon CLI

停止数据库实例

以下 stop-db-instance 示例停止了指定的数据库实例。

```
aws rds stop-db-instance \
    --db-instance-identifier test-instance
```

输出：

```
{
    "DBInstance": {
        "DBInstanceStatus": "stopping",
        ...some output truncated...
```

```
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopDbInstance。

switchover-blue-green-deployment

以下代码示例演示了如何使用 switchover-blue-green-deployment。

Amazon CLI

示例 1：切换 RDS 数据库实例的蓝绿部署

以下 switchover-blue-green-deployment 示例将指定的绿色环境提升为新的生产环境。

```
aws rds switchover-blue-green-deployment \
    --blue-green-deployment-identifier bgd-wi89nwzglccsfake \
    --switchover-timeout 300
```

输出：

```
{
    "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",
        "BlueGreenDeploymentName": "bgd-cli-test-instance",
        "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
        "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-
blhi1e",
        "SwitchoverDetails": [
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-green-blhi1e",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1-green-k5fv7u",
                "Status": "AVAILABLE"
```

```
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-green-ggsh8m",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-o2vwm0",
                "Status": "AVAILABLE"
            }
        ],
        "Tasks": [
            {
                "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                "Status": "COMPLETED"
            },
            {
                "Name": "DB_ENGINE_VERSION_UPGRADE",
                "Status": "COMPLETED"
            },
            {
                "Name": "CONFIGURE_BACKUPS",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATING_TOPOLOGY_OF_SOURCE",
                "Status": "COMPLETED"
            }
        ],
        "Status": "SWITCHOVER_IN_PROGRESS",
        "CreateTime": "2022-02-25T22:33:22.225000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的切换蓝绿部署。

示例 2：为 Aurora MySQL 数据库集群提升蓝绿部署

以下 `switchover-blue-green-deployment` 示例将指定的绿色环境提升为新的生产环境。

```
aws rds switchover-blue-green-deployment \
    --blue-green-deployment-identifier bgd-wi89nwzglccsfake \
    --switchover-timeout 300
```

输出：

```
{
    "BlueGreenDeployment": {
        "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
        "BlueGreenDeploymentName": "my-blue-green-deployment",
        "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
        "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3ud8z6",
        "SwitchoverDetails": [
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3ud8z6",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1-green-bvxc73",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2-green-7wc4ie",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3-green-p4xxkz",
                "Status": "AVAILABLE"
            },
```

```
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-np1ikl",
                "Status": "AVAILABLE"
            },
            {
                "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
                "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-miszlf",
                "Status": "AVAILABLE"
            }
        ],
        "Tasks": [
            {
                "Name": "CREATING_READ_REPLICA_OF_SOURCE",
                "Status": "COMPLETED"
            },
            {
                "Name": "DB_ENGINE_VERSION_UPGRADE",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
                "Status": "COMPLETED"
            },
            {
                "Name": "CREATE_CUSTOM_ENDPOINTS",
                "Status": "COMPLETED"
            }
        ],
        "Status": "SWITCHOVER_IN_PROGRESS",
        "CreateTime": "2022-02-25T22:38:49.522000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Aurora 用户指南》中的切换蓝绿部署。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SwitchoverBlueGreenDeployment。

使用 Amazon CLI 的 Amazon RDS 数据服务示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon RDS 数据服务结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-execute-statement`

以下代码示例演示了如何使用 `batch-execute-statement`。

Amazon CLI

执行批处理 SQL 语句

以下 `batch-execute-statement` 实例使用参数集对数据数组运行批处理 SQL 语句。

```
aws rds-data batch-execute-statement \
    --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
    --database "mydb" \
    --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \
    --sql "insert into mytable values (:id, :val)" \
    --parameter-sets "[[{\"name\": \"id\", \"value\": {\"longValue\": 1}},{\"name\":
 \"val\", \"value\": {\"stringValue\": \"ValueOne\"}}],
        [{\"name\": \"id\", \"value\": {\"longValue\": 2}},{\"name\": \"val\",
 \"value\": {\"stringValue\": \"ValueTwo\"}}],
        [{\"name\": \"id\", \"value\": {\"longValue\": 3}},{\"name\": \"val\",
 \"value\": {\"stringValue\": \"ValueThree\"}}]]"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 Aurora Serverless 数据 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchExecuteStatement。

begin-transaction

以下代码示例演示了如何使用 begin-transaction。

Amazon CLI

开始 SQL 事务

以下 begin-transaction 示例运行 SQL 事务。

```
aws rds-data begin-transaction \
    --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
    --database "mydb" \
    --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret"
```

输出：

```
{
    "transactionId": "ABC1234567890xyz"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 Aurora Serverless 数据 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BeginTransaction。

commit-transaction

以下代码示例演示了如何使用 commit-transaction。

Amazon CLI

提交 SQL 事务

以下 commit-transaction 示例结束指定的 SQL 事务并提交您在其中所做的更改。

```
aws rds-data commit-transaction \
    --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
    --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \
```

```
    --transaction-id "ABC1234567890xyz"
```

输出：

```
{
    "transactionStatus": "Transaction Committed"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 Aurora Serverless 数据 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CommitTransaction。

execute-statement

以下代码示例演示了如何使用 execute-statement。

Amazon CLI

示例 1：执行属于事务的 SQL 语句

以下 execute-statement 示例运行属于事务的 SQL 语句。

```
aws rds-data execute-statement \
    --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
    --database "mydb" \
    --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \
    --sql "update mytable set quantity=5 where id=201" \
    --transaction-id "ABC1234567890xyz"
```

输出：

```
{
    "numberOfRecordsUpdated": 1
}
```

示例 2：执行带有参数的 SQL 语句

以下 execute-statement 示例运行带有参数的 SQL 语句。

```
aws rds-data execute-statement \
```

```
    --resource-arn "arn:aws:rds:us-east-1:123456789012:cluster:mydbcluster" \
    --database "mydb" \
    --secret-arn "arn:aws:secretsmanager:us-east-1:123456789012:secret:mysecret" \
    --sql "insert into mytable values (:id, :val)" \
    --parameters "[{\"name\": \"id\", \"value\": {\"longValue\": 1}},{\"name\":
 \"val\", \"value\": {\"stringValue\": \"value1\"}}]"
```

输出：

```
{
    "numberOfRecordsUpdated": 1
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 Aurora Serverless 数据 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ExecuteStatement。

rollback-transaction

以下代码示例演示了如何使用 rollback-transaction。

Amazon CLI

回滚 SQL 事务

以下 rollback-transaction 示例回滚指定的 SQL 事务。

```
aws rds-data rollback-transaction \
    --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
    --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \
    --transaction-id "ABC1234567890xyz"
```

输出：

```
{
    "transactionStatus": "Rollback Complete"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用 Aurora Serverless 数据 API。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RollbackTransaction。

使用 Amazon CLI 的 Amazon RDS 性能详情示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon RDS 性能详情结合使用，以执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-performance-analysis-report`

以下代码示例演示了如何使用 `create-performance-analysis-report`。

Amazon CLI

创建性能分析报告

以下 `create-performance-analysis-report` 示例为数据库 db-abcdefg123456789 创建了一个开始时间为 1682969503、结束时间为 1682979503 的性能分析报告。

```
aws pi create-performance-analysis-report \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --start-time 1682969503 \
    --end-time 1682979503
```

输出：

```
{
    "AnalysisReportId": "report-0234d3ed98e28fb17"
}
```

有关创建性能分析报告的更多信息，请参阅《Amazon RDS 用户指南》中的在性能详情中创建性能分析报告和《Amazon Aurora 用户指南》中的在性能详情中创建性能分析报告。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreatePerformanceAnalysisReport。

delete-performance-analysis-report

以下代码示例演示了如何使用 delete-performance-analysis-report。

Amazon CLI

删除性能分析报告

以下 delete-performance-analysis-report 示例删除报告 ID 为 report-0d99cc91c4422ee61 的性能分析报告。

```
aws pi delete-performance-analysis-report \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --analysis-report-id report-0d99cc91c4422ee61
```

此命令不生成任何输出。

有关删除性能分析报告的更多信息，请参阅《Amazon RDS 用户指南》中的在性能详情中删除性能分析报告和《Amazon Aurora 用户指南》中的在性能详情中删除性能分析报告。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeletePerformanceAnalysisReport。

describe-dimension-keys

以下代码示例演示了如何使用 describe-dimension-keys。

Amazon CLI

示例 1：描述维度键

此示例请求所有等待事件的名称。数据按事件名称以及指定时间段内这些事件的汇总值进行汇总。

命令:

```
aws pi describe-dimension-keys --service-type RDS --identifier db-
LKCGOBK26374TPTDFXOIWVCPPM --start-time 1527026400 --end-time 1527080400 --
metric db.load.avg --group-by '{"Group":"db.wait_event"}'
```

输出：


```
{
    "AlignedEndTime": 1.5270804E9,
    "AlignedStartTime": 1.5270264E9,
    "Keys": [
        {
            "Dimensions": {"db.wait_event.name": "wait/synch/mutex/innodb/
aurora_lock_thread_slot_futex"},
            "Total": 0.05906906851195666
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/io/aurora_redo_log_flush"},
            "Total": 0.015824722186149193
        },
        {
            "Dimensions": {"db.wait_event.name": "CPU"},
            "Total": 0.008014396230265477
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/io/
aurora_respond_to_client"},
            "Total": 0.0036361612526204477
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/io/table/sql/handler"},
            "Total": 0.0019108398419382965
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/synch/cond/mysys/
my_thread_var::suspend"},
            "Total": 8.533847837782684E-4
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/io/file/csv/data"},
            "Total": 6.864181956477376E-4
        },
        {
            "Dimensions": {"db.wait_event.name": "Unknown"},
            "Total": 3.895887056379051E-4
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/synch/mutex/sql/
FILE_AS_TABLE::LOCK_shim_lists"},
```

```
            "Total": 3.710368625122906E-5
        },
        {
            "Dimensions": {"db.wait_event.name": "wait/lock/table/sql/handler"},
            "Total": 0
        }
    ]
}
```

示例 2：查找对数据库负载贡献最大的语句的 SQL ID

以下 `describe-dimension-keys` 请求对数据库负载贡献最大的 10 条语句的 SQL 语句和 SQL ID。

```
aws pi describe-dimension-keys \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --start-time 2023-05-01T00:00:00Z \
    --end-time 2023-05-01T01:00:00Z \
    --metric db.load.avg \
    --group-by '{"Group": "db.sql", "Dimensions": ["db.sql.id",
 "db.sql.statement"],"Limit": 10}'
```

输出：

```
{
    "AlignedEndTime": 1.5270804E9,
    "AlignedStartTime": 1.5270264E9,
    "Identifier": "db-abcdefg123456789",
    "MetricList": [
        {
            "Keys": [
                {
                    "Dimensions": {"db.sql.id": "AKIAIOSFODNN7EXAMPLE",
 "db.sql.statement": "SELECT * FROM customers WHERE customer_id = 123"},
                    "Total": 25.5,"Partitions": [12.3, 13.2]
                }
            ]
        }
    ]
}
```

有关性能详情中维度的更多信息，请参阅《Amazon RDS 用户指南》中的数据库负载和《Amazon Aurora 用户指南》中的数据库负载。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDimensionKeys。

get-dimension-key-details

以下代码示例演示了如何使用 get-dimension-key-details。

Amazon CLI

获取数据库实例的指定维度组的详细信息

以下 get-dimension-key-details 示例检索数据库实例 db-10BCD2EFGHIJ3KL4M5NO6PQRS5 的 SQL 语句的全文。--group 是 db.sql，--group-identifier 是 db.sql.id。在此示例中，example-sql-id 表示通过使用 get-resource-metrics 或 describe-dimension-keys 检索到的 SQL ID。在此示例中，有维度详细信息。因此，Performance Insights 将检索 SQL 语句的全文，而不会将其截断。

```
aws pi get-dimension-key-details \
    --service-type RDS \
    --identifier db-10BCD2EFGHIJ3KL4M5NO6PQRS5 \
    --group db.sql \
    --group-identifier example-sql-id \
    --requested-dimensions statement
```

输出：

```
{
    "Dimensions":[
        {
            "Value": "SELECT e.last_name, d.department_name FROM employees e,
 departments d WHERE e.department_id=d.department_id",
            "Dimension": "db.sql.statement",
            "Status": "AVAILABLE"
        },
    ...
    ]
}
```

有关性能详情中维度的更多信息，请参阅《Amazon RDS 用户指南》中的数据库负载和《Amazon Aurora 用户指南》中的数据库负载。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDimensionKeyDetails。

get-performance-analysis-report

以下代码示例演示了如何使用 get-performance-analysis-report。

Amazon CLI

获取性能分析报告

以下 get-performance-analysis-report 示例获取数据库 db-abcdefg123456789 的报告 ID 为 report-0d99cc91c4422ee61 的性能分析报告。响应提供报告状态、ID、时间详细信息和见解。

```
aws pi get-performance-analysis-report \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --analysis-report-id report-0d99cc91c4422ee61
```

输出：

```
{
    "AnalysisReport": {
        "Status": "Succeeded",
        "ServiceType": "RDS",
        "Identifier": "db-abcdefg123456789",
        "StartTime": 1680583486.584,
        "AnalysisReportId": "report-0d99cc91c4422ee61",
        "EndTime": 1680587086.584,
        "CreateTime": 1680587087.139,
        "Insights": [
            ... (Condensed for space)
       ]
    }
}
```

有关性能分析报告的更多信息，请参阅《Amazon RDS 用户指南》中的分析一段时间内的数据库性能和《Amazon Aurora 用户指南》中的分析一段时间内的数据库性能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetPerformanceAnalysisReport。

get-resource-metadata

以下代码示例演示了如何使用 get-resource-metadata。

Amazon CLI

获取数据库的资源元数据

以下 get-resource-metadata 示例获取数据库 db-abcdefg123456789 的资源元数据。响应显示 SQL 摘要统计信息已启用。

```
aws pi get-resource-metadata \
    --service-type RDS \
    --identifier db-abcdefg123456789
```

输出：

```
{
    "Identifier": "db-abcdefg123456789",
    "Features":{
        "SQL_DIGEST_STATISTICS":{
            "Status": "ENABLED"
        }
    }
}
```

有关性能详情的 SQL 统计数据的更多信息，请参阅《Amazon RDS 用户指南》中的性能详情的 SQL 统计数据和《Amazon Aurora 用户指南》中的性能详情的 SQL 统计数据。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetResourceMetadata。

get-resource-metrics

以下代码示例演示了如何使用 get-resource-metrics。

Amazon CLI

获取资源指标

此示例为 db.wait_event 维度组以及该组中的 db.wait_event.name 维度请求数据点。在响应中，相关的数据点按请求的维度（db.wait_event.name）分组。

命令:

```
aws pi get-resource-metrics --service-type RDS --identifier db-LKCGOBK26374TPTDFXOIWVCPPM --start-time 1527026400 --end-time 1527080400 --period-in-seconds 300 --metric db.load.avg --metric-queries file://metric-queries.json
```

--metric-queries 的参数存储在 JSON 文件（metric-queries.json）中。以下是该文件的内容：

```
[
    {
        "Metric": "db.load.avg",
        "GroupBy": {
            "Group":"db.wait_event"
        }
    }
]
```

输出：

```
{
    "AlignedEndTime": 1.5270804E9,
    "AlignedStartTime": 1.5270264E9,
    "Identifier": "db-LKCGOBK26374TPTDFXOIWVCPPM",
    "MetricList": [
        {
            "Key": {
                "Metric": "db.load.avg"
            },
            "DataPoints": [
                {
                    "Timestamp": 1527026700.0,
                    "Value": 1.3533333333333333
                },
                {
                    "Timestamp": 1527027000.0,
                    "Value": 0.88
                },
                <...remaining output omitted...>
            ]
        },
        {
```

```
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.name": "wait/synch/mutex/innodb/
aurora_lock_thread_slot_futex"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1527026700.0,
                    "Value": 0.8566666666666667
                },
                {
                    "Timestamp": 1527027000.0,
                    "Value": 0.8633333333333333
                },
                <...remaining output omitted...>
            ],
        },
            <...remaining output omitted...>
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceMetrics。

list-available-resource-dimensions

以下代码示例演示了如何使用 list-available-resource-dimensions。

Amazon CLI

列出可以在数据库实例上针对指标类型查询的维度

以下 list-available-resource-dimensions 示例列出了您可以为数据库 db-abcdefg123456789 查询的 db.load 指标。

```
aws pi list-available-resource-dimensions \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --metrics db.load
```

输出：

```
{
    "MetricDimensions": [
        {
            "Metric": "db.load",
            "Groups": [
                {
                    "Group": "db.user",
                    "Dimensions": [
                        {
                            "Identifier": "db.user.id"
                        },
                        {
                            "Identifier": "db.user.name"
                        }
                    ]
                },
                {
                    "Group": "db.sql_tokenized",
                    "Dimensions": [
                        {
                            "Identifier": "db.sql_tokenized.id"
                        },
                        {
                            "Identifier": "db.sql_tokenized.db_id"
                        },
                        {
                            "Identifier": "db.sql_tokenized.statement"
                        }
                    ]
                },
                ...
            ]
        }
    ]
}
```

有关性能详情中维度的更多信息，请参阅《Amazon RDS 用户指南》中的数据库负载和《Amazon Aurora 用户指南》中的数据库负载。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAvailableResourceDimensions。

list-available-resource-metrics

以下代码示例演示了如何使用 list-available-resource-metrics。

Amazon CLI

列出可以在数据库实例上针对指标类型查询的指标

以下 list-available-resource-metrics 示例列出了您可以为数据库 db-abcdefg123456789 查询的 db.load 指标。

```
aws pi list-available-resource-metrics \
    --service-type RDS \
    --identifier db-abcdefg123456789 \
    --metric-types "os" "db"
```

输出：

```
{
    "Metrics": [
        {
            "Description": "The number of virtual CPUs for the DB instance",
            "Metric": "os.general.numVCPUs",
            "Unit": "vCPUs"
        },
        ......,
        {
            "Description": "Time spent reading data file blocks by backends in this
 instance",
            "Metric": "db.IO.read_latency",
            "Unit": "Milliseconds per block"
        },
        ......
    ]
}
```

有关性能详情中指标的更多信息，请参阅《Amazon RDS 用户指南》中的数据库负载和《Amazon Aurora 用户指南》中的数据库负载。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAvailableResourceMetrics。

list-performance-analysis-reports

以下代码示例演示了如何使用 list-performance-analysis-reports。

Amazon CLI

列出数据库的性能分析报告

以下 list-performance-analysis-reports 示例列出了数据库 db-abcdefg123456789 的性能分析报告。响应列出了所有带有报告 ID、状态和时间段详细信息的报告。

```
aws pi list-performance-analysis-reports \
    --service-type RDS \
    --identifier db-abcdefg123456789
```

输出：

```
{
    "AnalysisReports": [
        {
            "Status": "Succeeded",
            "EndTime": 1680587086.584,
            "CreateTime": 1680587087.139,
            "StartTime": 1680583486.584,
            "AnalysisReportId": "report-0d99cc91c4422ee61"
        },
        {
            "Status": "Succeeded",
            "EndTime": 1681491137.914,
            "CreateTime": 1681491145.973,
            "StartTime": 1681487537.914,
            "AnalysisReportId": "report-002633115cc002233"
        },
        {
            "Status": "Succeeded",
            "EndTime": 1681493499.849,
            "CreateTime": 1681493507.762,
            "StartTime": 1681489899.849,
            "AnalysisReportId": "report-043b1e006b47246f9"
        },
        {
            "Status": "InProgress",
            "EndTime": 1682979503.0,
```

```
            "CreateTime": 1682979618.994,
            "StartTime": 1682969503.0,
            "AnalysisReportId": "report-01ad15f9b88bcbd56"
        }
    ]
}
```

有关性能分析报告的更多信息，请参阅《Amazon RDS 用户指南》中的分析一段时间内的数据库性能和《Amazon Aurora 用户指南》中的分析一段时间内的数据库性能。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListPerformanceAnalysisReports。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出性能分析报告的标签

以下 `list-tags-for-resource` 示例列出报告 ID 为 `report-0d99cc91c4422ee61` 的性能分析报告的标签。

```
aws pi list-tags-for-resource \
    --service-type RDS \
    --resource-arn arn:aws:pi:us-west-2:123456789012:perf-reports/RDS/db-abcdefg123456789/report-0d99cc91c4422ee61
```

输出：

```
{
    "Tags": [
        {
            "Value": "test-tag",
            "Key": "name"
        }
    ]
}
```

有关向性能分析报告添加标签的更多信息，请参阅《Amazon RDS 用户指南》中的在性能详情中向性能分析报告添加标签和《Amazon Aurora 用户指南》中的在性能详情中向性能分析报告添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

向性能分析报告添加标签

以下 tag-resource 示例将标签键 name 以及标签值 test-tag 添加到报告 ID 为 report-0d99cc91c4422ee61 的性能分析报告。

```
aws pi tag-resource \
    --service-type RDS \
    --resource-arn arn:aws:pi:us-west-2:123456789012:perf-reports/RDS/db-abcdefg123456789/report-0d99cc91c4422ee61 \
    --tags Key=name,Value=test-tag
```

此命令不生成任何输出。

有关向性能分析报告添加标签的更多信息，请参阅《Amazon RDS 用户指南》中的在性能详情中向性能分析报告添加标签和《Amazon Aurora 用户指南》中的在性能详情中向性能分析报告添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

删除性能分析报告的标签

以下 untag-resource 示例删除报告 ID 为 report-0d99cc91c4422ee61 的性能分析报告的标签 name。

```
aws pi untag-resource \
    --service-type RDS \
    --resource-arn arn:aws:pi:us-west-2:123456789012:perf-reports/RDS/db-abcdefg123456789/report-0d99cc91c4422ee61 \
    --tag-keys name
```

此命令不生成任何输出。

有关向性能分析报告添加标签的更多信息，请参阅《Amazon RDS 用户指南》中的在性能详情中向性能分析报告添加标签和《Amazon Aurora 用户指南》中的在性能详情中向性能分析报告添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

使用 Amazon CLI 的 Amazon Redshift 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Redshift 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-reserved-node-exchange`

以下代码示例演示了如何使用 `accept-reserved-node-exchange`。

Amazon CLI

接受预留节点交换

以下 `accept-reserved-node-exchange` 示例接受 DC1 预留节点与 DC2 预留节点的交换。

```
aws redshift accept-reserved-node-exchange /
    --reserved-node-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE /
    --target-reserved-node-offering-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE
```

输出：

```
{
    "ExchangedReservedNode": {
        "ReservedNodeId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
        "ReservedNodeOfferingId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
        "NodeType": "dc2.large",
        "StartTime": "2019-12-06T21:17:26Z",
        "Duration": 31536000,
        "FixedPrice": 0.0,
        "UsagePrice": 0.0,
        "CurrencyCode": "USD",
        "NodeCount": 1,
        "State": "exchanging",
        "OfferingType": "All Upfront",
        "RecurringCharges": [
            {
                "RecurringChargeAmount": 0.0,
                "RecurringChargeFrequency": "Hourly"
            }
        ],
        "ReservedNodeOfferingType": "Regular"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的使用 Amazon CLI 升级预留节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptReservedNodeExchange。

authorize-cluster-security-group-ingress

以下代码示例演示了如何使用 authorize-cluster-security-group-ingress。

Amazon CLI

授权访问 EC2 安全组 此示例授权访问指定的 Amazon EC2 安全组。命令：

```
aws redshift authorize-cluster-security-group-ingress --cluster-security-group-name
 mysecuritygroup --ec2-security-group-name myec2securitygroup --ec2-security-group-
owner-id 123445677890
```

授权访问 CIDR 范围 此示例授权访问 CIDR 范围。命令：

```
aws redshift authorize-cluster-security-group-ingress --cluster-security-group-name
 mysecuritygroup --cidrip 192.168.100.100/32
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeClusterSecurityGroupIngress。

authorize-snapshot-access

以下代码示例演示了如何使用 authorize-snapshot-access。

Amazon CLI

授权 Amazon 账户还原快照 此示例授权 Amazon 账户 444455556666 还原快照 my-snapshot-id。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift authorize-snapshot-access --snapshot-id my-snapshot-id --account-with-
restore-access 444455556666
```

结果：

```
{
   "Snapshot": {
      "Status": "available",
      "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
      "EstimatedSecondsToCompletion": 0,
      "AvailabilityZone": "us-east-1a",
      "ClusterVersion": "1.0",
      "MasterUsername": "adminuser",
      "Encrypted": false,
      "OwnerAccount": "111122223333",
      "BackupProgressInMegabytes": 11.0,
      "ElapsedTimeInSeconds": 0,
      "DBName": "dev",
      "CurrentBackupRateInMegabytesPerSecond: 0.1534,
```

```
        "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
        "ActualIncrementalBackupSizeInMegabytes": 11.0,
        "SnapshotType": "manual",
        "NodeType": "dw.hs1.xlarge",
        "ClusterIdentifier": "mycluster",
        "TotalBackupSizeInMegabytes": 20.0,
        "Port": 5439,
        "NumberOfNodes": 2,
        "SnapshotIdentifier": "my-snapshot-id"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeSnapshotAccess。

batch-delete-cluster-snapshots

以下代码示例演示了如何使用 batch-delete-cluster-snapshots。

Amazon CLI

删除一组集群快照

以下 batch-delete-cluster-snapshots 示例删除了一组手动集群快照。

```
aws redshift batch-delete-cluster-snapshots \
        --
identifiers SnapshotIdentifier=mycluster-2019-11-06-14-12 SnapshotIdentifier=mycluster-2019-
```

输出：

```
{
    "Resources": [
        "mycluster-2019-11-06-14-12",
        "mycluster-2019-11-06-14-20"
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteClusterSnapshots。

batch-modify-cluster-snapshots

以下代码示例演示了如何使用 batch-modify-cluster-snapshots。

Amazon CLI

修改一组集群快照

以下 batch-modify-cluster-snapshots 示例修改了一组集群快照的设置。

```
aws redshift batch-modify-cluster-snapshots \
    --snapshot-identifier-list mycluster-2019-11-06-16-31 mycluster-2019-11-06-16-32
 \
    --manual-snapshot-retention-period 30
```

输出：

```
{
    "Resources": [
        "mycluster-2019-11-06-16-31",
        "mycluster-2019-11-06-16-32"
    ],
    "Errors": [],
    "ResponseMetadata": {
        "RequestId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
        "HTTPStatusCode": 200,
        "HTTPHeaders": {
                "x-amzn-requestid": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE,
                "content-type": "text/xml",
                "content-length": "480",
                "date": "Sat, 07 Dec 2019 00:36:09 GMT",
                "connection": "keep-alive"
        },
        "RetryAttempts": 0
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchModifyClusterSnapshots。

cancel-resize

以下代码示例演示了如何使用 cancel-resize。

Amazon CLI

取消调整集群的大小

以下 cancel-resize 示例取消了集群经典的调整大小操作。

```
aws redshift cancel-resize \
    --cluster-identifier mycluster
```

输出：

```
{
    "TargetNodeType": "dc2.large",
    "TargetNumberOfNodes": 2,
    "TargetClusterType": "multi-node",
    "Status": "CANCELLING",
    "ResizeType": "ClassicResize",
    "TargetEncryptionType": "NONE"
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的调整 Amazon Redshift 中集群的大小。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelResize。

copy-cluster-snapshot

以下代码示例演示了如何使用 copy-cluster-snapshot。

Amazon CLI

获取所有集群版本的描述 此示例返回所有集群版本的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift copy-cluster-snapshot --source-snapshot-identifier
 cm:examplecluster-2013-01-22-19-27-58 --target-snapshot-identifier my-saved-
snapshot-copy
```

结果：

```
{
   "Snapshot": {
      "Status": "available",
      "SnapshotCreateTime": "2013-01-22T19:27:58.931Z",
      "AvailabilityZone": "us-east-1c",
      "ClusterVersion": "1.0",
      "MasterUsername": "adminuser",
      "DBName": "dev",
      "ClusterCreateTime": "2013-01-22T19:23:59.368Z",
      "SnapshotType": "manual",
      "NodeType": "dw.hs1.xlarge",
      "ClusterIdentifier": "examplecluster",
      "Port": 5439,
      "NumberOfNodes": "2",
      "SnapshotIdentifier": "my-saved-snapshot-copy"
   },
   "ResponseMetadata": {
      "RequestId": "3b279691-64e3-11e2-bec0-17624ad140dd"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyClusterSnapshot。

create-cluster-parameter-group

以下代码示例演示了如何使用 create-cluster-parameter-group。

Amazon CLI

创建集群参数组 此示例创建了一个新的集群参数组。命令：

```
aws redshift create-cluster-parameter-group --parameter-group-name
 myclusterparametergroup --parameter-group-family redshift-1.0 --description "My
 first cluster parameter group"
```

结果：

```
{
   "ClusterParameterGroup": {
      "ParameterGroupFamily": "redshift-1.0",
```

```
      "Description": "My first cluster parameter group",
      "ParameterGroupName": "myclusterparametergroup"
   },
   "ResponseMetadata": {
      "RequestId": "739448f0-64cc-11e2-8f7d-3b939af52818"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClusterParameterGroup。

create-cluster-security-group

以下代码示例演示了如何使用 create-cluster-security-group。

Amazon CLI

创建集群安全组 此示例创建了一个新的集群安全组。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift create-cluster-security-group --cluster-security-group-name
 mysecuritygroup --description "This is my cluster security group"
```

结果：

```
{
   "create_cluster_security_group_response": {
      "create_cluster_security_group_result": {
         "cluster_security_group": {
            "description": "This is my cluster security group",
            "owner_id": "300454760768",
            "cluster_security_group_name": "mysecuritygroup",
            "ec2_security_groups": \[],
            "ip_ranges": \[]
         }
      },
      "response_metadata": {
         "request_id": "5df486a0-343a-11e2-b0d8-d15d0ef48549"
      }
   }
}
```

您也可以使用 --output text 选项以文本格式获取相同的信息。命令：

--output text 选项。命令：

选项。命令：

```
aws redshift create-cluster-security-group --cluster-security-group-name
 mysecuritygroup --description "This is my cluster security group" --output text
```

结果：

```
This is my cluster security group   300454760768    mysecuritygroup
a0c0bfab-343a-11e2-95d2-c3dc9fe8ab57
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClusterSecurityGroup。

create-cluster-snapshot

以下代码示例演示了如何使用 create-cluster-snapshot。

Amazon CLI

创建集群快照 此示例创建了一个新的集群快照。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift create-cluster-snapshot --cluster-identifier mycluster --snapshot-
identifier my-snapshot-id
```

结果：

```
{
   "Snapshot": {
      "Status": "creating",
      "SnapshotCreateTime": "2013-01-22T22:20:33.548Z",
      "AvailabilityZone": "us-east-1a",
      "ClusterVersion": "1.0",
      "MasterUsername": "adminuser",
      "DBName": "dev",
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
      "SnapshotType": "manual",
      "NodeType": "dw.hs1.xlarge",
      "ClusterIdentifier": "mycluster",
      "Port": 5439,
```

```
      "NumberOfNodes": "2",
      "SnapshotIdentifier": "my-snapshot-id"
   },
   "ResponseMetadata": {
      "RequestId": "f024d1a5-64e1-11e2-88c5-53eb05787dfb"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClusterSnapshot。

create-cluster-subnet-group

以下代码示例演示了如何使用 create-cluster-subnet-group。

Amazon CLI

创建集群子网组 此示例创建了一个新的集群子网组。命令：

```
aws redshift create-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
 --description "My subnet group" --subnet-ids subnet-763fdd1c
```

结果：

```
{
   "ClusterSubnetGroup": {
      "Subnets": [
         {
            "SubnetStatus": "Active",
            "SubnetIdentifier": "subnet-763fdd1c",
            "SubnetAvailabilityZone": {
               "Name": "us-east-1a"
            }
         } ],
      "VpcId": "vpc-7e3fdd14",
      "SubnetGroupStatus": "Complete",
      "Description": "My subnet group",
      "ClusterSubnetGroupName": "mysubnetgroup"
   },
   "ResponseMetadata": {
      "RequestId": "500b8ce2-698f-11e2-9790-fd67517fb6fd"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateClusterSubnetGroup。

create-cluster

以下代码示例演示了如何使用 create-cluster。

Amazon CLI

使用最少参数创建集群 此示例使用最少参数集创建集群。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift create-cluster --node-type dw.hs1.xlarge --number-of-nodes 2 --master-
username adminuser --master-user-password TopSecret1 --cluster-identifier mycluster
```

结果：

```
{
   "Cluster": {
      "NodeType": "dw.hs1.xlarge",
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
      "ClusterParameterGroups": [
         {
            "ParameterApplyStatus": "in-sync",
            "ParameterGroupName": "default.redshift-1.0"
         } ],
      "ClusterSecurityGroups": [
         {
            "Status": "active",
            "ClusterSecurityGroupName": "default"
         } ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "creating",
      "ClusterIdentifier": "mycluster",
      "DBName": "dev",
      "NumberOfNodes": 2,
      "PendingModifiedValues": {
         "MasterUserPassword": "\****"
```

```
        }
    },
    "ResponseMetadata": {
        "RequestId": "7cf4bcfc-64dd-11e2-bea9-49e0ce183f07"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCluster。

create-event-subscription

以下代码示例演示了如何使用 create-event-subscription。

Amazon CLI

创建事件通知订阅

以下 create-event-subscription 示例创建了事件通知订阅。

```
aws redshift create-event-subscription \
    --subscription-name mysubscription \
    --sns-topic-arn arn:aws:sns:us-west-2:123456789012:MySNStopic \
    --source-type cluster \
    --source-ids mycluster
```

输出：

```
{
        "EventSubscription": {
        "CustomerAwsId": "123456789012",
        "CustSubscriptionId": "mysubscription",
        "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNStopic",
        "Status": "active",
        "SubscriptionCreationTime": "2019-12-09T20:05:19.365Z",
        "SourceType": "cluster",
        "SourceIdsList": [
            "mycluster"
        ],
        "EventCategoriesList": [],
        "Severity": "INFO",
        "Enabled": true,
        "Tags": []
```

```
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的订阅 Amazon Redshift 事件通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateEventSubscription。

create-hsm-client-certificate

以下代码示例演示了如何使用 create-hsm-client-certificate。

Amazon CLI

创建 HSM 客户端证书

以下 create-hsm-client-certificate 示例生成了可供集群连接到 HSM 的 HSM 客户端证书。

```
aws redshift create-hsm-client-certificate \
    --hsm-client-certificate-identifier myhsmclientcert
```

输出：

```
{
    "HsmClientCertificate": {
        "HsmClientCertificateIdentifier": "myhsmclientcert",
        "HsmClientCertificatePublicKey": "-----BEGIN CERTIFICATE-----
        MIICiEXAMPLECQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
        VVMxCzAJBgNVBAgTEXAMPLEwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
        b24xFDASBgNVBAsTC0lBTSBDb25EXAMPLEIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
        BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb2EXAMPLETEwNDI1MjA0NTIxWhcN
        MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBEXAMPLEMRAwDgYD
        EXAMPLETZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
        b2xlMRIwEAEXAMPLEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
        YXpvbi5jb20wgZ8wDQYJKEXAMPLEAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
        21uUSfwfEvySWtC2XADZ4nB+BLYgVIk6EXAMPLE3G93vUEIO3IyNoH/f0wYK8m9T
        rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugEXAMPLEzZswY6786m86gpE
        Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEEXAMPLEEAtCu4
        nUhVVxYUEXAMPLEh8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
        FFBjvSfpJIlJ00zbhNYS5f6GEXAMPLEl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
        NYiytVbZPQUQ5Yaxu2jXnimvw3rEXAMPLE=-----END CERTIFICATE-----\n",
    "Tags": []
    }
```

```
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift API 权限参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHsmClientCertificate。

create-hsm-configuration

以下代码示例演示了如何使用 create-hsm-configuration。

Amazon CLI

创建 HSM 配置

以下 create-hsm-configuration 示例创建了指定的 HSM 配置，其中包含集群在硬件安全模块（HSM）中存储并使用数据库加密密钥所需的信息。

```
aws redshift create-hsm-configuration /
    --hsm-configuration-identifier myhsmconnection
    --description "My HSM connection"
    --hsm-ip-address 192.0.2.09
    --hsm-partition-name myhsmpartition /
    --hsm-partition-password A1b2c3d4 /
    --hsm-server-public-certificate myhsmclientcert
```

输出：

```
{
    "HsmConfiguration": {
        "HsmConfigurationIdentifier": "myhsmconnection",
        "Description": "My HSM connection",
        "HsmIpAddress": "192.0.2.09",
        "HsmPartitionName": "myhsmpartition",
        "Tags": []
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHsmConfiguration。

create-snapshot-copy-grant

以下代码示例演示了如何使用 create-snapshot-copy-grant。

Amazon CLI

创建快照复制授予

以下 create-snapshot-copy-grant 示例创建了快照复制授予，并对目标 Amazon 区域中复制的快照进行了加密。

```
aws redshift create-snapshot-copy-grant \
    --snapshot-copy-grant-name mysnapshotcopygrantname
```

输出：

```
{
    "SnapshotCopyGrant": {
        "SnapshotCopyGrantName": "mysnapshotcopygrantname",
        "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
bPxRfih3yCo8nvbEXAMPLEKEY",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 数据库加密。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshotCopyGrant。

create-snapshot-schedule

以下代码示例演示了如何使用 create-snapshot-schedule。

Amazon CLI

创建快照计划

以下 create-snapshot-schedule 示例使用指定的描述和每 12 小时一次的速率创建了快照计划。

```
aws redshift create-snapshot-schedule \
    --schedule-definitions "rate(12 hours)" \
    --schedule-identifier mysnapshotschedule \
    --schedule-description "My schedule description"
```

输出：

```
{
    "ScheduleDefinitions": [
        "rate(12 hours)"
    ],
    "ScheduleIdentifier": "mysnapshotschedule",
    "ScheduleDescription": "My schedule description",
    "Tags": []
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的自动快照计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSnapshotSchedule。

create-tags

以下代码示例演示了如何使用 create-tags。

Amazon CLI

为集群创建标签

以下 create-tags 示例将指定的标签键/值对添加到指定的集群。

```
aws redshift create-tags \
    --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \
    --tags "Key"="mytags","Value"="tag1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的在 Amazon Redshift 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTags。

delete-cluster-parameter-group

以下代码示例演示了如何使用 delete-cluster-parameter-group。

Amazon CLI

删除集群参数组 此示例删除了集群参数组。命令：

```
aws redshift delete-cluster-parameter-group --parameter-group-name
 myclusterparametergroup
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClusterParameterGroup。

`delete-cluster-security-group`

以下代码示例演示了如何使用 `delete-cluster-security-group`。

Amazon CLI

删除集群安全组 此示例删除了集群安全组。命令：

```
aws redshift delete-cluster-security-group --cluster-security-group-name
 mysecuritygroup
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClusterSecurityGroup。

`delete-cluster-snapshot`

以下代码示例演示了如何使用 `delete-cluster-snapshot`。

Amazon CLI

删除集群快照 此示例删除了集群快照。命令：

```
aws redshift delete-cluster-snapshot --snapshot-identifier my-snapshot-id
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClusterSnapshot。

`delete-cluster-subnet-group`

以下代码示例演示了如何使用 `delete-cluster-subnet-group`。

Amazon CLI

删除集群子网组 此示例删除了集群子网组。命令：

```
aws redshift delete-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
```

结果：

```
{
   "ResponseMetadata": {
      "RequestId": "253fbffd-6993-11e2-bc3a-47431073908a"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteClusterSubnetGroup。

delete-cluster

以下代码示例演示了如何使用 delete-cluster。

Amazon CLI

删除没有最终集群快照的集群 此示例删除集群，强制删除数据，因此不会创建最终集群快照。命令：

```
aws redshift delete-cluster --cluster-identifier mycluster --skip-final-cluster-snapshot
```

删除集群，支持使用最终集群快照 此示例删除集群，但指定最终集群快照。命令：

```
aws redshift delete-cluster --cluster-identifier mycluster --final-cluster-snapshot-identifier myfinalsnapshot
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCluster。

delete-event-subscription

以下代码示例演示了如何使用 delete-event-subscription。

Amazon CLI

删除事件订阅

以下 delete-event-subscription 示例删除了指定的事件通知订阅。

```
aws redshift delete-event-subscription \
```

```
    --subscription-name mysubscription
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的订阅 Amazon Redshift 事件通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEventSubscription。

delete-hsm-client-certificate

以下代码示例演示了如何使用 delete-hsm-client-certificate。

Amazon CLI

删除 HSM 客户端证书

以下 delete-hsm-client-certificate 示例删除了 HSM 客户端证书。

```
aws redshift delete-hsm-client-certificate \
    --hsm-client-certificate-identifier myhsmclientcert
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift API 权限参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHsmClientCertificate。

delete-hsm-configuration

以下代码示例演示了如何使用 delete-hsm-configuration。

Amazon CLI

删除 HSM 配置

以下 delete-hsm-configuration 示例从当前 Amazon 账户中删除了指定的 HSM 配置。

```
aws redshift delete-hsm-configuration /
    --hsm-configuration-identifier myhsmconnection
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHsmConfiguration。

delete-scheduled-action

以下代码示例演示了如何使用 delete-scheduled-action。

Amazon CLI

删除计划操作

以下 delete-scheduled-action 示例删除指定的计划操作。

```
aws redshift delete-scheduled-action \
    --scheduled-action-name myscheduledaction
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteScheduledAction。

delete-snapshot-copy-grant

以下代码示例演示了如何使用 delete-snapshot-copy-grant。

Amazon CLI

删除快照复制授予

以下 delete-snapshot-copy-grant 示例删除了指定的快照复制授予。

```
aws redshift delete-snapshot-copy-grant \
    --snapshot-copy-grant-name mysnapshotcopygrantname
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 数据库加密。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSnapshotCopyGrant。

delete-snapshot-schedule

以下代码示例演示了如何使用 delete-snapshot-schedule。

Amazon CLI

删除快照计划

以下 delete-snapshot-schedule 示例删除了指定的快照计划。在删除计划之前，必须取消集群的关联。

```
aws redshift delete-snapshot-schedule \
    --schedule-identifier mysnapshotschedule
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的自动快照计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSnapshotSchedule。

delete-tags

以下代码示例演示了如何使用 delete-tags。

Amazon CLI

从集群中删除标签

以下 delete-tags 示例从指定集群中移除具有指定键名的标签。

```
aws redshift delete-tags \
    --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \
    --tag-keys "clustertagkey" "clustertagvalue"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的在 Amazon Redshift 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTags。

describe-account-attributes

以下代码示例演示了如何使用 describe-account-attributes。

Amazon CLI

描述 Amazon 账户的属性

以下 describe-account-attributes 示例显示了与 Amazon 调用账户关联的属性。

```
aws redshift describe-account-attributes
```

输出：

```
{
    "AccountAttributes": [
        {
            "AttributeName": "max-defer-maintenance-duration",
            "AttributeValues": [
                {
                    "AttributeValue": "45"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAttributes。

describe-cluster-db-revisions

以下代码示例演示了如何使用 describe-cluster-db-revisions。

Amazon CLI

描述集群的数据库版本

以下 describe-cluster-db-revisions 示例显示了指定集群的 ClusterDbRevision 对象数组的详细信息。

```
aws redshift describe-cluster-db-revisions \
    --cluster-identifier mycluster
```

输出：

```
{
    "ClusterDbRevisions": [
        {
            "ClusterIdentifier": "mycluster",
```

```
            "CurrentDatabaseRevision": "11420",
            "DatabaseRevisionReleaseDate": "2019-11-22T16:43:49.597Z",
            "RevisionTargets": []
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterDbRevisions。

`describe-cluster-parameter-groups`

以下代码示例演示了如何使用 `describe-cluster-parameter-groups`。

Amazon CLI

获取所有集群参数组的描述 此示例返回账户中所有集群参数组的描述以及列标题。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-parameter-groups
```

结果：

```
{
   "ParameterGroups": [
      {
         "ParameterGroupFamily": "redshift-1.0",
         "Description": "My first cluster parameter group",
         "ParameterGroupName": "myclusterparametergroup"
      } ],
   "ResponseMetadata": {
      "RequestId": "8ceb8f6f-64cc-11e2-bea9-49e0ce183f07"
   }
}
```

您也可以使用 `--output text` 选项以文本格式获取相同的信息。命令：

`--output text` 选项。命令：

选项。命令：

```
aws redshift describe-cluster-parameter-groups --output text
```

结果：

```
redshift-1.0        My first cluster parameter group        myclusterparametergroup
RESPONSEMETADATA    9e665a36-64cc-11e2-8f7d-3b939af52818
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterParameterGroups。

`describe-cluster-parameters`

以下代码示例演示了如何使用 `describe-cluster-parameters`。

Amazon CLI

检索指定集群参数组的参数 此示例检索指定参数组的参数。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-parameters --parameter-group-name
 myclusterparametergroup
```

结果：

```
{
   "Parameters": [
      {
         "Description": "Sets the display format for date and time values.",
         "DataType": "string",
         "IsModifiable": true,
         "Source": "engine-default",
         "ParameterValue": "ISO, MDY",
         "ParameterName": "datestyle"
      },
      {
         "Description": "Sets the number of digits displayed for floating-point
 values",
         "DataType": "integer",
         "IsModifiable": true,
         "AllowedValues": "-15-2",
         "Source": "engine-default",
         "ParameterValue": "0",
         "ParameterName": "extra_float_digits"
      },
```

```
      (...remaining output omitted...)
   ]
}
```

您也可以使用 --output text 选项以文本格式获取相同的信息。命令：

--output text 选项。命令：

选项。命令：

```
aws redshift describe-cluster-parameters --parameter-group-name
 myclusterparametergroup --output text
```

结果：

```
RESPONSEMETADATA    cdac40aa-64cc-11e2-9e70-918437dd236d
Sets the display format for date and time values.   string  True    engine-default
 ISO, MDY        datestyle
Sets the number of digits displayed for floating-point values       integer True
 -15-2   engine-default  0       extra_float_digits
This parameter applies a user-defined label to a group of queries that are run
 during the same session..    string  True    engine-default  default query_group
require ssl for all databaseconnections     boolean True    true,false      engine-
default  false   require_ssl
Sets the schema search order for names that are not schema-qualified.       string
 True    engine-default  $user, public   search_path
Aborts any statement that takes over the specified number of milliseconds.  integer
 True    engine-default  0       statement_timeout
wlm json configuration      string  True    engine-default
 \[{"query_concurrency":5}]      wlm_json_configuration
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterParameters。

describe-cluster-security-groups

以下代码示例演示了如何使用 describe-cluster-security-groups。

Amazon CLI

获取所有集群安全组的描述 此示例返回账户中所有集群安全组的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-security-groups
```

结果：

```
{
   "ClusterSecurityGroups": [
      {
         "OwnerId": "100447751468",
         "Description": "default",
         "ClusterSecurityGroupName": "default",
         "EC2SecurityGroups": \[],
         "IPRanges": [
            {
               "Status": "authorized",
               "CIDRIP": "0.0.0.0/0"
            }
         ]
      },
      {
         "OwnerId": "100447751468",
         "Description": "This is my cluster security group",
         "ClusterSecurityGroupName": "mysecuritygroup",
         "EC2SecurityGroups": \[],
         "IPRanges": \[]
      },
      (...remaining output omitted...)
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterSecurityGroups。

`describe-cluster-snapshots`

以下代码示例演示了如何使用 `describe-cluster-snapshots`。

Amazon CLI

获取所有集群快照的描述 此示例返回账户中所有集群快照的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-snapshots
```

结果：

```
{
   "Snapshots": [
      {
         "Status": "available",
         "SnapshotCreateTime": "2013-07-17T22:02:22.852Z",
         "EstimatedSecondsToCompletion": -1,
         "AvailabilityZone": "us-east-1a",
         "ClusterVersion": "1.0",
         "MasterUsername": "adminuser",
         "Encrypted": false,
         "OwnerAccount": "111122223333",
         "BackupProgressInMegabytes": 20.0,
         "ElapsedTimeInSeconds": 0,
         "DBName": "dev",
         "CurrentBackupRateInMegabytesPerSecond: 0.0,
         "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
         "ActualIncrementalBackupSizeInMegabytes"; 20.0
         "SnapshotType": "automated",
         "NodeType": "dw.hs1.xlarge",
         "ClusterIdentifier": "mycluster",
         "Port": 5439,
         "TotalBackupSizeInMegabytes": 20.0,
         "NumberOfNodes": "2",
         "SnapshotIdentifier": "cm:mycluster-2013-01-22-22-04-18"
      },
      {
         "EstimatedSecondsToCompletion": 0,
         "OwnerAccount": "111122223333",
         "CurrentBackupRateInMegabytesPerSecond: 0.1534,
         "ActualIncrementalBackupSizeInMegabytes"; 11.0,
         "NumberOfNodes": "2",
         "Status": "available",
         "ClusterVersion": "1.0",
         "MasterUsername": "adminuser",
         "AccountsWithRestoreAccess": [
            {
               "AccountID": "444455556666"
            } ],
         "TotalBackupSizeInMegabytes": 20.0,
         "DBName": "dev",
         "BackupProgressInMegabytes": 11.0,
         "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
```

```
         "ElapsedTimeInSeconds": 0,
         "ClusterIdentifier": "mycluster",
         "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
         "AvailabilityZone": "us-east-1a",
         "NodeType": "dw.hs1.xlarge",
         "Encrypted": false,
         "SnapshotType": "manual",
         "Port": 5439,
         "SnapshotIdentifier": "my-snapshot-id"
      } ]
   }
   (...remaining output omitted...)
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterSnapshots。

describe-cluster-subnet-groups

以下代码示例演示了如何使用 describe-cluster-subnet-groups。

Amazon CLI

获取所有集群子网组的描述 此示例返回所有集群子网组的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-subnet-groups
```

结果：

```
{
   "ClusterSubnetGroups": [
      {
         "Subnets": [
            {
               "SubnetStatus": "Active",
               "SubnetIdentifier": "subnet-763fdd1c",
               "SubnetAvailabilityZone": {
                  "Name": "us-east-1a"
               }
            }
         ],
         "VpcId": "vpc-7e3fdd14",
         "SubnetGroupStatus": "Complete",
```

```
            "Description": "My subnet group",
            "ClusterSubnetGroupName": "mysubnetgroup"
        }
    ],
    "ResponseMetadata": {
        "RequestId": "37fa8c89-6990-11e2-8f75-ab4018764c77"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterSubnetGroups。

describe-cluster-tracks

以下代码示例演示了如何使用 describe-cluster-tracks。

Amazon CLI

描述集群跟踪

以下 describe-cluster-tracks 示例显示了可用维护跟踪的详细信息。

```
aws redshift describe-cluster-tracks \
    --maintenance-track-name current
```

输出：

```
{
    "MaintenanceTracks": [
        {
            "MaintenanceTrackName": "current",
            "DatabaseVersion": "1.0.11420",
            "UpdateTargets": [
                {
                    "MaintenanceTrackName": "preview_features",
                    "DatabaseVersion": "1.0.11746",
                    "SupportedOperations": [
                        {
                            "OperationName": "restore-from-cluster-snapshot"
                        }
                    ]
                },
                {
```

```
                    "MaintenanceTrackName": "trailing",
                    "DatabaseVersion": "1.0.11116",
                    "SupportedOperations": [
                        {
                            "OperationName": "restore-from-cluster-snapshot"
                        },
                        {
                            "OperationName": "modify-cluster"
                        }
                    ]
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的选择集群维护跟踪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterTracks。

`describe-cluster-versions`

以下代码示例演示了如何使用 `describe-cluster-versions`。

Amazon CLI

获取所有集群版本的描述 此示例返回所有集群版本的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-cluster-versions
```

结果：

```
{
   "ClusterVersions": [
      {
      "ClusterVersion": "1.0",
      "Description": "Initial release",
      "ClusterParameterGroupFamily": "redshift-1.0"
      } ],
   "ResponseMetadata": {
      "RequestId": "16a53de3-64cc-11e2-bec0-17624ad140dd"
   }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusterVersions。

`describe-clusters`

以下代码示例演示了如何使用 `describe-clusters`。

Amazon CLI

获取所有集群的描述 此示例返回账户中所有集群的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-clusters
```

结果：

```
{
   "Clusters": [
   {
      "NodeType": "dw.hs1.xlarge",
      "Endpoint": {
         "Port": 5439,
         "Address": "mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com"
      },
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
      "ClusterParameterGroups": [
         {
            "ParameterApplyStatus": "in-sync",
            "ParameterGroupName": "default.redshift-1.0"
         } ],
      "ClusterSecurityGroups": [
         {
            "Status": "active",
            "ClusterSecurityGroupName": "default"
         } ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "AvailabilityZone": "us-east-1a",
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
```

```
      "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "available",
      "ClusterIdentifier": "mycluster",
      "DBName": "dev",
      "NumberOfNodes": 2,
      "PendingModifiedValues": {}
   } ],
   "ResponseMetadata": {
      "RequestId": "65b71cac-64df-11e2-8f5b-e90bd6c77476"
   }
}
```

您也可以使用 --output text 选项以文本格式获取相同的信息。命令：

--output text 选项。命令：

选项。命令：

```
aws redshift describe-clusters --output text
```

结果：

```
dw.hs1.xlarge       1.0     true    adminuser       True    us-east-1a
 2013-01-22T21:59:29.559Z        sat:03:30-sat:04:00     1       available
 mycluster       dev     2
ENDPOINT    5439    mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com
in-sync     default.redshift-1.0
active      default
PENDINGMODIFIEDVALUES
RESPONSEMETADATA    934281a8-64df-11e2-b07c-f7fbdd006c67
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeClusters。

describe-default-cluster-parameters

以下代码示例演示了如何使用 describe-default-cluster-parameters。

Amazon CLI

获取默认集群参数的描述 此示例返回 redshift-1.0 系列中默认集群参数的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-default-cluster-parameters --parameter-group-family
 redshift-1.0
```

结果：

```
{
   "DefaultClusterParameters": {
   "ParameterGroupFamily": "redshift-1.0",
   "Parameters": [
      {
         "Description": "Sets the display format for date and time values.",
         "DataType": "string",
         "IsModifiable": true,
         "Source": "engine-default",
         "ParameterValue": "ISO, MDY",
         "ParameterName": "datestyle"
      },
      {
         "Description": "Sets the number of digits displayed for floating-point
 values",
         "DataType": "integer",
         "IsModifiable": true,
         "AllowedValues": "-15-2",
         "Source": "engine-default",
         "ParameterValue": "0",
         "ParameterName": "extra_float_digits"
      },
      (...remaining output omitted...)
      ]
   }
}
```

要查看有效参数组系列的列表，请使用 describe-cluster-parameter-groups 命令。

describe-cluster-parameter-groups 命令。

命令。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDefaultClusterParameters。

describe-event-categories

以下代码示例演示了如何使用 describe-event-categories。

Amazon CLI

描述集群的事件类别

以下 describe-event-categories 示例显示了集群的事件类别的详细信息。

```
aws redshift describe-event-categories \
    --source-type cluster
```

输出：

```
{
    "EventCategoriesMapList": [
        {
            "SourceType": "cluster",
            "Events": [
                {
                    "EventId": "REDSHIFT-EVENT-2000",
                    "EventCategories": [
                        "management"
                    ],
                    "EventDescription": "Cluster <cluster name> created at <time in
 UTC>.",
                    "Severity": "INFO"
                },
                {
                    "EventId": "REDSHIFT-EVENT-2001",
                    "EventCategories": [
                        "management"
                    ],
                    "EventDescription": "Cluster <cluster name> deleted at <time in
 UTC>.",
                    "Severity": "INFO"
                },
                {
                    "EventId": "REDSHIFT-EVENT-3625",
                    "EventCategories": [
                        "monitoring"
                    ],
```

```
                    "EventDescription": "The cluster <cluster name> can't be resumed
 with its previous elastic network interface <ENI id>. We will allocate a new
 elastic network interface and associate it with the cluster node.",
                    "Severity": "INFO"
                }
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventCategories。

describe-event-subscriptions

以下代码示例演示了如何使用 describe-event-subscriptions。

Amazon CLI

描述事件订阅

以下 describe-event-subscriptions 示例显示指定订阅的事件通知订阅。

```
aws redshift describe-event-subscriptions \
    --subscription-name mysubscription
```

输出：

```
{
    "EventSubscriptionsList": [
        {
            "CustomerAwsId": "123456789012",
            "CustSubscriptionId": "mysubscription",
            "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNStopic",
            "Status": "active",
            "SubscriptionCreationTime": "2019-12-09T21:50:21.332Z",
            "SourceIdsList": [],
            "EventCategoriesList": [
                "management"
            ],
            "Severity": "ERROR",
            "Enabled": true,
            "Tags": []
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的订阅 Amazon Redshift 事件通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEventSubscriptions。

`describe-events`

以下代码示例演示了如何使用 `describe-events`。

Amazon CLI

描述所有事件 此示例返回所有事件。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-events
```

结果：

```
{
   "Events": [
      {
      "Date": "2013-01-22T19:17:03.640Z",
      "SourceIdentifier": "myclusterparametergroup",
      "Message": "Cluster parameter group myclusterparametergroup has been
 created.",
      "SourceType": "cluster-parameter-group"
      } ],
   "ResponseMetadata": {
      "RequestId": "9f056111-64c9-11e2-9390-ff04f2c1e638"
   }
}
```

您也可以使用 `--output text` 选项以文本格式获取相同的信息。命令：

`--output text` 选项。命令：

选项。命令：

```
aws redshift describe-events --output text
```

结果：

```
2013-01-22T19:17:03.640Z    myclusterparametergroup Cluster parameter group
 myclusterparametergroup has been created.       cluster-parameter-group
RESPONSEMETADATA    8e5fe765-64c9-11e2-bce3-e56f52c50e17
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEvents。

describe-hsm-client-certificates

以下代码示例演示了如何使用 describe-hsm-client-certificates。

Amazon CLI

描述 HSM 客户端证书

以下 describe-hsm-client-certificates 示例显示指定 HSM 客户端的详细信息。

```
aws redshift describe-hsm-client-certificates \
    --hsm-client-certificate-identifier myhsmclientcert
```

输出：

```
{
    "HsmClientCertificates": [
        {
        "HsmClientCertificateIdentifier": "myhsmclientcert",
        "HsmClientCertificatePublicKey": "-----BEGIN CERTIFICATE-----\
        EXAMPLECAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
        VVMxCzAJBgNVBAEXAMPLERAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
        b24xFDASBgNVBAsTC0lBTSBDb25zEXAMPLEwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
        BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhEXAMPLEDI1MjA0EXAMPLEN
        EXAMPLE0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
        VQQHEwdTZWF0dGEXAMPLEQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
        b2xlMRIwEAYDVQQDEwlUZXN0Q2lsEXAMPLEdBgkqhkiG9w0BCQEWEG5vb25lQGFt
        YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIEXAMPLEMaK0dn+a4GmWIWJ
        21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
        rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY67EXAMPLEE
        EXAMPLEZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
        nUhVVxYUntneD9EXAMPLE6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
        FFBjvSfpJIlJ00zbhNYS5f6GuoEDEXAMPLEBHjJnyp378OD8uTs7fLvjx79LjSTb
        NYiytVbZPQUQ5Yaxu2jXnimvw3rEXAMPLE=-----END CERTIFICATE-----\n",
```

```
        "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift API 权限参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHsmClientCertificates。

describe-hsm-configurations

以下代码示例演示了如何使用 describe-hsm-configurations。

Amazon CLI

描述 HSM 配置

以下 describe-hsm-configurations 示例显示了 Amazon 调用账户的可用 HSM 配置的详细信息。

```
aws redshift describe-hsm-configurations /
    --hsm-configuration-identifier myhsmconnection
```

输出：

```
{
    "HsmConfigurations": [
        {
            "HsmConfigurationIdentifier": "myhsmconnection",
            "Description": "My HSM connection",
            "HsmIpAddress": "192.0.2.09",
            "HsmPartitionName": "myhsmpartition",
            "Tags": []
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHsmConfigurations。

describe-logging-status

以下代码示例演示了如何使用 describe-logging-status。

Amazon CLI

描述集群的记录状态

以下 describe-logging-status 示例显示是否为集群记录信息（例如查询和连接尝试次数）。

```
aws redshift describe-logging-status \
    --cluster-identifier mycluster
```

输出：

```
{
    "LoggingEnabled": false
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的数据库审计日志记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLoggingStatus。

describe-node-configuration-options

以下代码示例演示了如何使用 describe-node-configuration-options。

Amazon CLI

描述节点配置选项

以下 describe-node-configuration-options 示例显示可能节点配置的属性，例如节点类型、节点数以及指定集群快照的磁盘使用情况。

```
aws redshift describe-node-configuration-options \
    --action-type restore-cluster \
    --snapshot-identifier rs:mycluster-2019-12-09-16-42-43
```

输出：

```
{
    "NodeConfigurationOptionList": [
        {
            "NodeType": "dc2.large",
```

```
            "NumberOfNodes": 2,
            "EstimatedDiskUtilizationPercent": 19.61
        },
        {
            "NodeType": "dc2.large",
            "NumberOfNodes": 4,
            "EstimatedDiskUtilizationPercent": 9.96
        },
        {
            "NodeType": "ds2.xlarge",
            "NumberOfNodes": 2,
            "EstimatedDiskUtilizationPercent": 1.53
        },
        {
            "NodeType": "ds2.xlarge",
            "NumberOfNodes": 4,
            "EstimatedDiskUtilizationPercent": 0.78
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的购买 Amazon Redshift 预留节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNodeConfigurationOptions。

describe-orderable-cluster-options

以下代码示例演示了如何使用 describe-orderable-cluster-options。

Amazon CLI

描述所有可排序集群选项 此示例返回所有可订购集群选项的描述。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift describe-orderable-cluster-options
```

结果：

```
{
   "OrderableClusterOptions": [
      {
```

```
         "NodeType": "dw.hs1.8xlarge",
         "AvailabilityZones": [
            { "Name": "us-east-1a" },
            { "Name": "us-east-1b" },
            { "Name": "us-east-1c" } ],
         "ClusterVersion": "1.0",
         "ClusterType": "multi-node"
      },
      {
         "NodeType": "dw.hs1.xlarge",
         "AvailabilityZones": [
            { "Name": "us-east-1a" },
            { "Name": "us-east-1b" },
            { "Name": "us-east-1c" } ],
         "ClusterVersion": "1.0",
         "ClusterType": "multi-node"
      },
      {
      "NodeType": "dw.hs1.xlarge",
      "AvailabilityZones": [
         { "Name": "us-east-1a" },
         { "Name": "us-east-1b" },
         { "Name": "us-east-1c" } ],
      "ClusterVersion": "1.0",
      "ClusterType": "single-node"
      } ],
   "ResponseMetadata": {
      "RequestId": "f6000035-64cb-11e2-9135-ff82df53a51a"
   }
}
```

您也可以使用 --output text 选项以文本格式获取相同的信息。命令：

--output text 选项。命令：

选项。命令：

```
aws redshift describe-orderable-cluster-options --output text
```

结果：

```
dw.hs1.8xlarge      1.0     multi-node
```

```
us-east-1a
us-east-1b
us-east-1c
dw.hs1.xlarge       1.0     multi-node
us-east-1a
us-east-1b
us-east-1c
dw.hs1.xlarge       1.0     single-node
us-east-1a
us-east-1b
us-east-1c
RESPONSEMETADATA    e648696b-64cb-11e2-bec0-17624ad140dd
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrderableClusterOptions。

`describe-reserved-node-offerings`

以下代码示例演示了如何使用 `describe-reserved-node-offerings`。

Amazon CLI

描述预留节点产品 此示例显示了所有可供购买的预留节点产品。命令：

```
aws redshift describe-reserved-node-offerings
```

结果：

```
{
   "ReservedNodeOfferings": [
      {
         "OfferingType": "Heavy Utilization",
         "FixedPrice": "",
         "NodeType": "dw.hs1.xlarge",
         "UsagePrice": "",
         "RecurringCharges": [
            {
               "RecurringChargeAmount": "",
               "RecurringChargeFrequency": "Hourly"
            } ],
         "Duration": 31536000,
         "ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
      },
```

```
      {
         "OfferingType": "Heavy Utilization",
         "FixedPrice": "",
         "NodeType": "dw.hs1.8xlarge",
         "UsagePrice": "",
         "RecurringCharges": [
            {
            "RecurringChargeAmount": "",
            "RecurringChargeFrequency": "Hourly"
            } ],
         "Duration": 31536000,
         "ReservedNodeOfferingId": "e5a2ff3b-352d-4a9c-ad7d-373c4cab5dd2"
      },
      ...remaining output omitted...
   ],
   "ResponseMetadata": {
      "RequestId": "8b1a1a43-75ff-11e2-9666-e142fe91ddd1"
   }
}
```

如果您想购买预留节点产品，则可以使用有效的 ReservedNodeOfferingId 调用 `purchase-reserved-node-offering`。

`purchase-reserved-node-offering` 使用有效的 ReservedNodeOfferingId。

使用有效的 ReservedNodeOfferingId。

ReservedNodeOfferingId。

.

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedNodeOfferings。

`describe-reserved-nodes`

以下代码示例演示了如何使用 `describe-reserved-nodes`。

Amazon CLI

描述预留节点 此示例显示已购买的预留节点产品。命令：

```
aws redshift describe-reserved-nodes
```

结果：

```
{
   "ResponseMetadata": {
      "RequestId": "bc29ce2e-7600-11e2-9949-4b361e7420b7"
   },
   "ReservedNodes": [
      {
         "OfferingType": "Heavy Utilization",
         "FixedPrice": "",
         "NodeType": "dw.hs1.xlarge",
         "ReservedNodeId": "1ba8e2e3-bc01-4d65-b35d-a4a3e931547e",
         "UsagePrice": "",
         "RecurringCharges": [
            {
               "RecurringChargeAmount": "",
               "RecurringChargeFrequency": "Hourly"
            } ],
         "NodeCount": 1,
         "State": "payment-pending",
         "StartTime": "2013-02-13T17:08:39.051Z",
         "Duration": 31536000,
         "ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
      }
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeReservedNodes。

describe-resize

以下代码示例演示了如何使用 describe-resize。

Amazon CLI

描述调整大小 此示例描述了集群的最新调整大小。请求的是 3 个 dw.hs1.8xlarge 类型的节点。
命令：

```
aws redshift describe-resize --cluster-identifier mycluster
```

结果：

```
{
   "Status": "NONE",
   "TargetClusterType": "multi-node",
   "TargetNodeType": "dw.hs1.8xlarge",
   "ResponseMetadata": {
      "RequestId": "9f52b0b4-7733-11e2-aa9b-318b2909bd27"
   },
   "TargetNumberOfNodes": "3"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResize。

describe-scheduled-actions

以下代码示例演示了如何使用 describe-scheduled-actions。

Amazon CLI

描述计划的操作

以下 describe-scheduled-actions 示例显示了所有当前计划操作的详细信息。

```
aws redshift describe-scheduled-actions
```

输出：

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "resizecluster",
            "TargetAction": {
                "ResizeCluster": {
                    "ClusterIdentifier": "mycluster",
                    "NumberOfNodes": 4,
                    "Classic": false
                }
            },
            "Schedule": "at(2019-12-10T00:07:00)",
            "IamRole": "arn:aws:iam::123456789012:role/myRedshiftRole",
            "State": "ACTIVE",
            "NextInvocations": [
```

```
                "2019-12-10T00:07:00Z"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeScheduledActions。

describe-snapshot-copy-grants

以下代码示例演示了如何使用 describe-snapshot-copy-grants。

Amazon CLI

描述快照复制授予

以下 describe-snapshot-copy-grants 示例显示了指定的集群快照复制授予的详细信息。

```
aws redshift describe-snapshot-copy-grants \
    --snapshot-copy-grant-name mysnapshotcopygrantname
```

输出：

```
{
    "SnapshotCopyGrants": [
        {
            "SnapshotCopyGrantName": "mysnapshotcopygrantname",
            "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
bPxRfih3yCo8nvbEXAMPLEKEY",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 数据库加密。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshotCopyGrants。

describe-snapshot-schedules

以下代码示例演示了如何使用 describe-snapshot-schedules。

Amazon CLI

描述快照计划

以下 describe-snapshot-schedules 示例显示了指定的集群快照计划的详细信息。

```
aws redshift describe-snapshot-schedules \
    --cluster-identifier mycluster \
    --schedule-identifier mysnapshotschedule
```

输出：

```
{
    "SnapshotSchedules": [
        {
            "ScheduleDefinitions": [
                "rate(12 hours)"
            ],
            "ScheduleIdentifier": "mysnapshotschedule",
            "ScheduleDescription": "My schedule description",
            "Tags": [],
            "AssociatedClusterCount": 1,
            "AssociatedClusters": [
                {
                    "ClusterIdentifier": "mycluster",
                    "ScheduleAssociationState": "ACTIVE"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的自动快照计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSnapshotSchedules。

describe-storage

以下代码示例演示了如何使用 describe-storage。

Amazon CLI

描述存储

以下 describe-storage 示例显示了有关该账户的备份存储和临时存储大小的详细信息。

```
aws redshift describe-storage
```

输出：

```
{
    "TotalBackupSizeInMegaBytes": 193149.0,
    "TotalProvisionedStorageInMegaBytes": 655360.0
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的管理快照存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStorage。

describe-table-restore-status

以下代码示例演示了如何使用 describe-table-restore-status。

Amazon CLI

描述来自集群快照的表还原请求的状态

以下 describe-table-restore-status 示例显示了为指定集群发出的表还原请求的详细信息。

```
aws redshift describe-table-restore-status /
    --cluster-identifier mycluster
```

输出：

```
{
    "TableRestoreStatusDetails": [
        {
            "TableRestoreRequestId": "z1116630-0e80-46f4-ba86-bd9670411ebd",
            "Status": "IN_PROGRESS",
            "RequestTime": "2019-12-27T18:22:12.257Z",
            "ClusterIdentifier": "mycluster",
            "SnapshotIdentifier": "mysnapshotid",
            "SourceDatabaseName": "dev",
            "SourceSchemaName": "public",
```

```
            "SourceTableName": "mytable",
            "TargetDatabaseName": "dev",
            "TargetSchemaName": "public",
            "NewTableName": "mytable-clone"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的从快照还原表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTableRestoreStatus。

describe-tags

以下代码示例演示了如何使用 describe-tags。

Amazon CLI

描述标签

以下 describe-tags 示例显示与指定标签名称和值关联的指定集群的资源。

```
aws redshift describe-tags \
    --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \
    --tag-keys clustertagkey \
    --tag-values clustertagvalue
```

输出：

```
{
    "TaggedResources": [
                    {
            "Tag": {
                "Key": "clustertagkey",
                "Value": "clustertagvalue"
            },
            "ResourceName": "arn:aws:redshift:us-
west-2:123456789012:cluster:mycluster",
            "ResourceType": "cluster"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的在 Amazon Redshift 中标记资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

disable-snapshot-copy

以下代码示例演示了如何使用 disable-snapshot-copy。

Amazon CLI

为集群禁用快照复制

以下 disable-snapshot-copy 示例为指定的集群禁用了快照自动复制。

```
aws redshift disable-snapshot-copy \
    --cluster-identifier mycluster
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "available",
        "ClusterAvailabilityStatus": "Available",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-i9b431cd",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
```

```
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-b1fel7t9",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 2,
            "ClusterType": "multi-node"
        },
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
                "ApplyStatus": "in-sync"
            }
        ],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [],
        "ExpectedNextSnapshotScheduleTime": "2019-12-10T04:42:43.390Z",
        "ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
        "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的将快照复制到另一个 Amazon 区域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableSnapshotCopy。

enable-snapshot-copy

以下代码示例演示了如何使用 enable-snapshot-copy。

Amazon CLI

为集群启用快照复制

以下 enable-snapshot-copy 示例为指定的集群启用了快照自动复制。

```
aws redshift enable-snapshot-copy \
    --cluster-identifier mycluster \
    --destination-region us-west-1
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "available",
        "ClusterAvailabilityStatus": "Available",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-f4c731cd",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
```

```
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-b1ael7t9",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 2,
            "ClusterType": "multi-node"
        },
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
        "ClusterSnapshotCopyStatus": {
            "DestinationRegion": "us-west-1",
            "RetentionPeriod": 7,
            "ManualSnapshotRetentionPeriod": -1
        },
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
                "ApplyStatus": "in-sync"
            }
        ],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [],
        "ExpectedNextSnapshotScheduleTime": "2019-12-10T04:42:43.390Z",
        "ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
        "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的将快照复制到另一个 Amazon 区域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableSnapshotCopy。

get-cluster-credentials

以下代码示例演示了如何使用 get-cluster-credentials。

Amazon CLI

获取 Amazon 账户的集群凭证

以下 get-cluster-credentials 示例检索允许访问 Amazon Redshift 数据库的临时凭证。

```
aws redshift get-cluster-credentials \
    --db-user adminuser --db-name dev \
    --cluster-identifier mycluster
```

输出：

```
{
    "DbUser": "IAM:adminuser",
    "DbPassword": "AMAFUyyuros/QjxPTtgzcsuQsqzIasdzJEN04aCtWDzXx1O9d6UmpkBtvEeqFly/
EXAMPLE==",
    "Expiration": "2019-12-10T17:25:05.770Z"
}
```

有关数据 API 的更多信息，请参阅《Amazon Redshift 集群管理指南》中的使用 Amazon Redshift CLI 或 API 生成 IAM 数据库凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetClusterCredentials。

get-reserved-node-exchange-offerings

以下代码示例演示了如何使用 get-reserved-node-exchange-offerings。

Amazon CLI

获取预留节点交换产品

以下 get-reserved-node-exchange-offerings 示例检索与指定的 DC1 预留节点匹配的 DC2 ReservedNodeOfferings 数组。

```
aws redshift get-reserved-node-exchange-offerings \
```

```
    --reserved-node-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE
```

输出：

```
{
    "ReservedNodeOfferings": [
        {
            "ReservedNodeOfferingId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
            "NodeType": "dc2.large",
            "Duration": 31536000,
            "FixedPrice": 0.0,
            "UsagePrice": 0.0,
            "CurrencyCode": "USD",
            "OfferingType": "All Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": 0.0,
                    "RecurringChargeFrequency": "Hourly"
                }
            ],
            "ReservedNodeOfferingType": "Regular"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的使用 Amazon CLI 升级预留节点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetReservedNodeExchangeOfferings。

modify-cluster-iam-roles

以下代码示例演示了如何使用 modify-cluster-iam-roles。

Amazon CLI

修改集群的 IAM 角色

以下 modify-cluster-iam-roles 示例将从指定集群中删除指定的 Amazon IAM 角色。

```
aws redshift modify-cluster-iam-roles \
    --cluster-identifier mycluster \
```

```
    --remove-iam-roles arn:aws:iam::123456789012:role/myRedshiftRole
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "available",
        "ClusterAvailabilityStatus": "Available",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-f9b731sd",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-b2fal7t9",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 2,
            "ClusterType": "multi-node"
        },
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
```

```
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
        "ClusterSnapshotCopyStatus": {
            "DestinationRegion": "us-west-1",
            "RetentionPeriod": 7,
            "ManualSnapshotRetentionPeriod": -1
        },
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [],
        "ExpectedNextSnapshotScheduleTime": "2019-12-11T04:42:55.631Z",
        "ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
        "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的将基于身份的策略（IAM 策略）用于 Amazon Redshift。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterIamRoles。

modify-cluster-maintenance

以下代码示例演示了如何使用 modify-cluster-maintenance。

Amazon CLI

修改集群维护

以下 modify-cluster-maintenance 示例将指定集群的维护推迟 30 天。

```
aws redshift modify-cluster-maintenance \
    --cluster-identifier mycluster \
    --defer-maintenance \
    --defer-maintenance-duration 30
```

输出：


```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "available",
        "ClusterAvailabilityStatus": "Available",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-a1a123ab",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-b1ael7t9",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 2,
            "ClusterType": "multi-node"
        },
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
```

```
        "ClusterSnapshotCopyStatus": {
            "DestinationRegion": "us-west-1",
            "RetentionPeriod": 7,
            "ManualSnapshotRetentionPeriod": -1
        },
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [
            {
                "DeferMaintenanceIdentifier": "dfm-mUdVIfFcT1B4SGhw6fyF",
                "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
                "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
            }
        ],
        "ExpectedNextSnapshotScheduleTime": "2019-12-11T04:42:55.631Z",
        "ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
        "NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的集群维护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterMaintenance。

`modify-cluster-parameter-group`

以下代码示例演示了如何使用 `modify-cluster-parameter-group`。

Amazon CLI

修改参数组中的参数

以下 `modify-cluster-parameter-group` 示例修改了用于工作负载管理的 wlm_json_configuration 参数。它接受来自包含下面所示的 JSON 内容的文件中的参数。

```
aws redshift modify-cluster-parameter-group \
```

```
    --parameter-group-name myclusterparametergroup \
    --parameters file://modify_pg.json
```

的内容modify_pg.json：

```
[
    {
        "ParameterName": "wlm_json_configuration",
        "ParameterValue": "[{\"user_group\":\"example_user_group1\",\"query_group\":
 \"example_query_group1\", \"query_concurrency\":7},{\"query_concurrency\":5}]"
    }
]
```

输出：

```
{
   "ParameterGroupStatus": "Your parameter group has been updated but changes won't
 get applied until you reboot the associated Clusters.",
   "ParameterGroupName": "myclusterparametergroup",
   "ResponseMetadata": {
      "RequestId": "09974cc0-64cd-11e2-bea9-49e0ce183f07"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterParameterGroup。

modify-cluster-snapshot-schedule

以下代码示例演示了如何使用 modify-cluster-snapshot-schedule。

Amazon CLI

修改集群快照计划

以下 modify-cluster-snapshot-schedule 示例将从指定集群中删除指定的快照计划。

```
aws redshift modify-cluster-snapshot-schedule \
    --cluster-identifier mycluster \
    --schedule-identifier mysnapshotschedule \
    --disassociate-schedule
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的自动快照计划。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterSnapshotSchedule。

modify-cluster-snapshot

以下代码示例演示了如何使用 modify-cluster-snapshot。

Amazon CLI

修改集群快照

以下 modify-cluster-snapshot 示例将指定集群快照的手动保留期设置为 10 天。

```
aws redshift modify-cluster-snapshot \
    --snapshot-identifier mycluster-2019-11-06-16-32 \
    --manual-snapshot-retention-period 10
```

输出：

```
{
    "Snapshot": {
        "SnapshotIdentifier": "mycluster-2019-11-06-16-32",
        "ClusterIdentifier": "mycluster",
        "SnapshotCreateTime": "2019-12-07T00:34:05.633Z",
        "Status": "available",
        "Port": 5439,
        "AvailabilityZone": "us-west-2f",
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "MasterUsername": "adminuser",
        "ClusterVersion": "1.0",
        "SnapshotType": "manual",
        "NodeType": "dc2.large",
        "NumberOfNodes": 2,
        "DBName": "dev",
        "VpcId": "vpc-b1ce17t9",
        "Encrypted": false,
        "EncryptedWithHSM": false,
        "OwnerAccount": "123456789012",
        "TotalBackupSizeInMegaBytes": 64384.0,
        "ActualIncrementalBackupSizeInMegaBytes": 24.0,
        "BackupProgressInMegaBytes": 24.0,
        "CurrentBackupRateInMegaBytesPerSecond": 13.0011,
```

```
        "EstimatedSecondsToCompletion": 0,
        "ElapsedTimeInSeconds": 1,
        "Tags": [
            {
                "Key": "mytagkey",
                "Value": "mytagvalue"
            }
        ],
        "EnhancedVpcRouting": false,
        "MaintenanceTrackName": "current",
        "ManualSnapshotRetentionPeriod": 10,
        "ManualSnapshotRemainingDays": 6,
        "SnapshotRetentionStartTime": "2019-12-07T00:34:07.479Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterSnapshot。

`modify-cluster-subnet-group`

以下代码示例演示了如何使用 `modify-cluster-subnet-group`。

Amazon CLI

修改集群子网组中的子网 此示例说明如何修改缓存子网组中的子网列表。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift modify-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
 --subnet-ids subnet-763fdd1 subnet-ac830e9
```

结果：

```
{
   "ClusterSubnetGroup":
   {
      "Subnets": [
         {
            "SubnetStatus": "Active",
            "SubnetIdentifier": "subnet-763fdd1c",
            "SubnetAvailabilityZone":
```

```
                { "Name": "us-east-1a" }
         },
         {
            "SubnetStatus": "Active",
            "SubnetIdentifier": "subnet-ac830e9",
            "SubnetAvailabilityZone":
               { "Name": "us-east-1b" }
         } ],
      "VpcId": "vpc-7e3fdd14",
      "SubnetGroupStatus": "Complete",
      "Description": "My subnet group",
      "ClusterSubnetGroupName": "mysubnetgroup"
   },
   "ResponseMetadata": {
      "RequestId": "8da93e89-8372-f936-93a8-873918938197a"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyClusterSubnetGroup。

`modify-cluster`

以下代码示例演示了如何使用 `modify-cluster`。

Amazon CLI

将安全组与集群关联 此示例演示了如何将集群安全组与指定的集群相关联。命令：

```
aws redshift modify-cluster --cluster-identifier mycluster --cluster-security-groups
 mysecuritygroup
```

修改集群的维护时段 此示例演示了如何将集群的每周首选维护时段更改为最少四小时的时段，从周日晚上 11:15 开始，到周一凌晨 3:15 结束。命令：

```
aws redshift modify-cluster --cluster-identifier mycluster --preferred-maintenance-
window Sun:23:15-Mon:03:15
```

更改集群的主密码 此示例演示了如何更改集群的主密码。命令：

```
aws redshift modify-cluster --cluster-identifier mycluster --master-user-password
 A1b2c3d4
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyCluster。

modify-event-subscription

以下代码示例演示了如何使用 modify-event-subscription。

Amazon CLI

修改事件订阅

以下 modify-event-subscription 示例禁用了指定的事件通知订阅。

```
aws redshift modify-event-subscription \
    --subscription-name mysubscription \
    --no-enabled
```

输出：

```
{
    "EventSubscription": {
        "CustomerAwsId": "123456789012",
        "CustSubscriptionId": "mysubscription",
        "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNStopic",
        "Status": "active",
        "SubscriptionCreationTime": "2019-12-09T21:50:21.332Z",
        "SourceIdsList": [],
        "EventCategoriesList": [
            "management"
        ],
        "Severity": "ERROR",
        "Enabled": false,
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的订阅 Amazon Redshift 事件通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyEventSubscription。

modify-scheduled-action

以下代码示例演示了如何使用 modify-scheduled-action。

Amazon CLI

修改计划操作

以下 modify-scheduled-action 示例为指定的现有计划操作添加了描述。

```
aws redshift modify-scheduled-action \
    --scheduled-action-name myscheduledaction \
    --scheduled-action-description "My scheduled action"
```

输出：

```
{
    "ScheduledActionName": "myscheduledaction",
    "TargetAction": {
        "ResizeCluster": {
            "ClusterIdentifier": "mycluster",
            "NumberOfNodes": 2,
            "Classic": false
        }
    },
    "Schedule": "at(2019-12-25T00:00:00)",
    "IamRole": "arn:aws:iam::123456789012:role/myRedshiftRole",
    "ScheduledActionDescription": "My scheduled action",
    "State": "ACTIVE",
    "NextInvocations": [
        "2019-12-25T00:00:00Z"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyScheduledAction。

modify-snapshot-copy-retention-period

以下代码示例演示了如何使用 modify-snapshot-copy-retention-period。

Amazon CLI

修改快照复制保留期

以下 modify-snapshot-copy-retention-period 示例修改了从源 Amazon 区域复制指定集群的快照后在目标 Amazon 区域中保留的天数。

```
aws redshift modify-snapshot-copy-retention-period \
    --cluster-identifier mycluster \
    --retention-period 15
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "available",
        "ClusterAvailabilityStatus": "Available",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-a1a123ab",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-b1fet7t9",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 2,
            "ClusterType": "multi-node"
        },
```

```
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
        "ClusterSnapshotCopyStatus": {
            "DestinationRegion": "us-west-1",
            "RetentionPeriod": 15,
            "ManualSnapshotRetentionPeriod": -1
        },
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [
            {
                "DeferMaintenanceIdentifier": "dfm-mUdVSfDcT1F4SGhw6fyF",
                "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
                "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
            }
        ],
        "NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的快照计划格式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySnapshotCopyRetentionPeriod。

`modify-snapshot-schedule`

以下代码示例演示了如何使用 `modify-snapshot-schedule`。

Amazon CLI

修改快照计划

以下 modify-snapshot-schedule 示例将指定快照计划的速率修改为每 10 小时一次。

```
aws redshift modify-snapshot-schedule \
    --schedule-identifier mysnapshotschedule \
    --schedule-definitions "rate(10 hours)"
```

输出：

```
{
    "ScheduleDefinitions": [
        "rate(10 hours)"
    ],
    "ScheduleIdentifier": "mysnapshotschedule",
    "ScheduleDescription": "My schedule description",
    "Tags": []
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的快照计划格式。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifySnapshotSchedule。

purchase-reserved-node-offering

以下代码示例演示了如何使用 purchase-reserved-node-offering。

Amazon CLI

购买预留节点 此示例说明如何购买预留节点产品。可通过调用 describe-reserved-node-offerings 来获取 reserved-node-offering-id。命令：

```
aws redshift purchase-reserved-node-offering --reserved-node-offering-id ceb6a579-cf4c-4343-be8b-d832c45ab51c
```

结果：

```
{
   "ReservedNode": {
      "OfferingType": "Heavy Utilization",
      "FixedPrice": "",
      "NodeType": "dw.hs1.xlarge",
      "ReservedNodeId": "1ba8e2e3-bc01-4d65-b35d-a4a3e931547e",
```

```
      "UsagePrice": "",
      "RecurringCharges": [
         {
            "RecurringChargeAmount": "",
            "RecurringChargeFrequency": "Hourly"
         }
      ],
      "NodeCount": 1,
      "State": "payment-pending",
      "StartTime": "2013-02-13T17:08:39.051Z",
      "Duration": 31536000,
      "ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
   },
   "ResponseMetadata": {
      "RequestId": "01bda7bf-7600-11e2-b605-2568d7396e7f"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurchaseReservedNodeOffering。

reboot-cluster

以下代码示例演示了如何使用 reboot-cluster。

Amazon CLI

重启集群 此示例重新启动了集群。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift reboot-cluster --cluster-identifier mycluster
```

结果：

```
{
   "Cluster": {
      "NodeType": "dw.hs1.xlarge",
      "Endpoint": {
         "Port": 5439,
         "Address": "mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com"
      },
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
```

```
      "ClusterParameterGroups": [
         {
         "ParameterApplyStatus": "in-sync",
         "ParameterGroupName": "default.redshift-1.0"
         }
      ],
      "ClusterSecurityGroups": [
         {
         "Status": "active",
         "ClusterSecurityGroupName": "default"
         }
      ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "AvailabilityZone": "us-east-1a",
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
      "PreferredMaintenanceWindow": "sun:23:15-mon:03:15",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "rebooting",
      "ClusterIdentifier": "mycluster",
      "DBName": "dev",
      "NumberOfNodes": 2,
      "PendingModifiedValues": {}
   },
   "ResponseMetadata": {
      "RequestId": "61c8b564-64e8-11e2-8f7d-3b939af52818"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootCluster。

reset-cluster-parameter-group

以下代码示例演示了如何使用 `reset-cluster-parameter-group`。

Amazon CLI

重置参数组中的参数 此示例说明如何重置参数组中的所有参数。命令：

```
aws redshift reset-cluster-parameter-group --parameter-group-name
 myclusterparametergroup --reset-all-parameters
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetClusterParameterGroup。

resize-cluster

以下代码示例演示了如何使用 resize-cluster。

Amazon CLI

调整集群大小

下面的 resize-cluster 示例对指定集群进行了大小调整。

```
aws redshift resize-cluster \
    --cluster-identifier mycluster \
    --cluster-type multi-node \
    --node-type dc2.large \
    --number-of-nodes 6 \
    --classic
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "resizing",
        "ClusterAvailabilityStatus": "Modifying",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
        "AutomatedSnapshotRetentionPeriod": 3,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-a1a123ab",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
```

```
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-a1abc1a1",
        "AvailabilityZone": "us-west-2f",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {
            "NodeType": "dc2.large",
            "NumberOfNodes": 6,
            "ClusterType": "multi-node"
        },
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 4,
        "PubliclyAccessible": false,
        "Encrypted": false,
        "ClusterSnapshotCopyStatus": {
            "DestinationRegion": "us-west-1",
            "RetentionPeriod": 15,
            "ManualSnapshotRetentionPeriod": -1
        },
        "Tags": [
            {
                "Key": "mytags",
                "Value": "tag1"
            }
        ],
        "EnhancedVpcRouting": false,
        "IamRoles": [],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [
            {
                "DeferMaintenanceIdentifier": "dfm-mUdVCfDcT1B4SGhw6fyF",
                "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
                "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
            }
        ],
        "NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z",
        "ResizeInfo": {
            "ResizeType": "ClassicResize",
            "AllowCancelResize": true
        }
    }
```

```
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的调整集群大小。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResizeCluster。

`restore-from-cluster-snapshot`

以下代码示例演示了如何使用 `restore-from-cluster-snapshot`。

Amazon CLI

从快照还原集群 此示例将从快照还原集群。命令：

```
aws redshift restore-from-cluster-snapshot --cluster-identifier mycluster-clone --
snapshot-identifier my-snapshot-id
```

结果：

```
{
   "Cluster": {
      "NodeType": "dw.hs1.xlarge",
      "ClusterVersion": "1.0",
      "PubliclyAccessible": "true",
      "MasterUsername": "adminuser",
      "ClusterParameterGroups": [
         {
         "ParameterApplyStatus": "in-sync",
         "ParameterGroupName": "default.redshift-1.0"
         }
      ],
      "ClusterSecurityGroups": [
         {
         "Status": "active",
         "ClusterSecurityGroupName": "default"
         }
      ],
      "AllowVersionUpgrade": true,
      "VpcSecurityGroups": \[],
      "PreferredMaintenanceWindow": "sun:23:15-mon:03:15",
      "AutomatedSnapshotRetentionPeriod": 1,
      "ClusterStatus": "creating",
      "ClusterIdentifier": "mycluster-clone",
```

```
        "DBName": "dev",
        "NumberOfNodes": 2,
        "PendingModifiedValues": {}
    },
    "ResponseMetadata": {
        "RequestId": "77fd512b-64e3-11e2-8f5b-e90bd6c77476"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreFromClusterSnapshot。

restore-table-from-cluster-snapshot

以下代码示例演示了如何使用 restore-table-from-cluster-snapshot。

Amazon CLI

从集群快照还原表

以下 restore-table-from-cluster-snapshot 示例根据指定集群快照中的指定表创建了一个新表。

```
aws redshift restore-table-from-cluster-snapshot /
    --cluster-identifier mycluster /
    --snapshot-identifier mycluster-2019-11-19-16-17 /
    --source-database-name dev /
    --source-schema-name public /
    --source-table-name mytable /
    --target-database-name dev /
    --target-schema-name public /
    --new-table-name mytable-clone
```

输出：

```
{
    "TableRestoreStatus": {
        "TableRestoreRequestId": "a123a12b-abc1-1a1a-a123-a1234ab12345",
        "Status": "PENDING",
        "RequestTime": "2019-12-20T00:20:16.402Z",
        "ClusterIdentifier": "mycluster",
        "SnapshotIdentifier": "mycluster-2019-11-19-16-17",
        "SourceDatabaseName": "dev",
```

```
        "SourceSchemaName": "public",
        "SourceTableName": "mytable",
        "TargetDatabaseName": "dev",
        "TargetSchemaName": "public",
        "NewTableName": "mytable-clone"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的从快照还原表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreTableFromClusterSnapshot。

`revoke-cluster-security-group-ingress`

以下代码示例演示了如何使用 `revoke-cluster-security-group-ingress`。

Amazon CLI

从 EC2 安全组撤消权限 此示例撤消了对指定的 Amazon EC2 安全组的访问权限。命令：

```
aws redshift revoke-cluster-security-group-ingress --cluster-security-group-name
 mysecuritygroup --ec2-security-group-name myec2securitygroup --ec2-security-group-
owner-id 123445677890
```

撤销访问 CIDR 范围 此示例撤消了对 CIDR 范围的访问。命令：

```
aws redshift revoke-cluster-security-group-ingress --cluster-security-group-name
 mysecuritygroup --cidrip 192.168.100.100/32
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeClusterSecurityGroupIngress。

`revoke-snapshot-access`

以下代码示例演示了如何使用 `revoke-snapshot-access`。

Amazon CLI

撤消 Amazon 账户还原快照的授权 此示例撤消了 Amazon 账户 444455556666 还原快照 my-snapshot-id 的授权。默认情况下，输出采用 JSON 格式。命令：

```
aws redshift revoke-snapshot-access --snapshot-id my-snapshot-id --account-with-
restore-access 444455556666
```

结果：

```
{
   "Snapshot": {
      "Status": "available",
      "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
      "EstimatedSecondsToCompletion": 0,
      "AvailabilityZone": "us-east-1a",
      "ClusterVersion": "1.0",
      "MasterUsername": "adminuser",
      "Encrypted": false,
      "OwnerAccount": "111122223333",
      "BackupProgressInMegabytes": 11.0,
      "ElapsedTimeInSeconds": 0,
      "DBName": "dev",
      "CurrentBackupRateInMegabytesPerSecond: 0.1534,
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
      "ActualIncrementalBackupSizeInMegabytes"; 11.0,
      "SnapshotType": "manual",
      "NodeType": "dw.hs1.xlarge",
      "ClusterIdentifier": "mycluster",
      "TotalBackupSizeInMegabytes": 20.0,
      "Port": 5439,
      "NumberOfNodes": 2,
      "SnapshotIdentifier": "my-snapshot-id"
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RevokeSnapshotAccess。

rotate-encryption-key

以下代码示例演示了如何使用 rotate-encryption-key。

Amazon CLI

为集群轮换加密密钥

以下 rotate-encryption-key 示例轮换所指定集群的加密密钥。

```
aws redshift rotate-encryption-key \
    --cluster-identifier mycluster
```

输出：

```
{
    "Cluster": {
        "ClusterIdentifier": "mycluster",
        "NodeType": "dc2.large",
        "ClusterStatus": "rotating-keys",
        "ClusterAvailabilityStatus": "Modifying",
        "MasterUsername": "adminuser",
        "DBName": "dev",
        "Endpoint": {
            "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
            "Port": 5439
        },
        "ClusterCreateTime": "2019-12-10T19:25:45.886Z",
        "AutomatedSnapshotRetentionPeriod": 30,
        "ManualSnapshotRetentionPeriod": -1,
        "ClusterSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sh-a1a123ab",
                "Status": "active"
            }
        ],
        "ClusterParameterGroups": [
            {
                "ParameterGroupName": "default.redshift-1.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "ClusterSubnetGroupName": "default",
        "VpcId": "vpc-a1abc1a1",
        "AvailabilityZone": "us-west-2a",
        "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
        "PendingModifiedValues": {},
        "ClusterVersion": "1.0",
        "AllowVersionUpgrade": true,
        "NumberOfNodes": 2,
        "PubliclyAccessible": false,
        "Encrypted": true,
```

```
        "Tags": [],
        "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
bPxRfih3yCo8nvbEXAMPLEKEY",
        "EnhancedVpcRouting": false,
        "IamRoles": [
            {
                "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
                "ApplyStatus": "in-sync"
            }
        ],
        "MaintenanceTrackName": "current",
        "DeferredMaintenanceWindows": [],
        "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
    }
}
```

有关更多信息，请参阅《Amazon Redshift 集群管理指南》中的 Amazon Redshift 数据库加密。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RotateEncryptionKey。

使用 Amazon CLI 的 Amazon Rekognition 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Rekognition 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`compare-faces`

以下代码示例演示了如何使用 `compare-faces`。

有关更多信息，请参阅比较图像中的人脸。

Amazon CLI

比较两张图像中的人脸

以下 compare-faces 命令将比较存储在 Amazon S3 存储桶中的两张图像中的人脸。

```
aws rekognition compare-faces \
    --source-image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"source.jpg"}}' \
    --target-image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"target.jpg"}}'
```

输出：

```
{
    "UnmatchedFaces": [],
    "FaceMatches": [
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.12368916720151901,
                    "Top": 0.16007372736930847,
                    "Left": 0.5901257991790771,
                    "Height": 0.25140416622161865
                },
                "Confidence": 100.0,
                "Pose": {
                    "Yaw": -3.7351467609405518,
                    "Roll": -0.10309021919965744,
                    "Pitch": 0.8637830018997192
                },
                "Quality": {
                    "Sharpness": 95.51618957519531,
                    "Brightness": 65.29893493652344
                },
                "Landmarks": [
                    {
                        "Y": 0.26721030473709106,
                        "X": 0.6204193830490112,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.26831310987472534,
                        "X": 0.6776827573776245,
                        "Type": "eyeRight"
```

```
                    },
                    {
                        "Y": 0.3514654338359833,
                        "X": 0.6241428852081299,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.35258132219314575,
                        "X": 0.6713621020317078,
                        "Type": "mouthRight"
                    },
                    {
                        "Y": 0.3140771687030792,
                        "X": 0.6428444981575012,
                        "Type": "nose"
                    }
                ]
            },
            "Similarity": 100.0
        }
    ],
    "SourceImageFace": {
        "BoundingBox": {
            "Width": 0.12368916720151901,
            "Top": 0.16007372736930847,
            "Left": 0.5901257991790771,
            "Height": 0.25140416622161865
        },
        "Confidence": 100.0
    }
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的比较图像中的人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompareFaces。

create-collection

以下代码示例演示了如何使用 create-collection。

有关更多信息，请参阅创建集合。

Amazon CLI

创建集合

以下 create-collection 命令创建具有指定名称的集合。

```
aws rekognition create-collection \
    --collection-id "MyCollection"
```

输出：

```
{
    "CollectionArn": "aws:rekognition:us-west-2:123456789012:collection/
MyCollection",
    "FaceModelVersion": "4.0",
    "StatusCode": 200
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的创建集合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCollection。

create-stream-processor

以下代码示例演示了如何使用 create-stream-processor。

Amazon CLI

创建新的流处理器

以下 create-stream-processor 示例使用指定配置创建了一个新的流处理器。

```
aws rekognition create-stream-processor --name my-stream-processor\
    --input '{"KinesisVideoStream":{"Arn":"arn:aws:kinesisvideo:us-
west-2:123456789012:stream/macwebcam/1530559711205"}}'\
    --stream-processor-output '{"KinesisDataStream":{"Arn":"arn:aws:kinesis:us-
west-2:123456789012:stream/AmazonRekognitionRekStream"}}'\
    --role-arn arn:aws:iam::123456789012:role/AmazonRekognitionDetect\
    --settings '{"FaceSearch":
{"CollectionId":"MyCollection","FaceMatchThreshold":85.5}}'
```

输出：

```
{
    "StreamProcessorArn": "arn:aws:rekognition:us-
west-2:123456789012:streamprocessor/my-stream-processor"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateStreamProcessor。

`delete-collection`

以下代码示例演示了如何使用 `delete-collection`。

有关更多信息，请参阅删除集合。

Amazon CLI

删除集合

以下 `delete-collection` 命令将删除指定的集合。

```
aws rekognition delete-collection \
    --collection-id MyCollection
```

输出：

```
{
    "StatusCode": 200
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的删除集合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCollection。

`delete-faces`

以下代码示例演示了如何使用 `delete-faces`。

有关更多信息，请参阅从集中删除人脸。

Amazon CLI

从集合中删除人脸

以下 delete-faces 命令将从集合中删除指定的人脸。

```
aws rekognition delete-faces \
    --collection-id MyCollection
    --face-ids '["0040279c-0178-436e-b70a-e61b074e96b0"]'
```

输出：

```
{
    "DeletedFaces": [
        "0040279c-0178-436e-b70a-e61b074e96b0"
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的从集合中删除人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFaces。

delete-stream-processor

以下代码示例演示了如何使用 delete-stream-processor。

Amazon CLI

删除流处理器

以下 delete-stream-processor 命令可删除指定的流处理器。

```
aws rekognition delete-stream-processor \
    --name my-stream-processor
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteStreamProcessor。

describe-collection

以下代码示例演示了如何使用 describe-collection。

有关更多信息，请参阅描述集合。

Amazon CLI

描述集合

以下 describe-collection 示例显示有关指定集合的详细信息。

```
aws rekognition describe-collection \
    --collection-id MyCollection
```

输出：

```
{
    "FaceCount": 200,
    "CreationTimestamp": 1569444828.274,
    "CollectionARN": "arn:aws:rekognition:us-west-2:123456789012:collection/
MyCollection",
    "FaceModelVersion": "4.0"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的描述集合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCollection。

describe-stream-processor

以下代码示例演示了如何使用 describe-stream-processor。

Amazon CLI

获取有关流处理器的信息

以下 describe-stream-processor 命令可显示有关指定流处理器的详细信息。

```
aws rekognition describe-stream-processor \
    --name my-stream-processor
```

输出：

```
{
    "Status": "STOPPED",
    "Name": "my-stream-processor",
    "LastUpdateTimestamp": 1532449292.712,
```

```
    "Settings": {
        "FaceSearch": {
            "FaceMatchThreshold": 80.0,
            "CollectionId": "my-collection"
        }
    },
    "RoleArn": "arn:aws:iam::123456789012:role/AmazonRekognitionDetectStream",
    "StreamProcessorArn": "arn:aws:rekognition:us-
west-2:123456789012:streamprocessor/my-stream-processpr",
    "Output": {
        "KinesisDataStream": {
            "Arn": "arn:aws:kinesis:us-west-2:123456789012:stream/
AmazonRekognitionRekStream"
        }
    },
    "Input": {
        "KinesisVideoStream": {
            "Arn": "arn:aws:kinesisvideo:us-west-2:123456789012:stream/
macwebcam/123456789012"
        }
    },
    "CreationTimestamp": 1532449292.712
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStreamProcessor。

`detect-faces`

以下代码示例演示了如何使用 `detect-faces`。

有关更多信息，请参阅检测图像中的人脸。

Amazon CLI

检测图像中的人脸

以下 `detect-faces` 命令将检测存储在 Amazon S3 存储桶中的指定图像中的人脸。

```
aws rekognition detect-faces \
    --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"MyFriend.jpg"}}' \
    --attributes "ALL"
```

输出：


```
{
    "FaceDetails": [
        {
            "Confidence": 100.0,
            "Eyeglasses": {
                "Confidence": 98.91107940673828,
                "Value": false
            },
            "Sunglasses": {
                "Confidence": 99.7966537475586,
                "Value": false
            },
            "Gender": {
                "Confidence": 99.56611633300781,
                "Value": "Male"
            },
            "Landmarks": [
                {
                    "Y": 0.26721030473709106,
                    "X": 0.6204193830490112,
                    "Type": "eyeLeft"
                },
                {
                    "Y": 0.26831310987472534,
                    "X": 0.6776827573776245,
                    "Type": "eyeRight"
                },
                {
                    "Y": 0.3514654338359833,
                    "X": 0.6241428852081299,
                    "Type": "mouthLeft"
                },
                {
                    "Y": 0.35258132219314575,
                    "X": 0.6713621020317078,
                    "Type": "mouthRight"
                },
                {
                    "Y": 0.3140771687030792,
                    "X": 0.6428444981575012,
                    "Type": "nose"
                },
```

```
                {
                    "Y": 0.24662546813488007,
                    "X": 0.6001564860343933,
                    "Type": "leftEyeBrowLeft"
                },
                {
                    "Y": 0.24326619505882263,
                    "X": 0.6303644776344299,
                    "Type": "leftEyeBrowRight"
                },
                {
                    "Y": 0.23818562924861908,
                    "X": 0.6146903038024902,
                    "Type": "leftEyeBrowUp"
                },
                {
                    "Y": 0.24373626708984375,
                    "X": 0.6640064716339111,
                    "Type": "rightEyeBrowLeft"
                },
                {
                    "Y": 0.24877218902111053,
                    "X": 0.7025929093360901,
                    "Type": "rightEyeBrowRight"
                },
                {
                    "Y": 0.23938551545143127,
                    "X": 0.6823262572288513,
                    "Type": "rightEyeBrowUp"
                },
                {
                    "Y": 0.265746533870697,
                    "X": 0.6112898588180542,
                    "Type": "leftEyeLeft"
                },
                {
                    "Y": 0.2676128149032593,
                    "X": 0.6317071914672852,
                    "Type": "leftEyeRight"
                },
                {
                    "Y": 0.262735515832901,
                    "X": 0.6201658248901367,
                    "Type": "leftEyeUp"
```

```
                },
                {
                    "Y": 0.27025148272514343,
                    "X": 0.6206279993057251,
                    "Type": "leftEyeDown"
                },
                {
                    "Y": 0.268223375082016,
                    "X": 0.6658390760421753,
                    "Type": "rightEyeLeft"
                },
                {
                    "Y": 0.2672517001628876,
                    "X": 0.687832236289978,
                    "Type": "rightEyeRight"
                },
                {
                    "Y": 0.26383838057518005,
                    "X": 0.6769183874130249,
                    "Type": "rightEyeUp"
                },
                {
                    "Y": 0.27138751745224,
                    "X": 0.676596462726593,
                    "Type": "rightEyeDown"
                },
                {
                    "Y": 0.32283174991607666,
                    "X": 0.6350004076957703,
                    "Type": "noseLeft"
                },
                {
                    "Y": 0.3219289481639862,
                    "X": 0.6567046642303467,
                    "Type": "noseRight"
                },
                {
                    "Y": 0.3420318365097046,
                    "X": 0.6450609564781189,
                    "Type": "mouthUp"
                },
                {
                    "Y": 0.3664324879646301,
                    "X": 0.6455618143081665,
```

```
                    "Type": "mouthDown"
                },
                {
                    "Y": 0.26721030473709106,
                    "X": 0.6204193830490112,
                    "Type": "leftPupil"
                },
                {
                    "Y": 0.26831310987472534,
                    "X": 0.6776827573776245,
                    "Type": "rightPupil"
                },
                {
                    "Y": 0.26343393325805664,
                    "X": 0.5946047306060791,
                    "Type": "upperJawlineLeft"
                },
                {
                    "Y": 0.3543180525302887,
                    "X": 0.6044883728027344,
                    "Type": "midJawlineLeft"
                },
                {
                    "Y": 0.4084877669811249,
                    "X": 0.6477024555206299,
                    "Type": "chinBottom"
                },
                {
                    "Y": 0.3562754988670349,
                    "X": 0.707981526851654,
                    "Type": "midJawlineRight"
                },
                {
                    "Y": 0.26580461859703064,
                    "X": 0.7234612107276917,
                    "Type": "upperJawlineRight"
                }
            ],
            "Pose": {
                "Yaw": -3.7351467609405518,
                "Roll": -0.10309021919965744,
                "Pitch": 0.8637830018997192
            },
            "Emotions": [
```

```
                {
                    "Confidence": 8.74203109741211,
                    "Type": "SURPRISED"
                },
                {
                    "Confidence": 2.501944065093994,
                    "Type": "ANGRY"
                },
                {
                    "Confidence": 0.7378743290901184,
                    "Type": "DISGUSTED"
                },
                {
                    "Confidence": 3.5296201705932617,
                    "Type": "HAPPY"
                },
                {
                    "Confidence": 1.7162904739379883,
                    "Type": "SAD"
                },
                {
                    "Confidence": 9.518536567687988,
                    "Type": "CONFUSED"
                },
                {
                    "Confidence": 0.45474427938461304,
                    "Type": "FEAR"
                },
                {
                    "Confidence": 72.79895782470703,
                    "Type": "CALM"
                }
            ],
            "AgeRange": {
                "High": 48,
                "Low": 32
            },
            "EyesOpen": {
                "Confidence": 98.93987274169922,
                "Value": true
            },
            "BoundingBox": {
                "Width": 0.12368916720151901,
                "Top": 0.16007372736930847,
```

```
                "Left": 0.5901257991790771,
                "Height": 0.25140416622161865
            },
            "Smile": {
                "Confidence": 93.4493179321289,
                "Value": false
            },
            "MouthOpen": {
                "Confidence": 90.53053283691406,
                "Value": false
            },
            "Quality": {
                "Sharpness": 95.51618957519531,
                "Brightness": 65.29893493652344
            },
            "Mustache": {
                "Confidence": 89.85221099853516,
                "Value": false
            },
            "Beard": {
                "Confidence": 86.1991195678711,
                "Value": true
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测图像中的人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectFaces。

detect-labels

以下代码示例演示了如何使用 detect-labels。

有关更多信息，请参阅检测图像中的标签。

Amazon CLI

检测图像中的标签

以下 detect-labels 示例将检测存储在 Amazon S3 存储桶中的图像中的场景和对象。

```
aws rekognition detect-labels \
```

```
    --image '{"S3Object":{"Bucket":"bucket","Name":"image"}}'
```

输出：

```
{
    "Labels": [
        {
            "Instances": [],
            "Confidence": 99.15271759033203,
            "Parents": [
                {
                    "Name": "Vehicle"
                },
                {
                    "Name": "Transportation"
                }
            ],
            "Name": "Automobile"
        },
        {
            "Instances": [],
            "Confidence": 99.15271759033203,
            "Parents": [
                {
                    "Name": "Transportation"
                }
            ],
            "Name": "Vehicle"
        },
        {
            "Instances": [],
            "Confidence": 99.15271759033203,
            "Parents": [],
            "Name": "Transportation"
        },
        {
            "Instances": [
                {
                    "BoundingBox": {
                        "Width": 0.10616336017847061,
                        "Top": 0.5039216876029968,
                        "Left": 0.0037978808395564556,
                        "Height": 0.18528179824352264
```

```
            },
            "Confidence": 99.15271759033203
        },
        {
            "BoundingBox": {
                "Width": 0.2429988533258438,
                "Top": 0.5251884460449219,
                "Left": 0.7309805154800415,
                "Height": 0.21577216684818268
            },
            "Confidence": 99.1286392211914
        },
        {
            "BoundingBox": {
                "Width": 0.14233611524105072,
                "Top": 0.5333095788955688,
                "Left": 0.6494812965393066,
                "Height": 0.15528248250484467
            },
            "Confidence": 98.48368072509766
        },
        {
            "BoundingBox": {
                "Width": 0.11086395382881165,
                "Top": 0.5354844927787781,
                "Left": 0.10355594009160995,
                "Height": 0.10271988064050674
            },
            "Confidence": 96.45606231689453
        },
        {
            "BoundingBox": {
                "Width": 0.06254628300666809,
                "Top": 0.5573825240135193,
                "Left": 0.46083059906959534,
                "Height": 0.053911514580249786
            },
            "Confidence": 93.65448760986328
        },
        {
            "BoundingBox": {
                "Width": 0.10105438530445099,
                "Top": 0.534368634223938,
                "Left": 0.5743985772132874,
```

```
                "Height": 0.12226245552301407
            },
            "Confidence": 93.06217193603516
        },
        {
            "BoundingBox": {
                "Width": 0.056389667093753815,
                "Top": 0.5235804319381714,
                "Left": 0.9427769780158997,
                "Height": 0.17163699865341187
            },
            "Confidence": 92.6864013671875
        },
        {
            "BoundingBox": {
                "Width": 0.06003860384225845,
                "Top": 0.5441341400146484,
                "Left": 0.22409997880458832,
                "Height": 0.06737709045410156
            },
            "Confidence": 90.4227066040039
        },
        {
            "BoundingBox": {
                "Width": 0.02848697081208229,
                "Top": 0.5107086896896362,
                "Left": 0,
                "Height": 0.19150497019290924
            },
            "Confidence": 86.65286254882812
        },
        {
            "BoundingBox": {
                "Width": 0.04067881405353546,
                "Top": 0.5566273927688599,
                "Left": 0.316415935754776,
                "Height": 0.03428703173995018
            },
            "Confidence": 85.36471557617188
        },
        {
            "BoundingBox": {
                "Width": 0.043411049991846085,
                "Top": 0.5394920110702515,
```

```
                    "Left": 0.18293385207653046,
                    "Height": 0.0893595889210701
                },
                "Confidence": 82.21705627441406
            },
            {
                "BoundingBox": {
                    "Width": 0.031183116137981415,
                    "Top": 0.5579366683959961,
                    "Left": 0.2853088080883026,
                    "Height": 0.03989990055561066
                },
                "Confidence": 81.0157470703125
            },
            {
                "BoundingBox": {
                    "Width": 0.031113790348172188,
                    "Top": 0.5504819750785828,
                    "Left": 0.2580395042896271,
                    "Height": 0.056484755128622055
                },
                "Confidence": 56.13441467285156
            },
            {
                "BoundingBox": {
                    "Width": 0.08586374670267105,
                    "Top": 0.5438792705535889,
                    "Left": 0.5128012895584106,
                    "Height": 0.08550430089235306
                },
                "Confidence": 52.37760925292969
            }
        ],
        "Confidence": 99.15271759033203,
        "Parents": [
            {
                "Name": "Vehicle"
            },
            {
                "Name": "Transportation"
            }
        ],
        "Name": "Car"
    },
```

```
        {
            "Instances": [],
            "Confidence": 98.9914321899414,
            "Parents": [],
            "Name": "Human"
        },
        {
            "Instances": [
                {
                    "BoundingBox": {
                        "Width": 0.19360728561878204,
                        "Top": 0.35072067379951477,
                        "Left": 0.43734854459762573,
                        "Height": 0.2742200493812561
                    },
                    "Confidence": 98.9914321899414
                },
                {
                    "BoundingBox": {
                        "Width": 0.03801717236638069,
                        "Top": 0.5010883808135986,
                        "Left": 0.9155802130699158,
                        "Height": 0.06597328186035156
                    },
                    "Confidence": 85.02790832519531
                }
            ],
            "Confidence": 98.9914321899414,
            "Parents": [],
            "Name": "Person"
        },
        {
            "Instances": [],
            "Confidence": 93.24951934814453,
            "Parents": [],
            "Name": "Machine"
        },
        {
            "Instances": [
                {
                    "BoundingBox": {
                        "Width": 0.03561960905790329,
                        "Top": 0.6468243598937988,
                        "Left": 0.7850857377052307,
```

```
                "Height": 0.08878646790981293
            },
            "Confidence": 93.24951934814453
        },
        {
            "BoundingBox": {
                "Width": 0.02217046171426773,
                "Top": 0.6149078607559204,
                "Left": 0.04757237061858177,
                "Height": 0.07136218994855881
            },
            "Confidence": 91.5025863647461
        },
        {
            "BoundingBox": {
                "Width": 0.016197510063648224,
                "Top": 0.6274210214614868,
                "Left": 0.6472989320755005,
                "Height": 0.04955997318029404
            },
            "Confidence": 85.14686584472656
        },
        {
            "BoundingBox": {
                "Width": 0.020207518711686134,
                "Top": 0.6348286867141724,
                "Left": 0.7295016646385193,
                "Height": 0.07059963047504425
            },
            "Confidence": 83.34547424316406
        },
        {
            "BoundingBox": {
                "Width": 0.020280985161662102,
                "Top": 0.6171894669532776,
                "Left": 0.08744934946298599,
                "Height": 0.05297485366463661
            },
            "Confidence": 79.9981460571289
        },
        {
            "BoundingBox": {
                "Width": 0.018318990245461464,
                "Top": 0.623889148235321,
```

```
                    "Left": 0.6836880445480347,
                    "Height": 0.06730121374130249
                },
                "Confidence": 78.87144470214844
            },
            {
                "BoundingBox": {
                    "Width": 0.021310249343514442,
                    "Top": 0.6167286038398743,
                    "Left": 0.004064912907779217,
                    "Height": 0.08317798376083374
                },
                "Confidence": 75.89361572265625
            },
            {
                "BoundingBox": {
                    "Width": 0.03604431077837944,
                    "Top": 0.7030032277107239,
                    "Left": 0.9254803657531738,
                    "Height": 0.04569442570209503
                },
                "Confidence": 64.402587890625
            },
            {
                "BoundingBox": {
                    "Width": 0.009834849275648594,
                    "Top": 0.5821820497512817,
                    "Left": 0.28094568848609924,
                    "Height": 0.01964157074689865
                },
                "Confidence": 62.79907989501953
            },
            {
                "BoundingBox": {
                    "Width": 0.01475677452981472,
                    "Top": 0.6137543320655823,
                    "Left": 0.5950819253921509,
                    "Height": 0.039063986390829086
                },
                "Confidence": 59.40483474731445
            }
        ],
        "Confidence": 93.24951934814453,
        "Parents": [
```

```
                {
                    "Name": "Machine"
                }
            ],
            "Name": "Wheel"
        },
        {
            "Instances": [],
            "Confidence": 92.61514282226562,
            "Parents": [],
            "Name": "Road"
        },
        {
            "Instances": [],
            "Confidence": 92.37877655029297,
            "Parents": [
                {
                    "Name": "Person"
                }
            ],
            "Name": "Sport"
        },
        {
            "Instances": [],
            "Confidence": 92.37877655029297,
            "Parents": [
                {
                    "Name": "Person"
                }
            ],
            "Name": "Sports"
        },
        {
            "Instances": [
                {
                    "BoundingBox": {
                        "Width": 0.12326609343290329,
                        "Top": 0.6332163214683533,
                        "Left": 0.44815489649772644,
                        "Height": 0.058117982000112534
                    },
                    "Confidence": 92.37877655029297
                }
            ],
```

```
            "Confidence": 92.37877655029297,
            "Parents": [
                {
                    "Name": "Person"
                },
                {
                    "Name": "Sport"
                }
            ],
            "Name": "Skateboard"
        },
        {
            "Instances": [],
            "Confidence": 90.62931060791016,
            "Parents": [
                {
                    "Name": "Person"
                }
            ],
            "Name": "Pedestrian"
        },
        {
            "Instances": [],
            "Confidence": 88.81334686279297,
            "Parents": [],
            "Name": "Asphalt"
        },
        {
            "Instances": [],
            "Confidence": 88.81334686279297,
            "Parents": [],
            "Name": "Tarmac"
        },
        {
            "Instances": [],
            "Confidence": 88.23201751708984,
            "Parents": [],
            "Name": "Path"
        },
        {
            "Instances": [],
            "Confidence": 80.26520538330078,
            "Parents": [],
            "Name": "Urban"
```

```
        },
        {
            "Instances": [],
            "Confidence": 80.26520538330078,
            "Parents": [
                {
                    "Name": "Building"
                },
                {
                    "Name": "Urban"
                }
            ],
            "Name": "Town"
        },
        {
            "Instances": [],
            "Confidence": 80.26520538330078,
            "Parents": [],
            "Name": "Building"
        },
        {
            "Instances": [],
            "Confidence": 80.26520538330078,
            "Parents": [
                {
                    "Name": "Building"
                },
                {
                    "Name": "Urban"
                }
            ],
            "Name": "City"
        },
        {
            "Instances": [],
            "Confidence": 78.37934875488281,
            "Parents": [
                {
                    "Name": "Car"
                },
                {
                    "Name": "Vehicle"
                },
                {
```

```
                    "Name": "Transportation"
                }
            ],
            "Name": "Parking Lot"
        },
        {
            "Instances": [],
            "Confidence": 78.37934875488281,
            "Parents": [
                {
                    "Name": "Car"
                },
                {
                    "Name": "Vehicle"
                },
                {
                    "Name": "Transportation"
                }
            ],
            "Name": "Parking"
        },
        {
            "Instances": [],
            "Confidence": 74.37590026855469,
            "Parents": [
                {
                    "Name": "Building"
                },
                {
                    "Name": "Urban"
                },
                {
                    "Name": "City"
                }
            ],
            "Name": "Downtown"
        },
        {
            "Instances": [],
            "Confidence": 69.84622955322266,
            "Parents": [
                {
                    "Name": "Road"
                }
```

```
            ],
            "Name": "Intersection"
        },
        {
            "Instances": [],
            "Confidence": 57.68518829345703,
            "Parents": [
                {
                    "Name": "Sports Car"
                },
                {
                    "Name": "Car"
                },
                {
                    "Name": "Vehicle"
                },
                {
                    "Name": "Transportation"
                }
            ],
            "Name": "Coupe"
        },
        {
            "Instances": [],
            "Confidence": 57.68518829345703,
            "Parents": [
                {
                    "Name": "Car"
                },
                {
                    "Name": "Vehicle"
                },
                {
                    "Name": "Transportation"
                }
            ],
            "Name": "Sports Car"
        },
        {
            "Instances": [],
            "Confidence": 56.59492111206055,
            "Parents": [
                {
                    "Name": "Path"
```

```
                }
            ],
            "Name": "Sidewalk"
        },
        {
            "Instances": [],
            "Confidence": 56.59492111206055,
            "Parents": [
                {
                    "Name": "Path"
                }
            ],
            "Name": "Pavement"
        },
        {
            "Instances": [],
            "Confidence": 55.58770751953125,
            "Parents": [
                {
                    "Name": "Building"
                },
                {
                    "Name": "Urban"
                }
            ],
            "Name": "Neighborhood"
        }
    ],
    "LabelModelVersion": "2.0"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测图像中的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectLabels。

`detect-moderation-labels`

以下代码示例演示了如何使用 `detect-moderation-labels`。

有关更多信息，请参阅检测不适宜的图像。

Amazon CLI

检测图像中不安全的内容

以下 detect-moderation-labels 命令将检测存储在 Amazon S3 存储桶中的指定图像中不安全的内容。

```
aws rekognition detect-moderation-labels \
    --image "S3Object={Bucket=MyImageS3Bucket,Name=gun.jpg}"
```

输出：

```
{
    "ModerationModelVersion": "3.0",
    "ModerationLabels": [
        {
            "Confidence": 97.29618072509766,
            "ParentName": "Violence",
            "Name": "Weapon Violence"
        },
        {
            "Confidence": 97.29618072509766,
            "ParentName": "",
            "Name": "Violence"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测不安全的图像。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectModerationLabels。

detect-text

以下代码示例演示了如何使用 detect-text。

有关更多信息，请参阅检测图像中的文本。

Amazon CLI

检测图像中的文本

以下 detect-text 命令将检测指定图像中的文本。

```
aws rekognition detect-text \
    --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"ExamplePicture.jpg"}}'
```

输出：


```
{
    "TextDetections": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.24624845385551453,
                    "Top": 0.28288066387176514,
                    "Left": 0.391388863325119,
                    "Height": 0.022687450051307678
                },
                "Polygon": [
                    {
                        "Y": 0.28288066387176514,
                        "X": 0.391388863325119
                    },
                    {
                        "Y": 0.2826388478279114,
                        "X": 0.6376373171806335
                    },
                    {
                        "Y": 0.30532628297805786,
                        "X": 0.637677013874054
                    },
                    {
                        "Y": 0.305568128824234,
                        "X": 0.39142853021621704
                    }
                ]
            },
            "Confidence": 94.35709381103516,
            "DetectedText": "ESTD 1882",
            "Type": "LINE",
            "Id": 0
        },
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33933889865875244,
                    "Top": 0.32603850960731506,
                    "Left": 0.34534579515457153,
                    "Height": 0.07126858830451965
                },
```

```
                "Polygon": [
                    {
                        "Y": 0.32603850960731506,
                        "X": 0.34534579515457153
                    },
                    {
                        "Y": 0.32633158564567566,
                        "X": 0.684684693813324
                    },
                    {
                        "Y": 0.3976001739501953,
                        "X": 0.684575080871582
                    },
                    {
                        "Y": 0.3973070979118347,
                        "X": 0.345236212015152
                    }
                ]
            },
            "Confidence": 99.95779418945312,
            "DetectedText": "BRAINS",
            "Type": "LINE",
            "Id": 1
        },
        {
            "Confidence": 97.22098541259766,
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.061079490929841995,
                    "Top": 0.2843210697174072,
                    "Left": 0.391391396522522,
                    "Height": 0.021029088646173477
                },
                "Polygon": [
                    {
                        "Y": 0.2843210697174072,
                        "X": 0.391391396522522
                    },
                    {
                        "Y": 0.2828207015991211,
                        "X": 0.4524524509906769
                    },
                    {
                        "Y": 0.3038259446620941,
```

```
                        "X": 0.4534534513950348
                    },
                    {
                        "Y": 0.30532634258270264,
                        "X": 0.3923923969268799
                    }
                ]
            },
            "DetectedText": "ESTD",
            "ParentId": 0,
            "Type": "WORD",
            "Id": 2
        },
        {
            "Confidence": 91.49320983886719,
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.07007007300853729,
                    "Top": 0.2828207015991211,
                    "Left": 0.5675675868988037,
                    "Height": 0.02250562608242035
                },
                "Polygon": [
                    {
                        "Y": 0.2828207015991211,
                        "X": 0.5675675868988037
                    },
                    {
                        "Y": 0.2828207015991211,
                        "X": 0.6376376152038574
                    },
                    {
                        "Y": 0.30532634258270264,
                        "X": 0.6376376152038574
                    },
                    {
                        "Y": 0.30532634258270264,
                        "X": 0.5675675868988037
                    }
                ]
            },
            "DetectedText": "1882",
            "ParentId": 0,
            "Type": "WORD",
```

```
            "Id": 3
        },
        {
            "Confidence": 99.95779418945312,
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33933934569358826,
                    "Top": 0.32633158564567566,
                    "Left": 0.3453453481197357,
                    "Height": 0.07127484679222107
                },
                "Polygon": [
                    {
                        "Y": 0.32633158564567566,
                        "X": 0.3453453481197357
                    },
                    {
                        "Y": 0.32633158564567566,
                        "X": 0.684684693813324
                    },
                    {
                        "Y": 0.39759939908981323,
                        "X": 0.6836836934089661
                    },
                    {
                        "Y": 0.39684921503067017,
                        "X": 0.3453453481197357
                    }
                ]
            },
            "DetectedText": "BRAINS",
            "ParentId": 1,
            "Type": "WORD",
            "Id": 4
        }
    ]
}
```


- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectText。

get-celebrity-info

以下代码示例演示了如何使用 get-celebrity-info。

Amazon CLI

获取有关名人的信息

以下 get-celebrity-info 命令显示有关指定名人的信息：id 参数来自先前对 recognize-celebrities 的调用。

```
aws rekognition get-celebrity-info --id nnnnnnn
```

输出：

```
{
    "Name": "Celeb A",
    "Urls": [
        "www.imdb.com/name/aaaaaaaaa"
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的获取有关名人的信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCelebrityInfo。

get-celebrity-recognition

以下代码示例演示了如何使用 get-celebrity-recognition。

Amazon CLI

获取名人识别操作的结果

以下 get-celebrity-recognition 命令显示了您之前通过调用 start-celebrity-recognition 启动的名人识别操作的结果。

```
aws rekognition get-celebrity-recognition  \
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "NextToken": "3D01ClxlCiT31VsRDkAO3IybLb/h5AtDWSGuhYi
+N1FIJwwPtAkuKzDhL2rV3GcwmNt77+12",
    "Celebrities": [
```

```
        {
            "Timestamp": 0,
            "Celebrity": {
                "Confidence": 96.0,
                "Face": {
                    "BoundingBox": {
                        "Width": 0.70333331823349,
                        "Top": 0.16750000417232513,
                        "Left": 0.19555555284023285,
                        "Height": 0.3956249952316284
                    },
                    "Landmarks": [
                        {
                            "Y": 0.31031012535095215,
                            "X": 0.441436767578125,
                            "Type": "eyeLeft"
                        },
                        {
                            "Y": 0.3081788718700409,
                            "X": 0.6437258720397949,
                            "Type": "eyeRight"
                        },
                        {
                            "Y": 0.39542075991630554,
                            "X": 0.5572493076324463,
                            "Type": "nose"
                        },
                        {
                            "Y": 0.4597957134246826,
                            "X": 0.4579732120037079,
                            "Type": "mouthLeft"
                        },
                        {
                            "Y": 0.45688048005104065,
                            "X": 0.6349081993103027,
                            "Type": "mouthRight"
                        }
                    ],
                    "Pose": {
                        "Yaw": 8.943398475646973,
                        "Roll": -2.0309247970581055,
                        "Pitch": -0.5674862861633301
                    },
                    "Quality": {
```

```
                    "Sharpness": 99.40211486816406,
                    "Brightness": 89.47132110595703
                },
                "Confidence": 99.99861145019531
            },
            "Name": "CelebrityA",
            "Urls": [
                "www.imdb.com/name/111111111"
            ],
            "Id": "nnnnnn"
        }
    },
    {
        "Timestamp": 467,
        "Celebrity": {
            "Confidence": 99.0,
            "Face": {
                "BoundingBox": {
                    "Width": 0.6877777576446533,
                    "Top": 0.18437500298023224,
                    "Left": 0.20555555820465088,
                    "Height": 0.3868750035762787
                },
                "Landmarks": [
                    {
                        "Y": 0.31895750761032104,
                        "X": 0.4411413371562958,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.3140959143638611,
                        "X": 0.6523157954216003,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.4016456604003906,
                        "X": 0.5682755708694458,
                        "Type": "nose"
                    },
                    {
                        "Y": 0.46894142031669617,
                        "X": 0.4597797095775604,
                        "Type": "mouthLeft"
                    },
```

```
                        {
                            "Y": 0.46971091628074646,
                            "X": 0.6286435127258301,
                            "Type": "mouthRight"
                        }
                    ],
                    "Pose": {
                        "Yaw": 10.433465957641602,
                        "Roll": -3.347442388534546,
                        "Pitch": 1.3709543943405151
                    },
                    "Quality": {
                        "Sharpness": 99.5531005859375,
                        "Brightness": 88.5764389038086
                    },
                    "Confidence": 99.99148559570312
                },
                "Name": "Jane Celebrity",
                "Urls": [
                    "www.imdb.com/name/111111111"
                ],
                "Id": "nnnnnn"
            }
        }
    ],
    "JobStatus": "SUCCEEDED",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.978118896484375,
        "Codec": "h264",
        "DurationMillis": 4570,
        "FrameHeight": 1920,
        "FrameWidth": 1080
    }
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的识别存储视频中的名人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCelebrityRecognition。

get-content-moderation

以下代码示例演示了如何使用 get-content-moderation。

Amazon CLI

获取不安全内容操作的结果

以下 get-content-moderation 命令显示了您之前通过调用 start-content-moderation 启动的不安全内容操作的结果。

```
aws rekognition get-content-moderation \
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "NextToken": "dlhcKMHMzpCBGFukz6IO3JMcWiJAamCVhXHt3r6b4b5Tfbyw3q7o+Jeezt
+ZpgfOnW9FCCgQ",
    "ModerationLabels": [
        {
            "Timestamp": 0,
            "ModerationLabel": {
                "Confidence": 97.39583587646484,
                "ParentName": "",
                "Name": "Violence"
            }
        },
        {
            "Timestamp": 0,
            "ModerationLabel": {
                "Confidence": 97.39583587646484,
                "ParentName": "Violence",
                "Name": "Weapon Violence"
            }
        }
    ],
    "JobStatus": "SUCCEEDED",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.97515869140625,
        "Codec": "h264",
        "DurationMillis": 6039,
        "FrameHeight": 1920,
        "FrameWidth": 1080
    }
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测不安全的存储视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContentModeration。

get-face-detection

以下代码示例演示了如何使用 get-face-detection。

Amazon CLI

获取人脸检测操作的结果

以下 get-face-detection 命令显示了您之前通过调用 start-face-detection 启动的人脸检测操作的结果。

```
aws rekognition get-face-detection \
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "Faces": [
        {
            "Timestamp": 467,
            "Face": {
                "BoundingBox": {
                    "Width": 0.1560753583908081,
                    "Top": 0.13555361330509186,
                    "Left": -0.0952017530798912,
                    "Height": 0.6934483051300049
                },
                "Landmarks": [
                    {
                        "Y": 0.4013825058937073,
                        "X": -0.041750285774469376,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.41695496439933777,
                        "X": 0.027979329228401184,
                        "Type": "eyeRight"
                    },
                    {
```

```
                    "Y": 0.6375303268432617,
                    "X": -0.04034662991762161,
                    "Type": "mouthLeft"
                },
                {
                    "Y": 0.6497718691825867,
                    "X": 0.013960429467260838,
                    "Type": "mouthRight"
                },
                {
                    "Y": 0.5238034129142761,
                    "X": 0.008022055961191654,
                    "Type": "nose"
                }
            ],
            "Pose": {
                "Yaw": -58.07863998413086,
                "Roll": 1.9384294748306274,
                "Pitch": -24.66305160522461
            },
            "Quality": {
                "Sharpness": 83.14741516113281,
                "Brightness": 25.75942611694336
            },
            "Confidence": 87.7622299194336
        }
    },
    {
        "Timestamp": 967,
        "Face": {
            "BoundingBox": {
                "Width": 0.28559377789497375,
                "Top": 0.19436298310756683,
                "Left": 0.024553587660193443,
                "Height": 0.7216082215309143
            },
            "Landmarks": [
                {
                    "Y": 0.4650231599807739,
                    "X": 0.16269078850746155,
                    "Type": "eyeLeft"
                },
                {
                    "Y": 0.4843238294124603,
```

```
                        "X": 0.2782580852508545,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.71530681848526,
                        "X": 0.1741468608379364,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.7310671210289001,
                        "X": 0.26857468485832214,
                        "Type": "mouthRight"
                    },
                    {
                        "Y": 0.582602322101593,
                        "X": 0.2566150426864624,
                        "Type": "nose"
                    }
                ],
                "Pose": {
                    "Yaw": 11.487052917480469,
                    "Roll": 5.074230670928955,
                    "Pitch": 15.396159172058105
                },
                "Quality": {
                    "Sharpness": 73.32209777832031,
                    "Brightness": 54.96497344970703
                },
                "Confidence": 99.99998474121094
            }
        }
    ],
    "NextToken":
"OzL223pDKy9116O/02KXRqFIEAwxjy4PkgYcm3hSo0rdysbXg5Ex0eFgTGEj0ADEac6S037U",
    "JobStatus": "SUCCEEDED",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.970617294311523,
        "Codec": "h264",
        "DurationMillis": 6806,
        "FrameHeight": 1080,
        "FrameWidth": 1920
    }
```

```
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的在存储视频中检测人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFaceDetection。

get-face-search

以下代码示例演示了如何使用 get-face-search。

Amazon CLI

获取人脸搜索操作的结果

以下 get-face-search 命令显示了您之前通过调用 start-face-search 启动的人脸搜索操作的结果。

```
aws rekognition get-face-search  \
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "Persons": [
        {
            "Timestamp": 467,
            "FaceMatches": [],
            "Person": {
                "Index": 0,
                "Face": {
                    "BoundingBox": {
                        "Width": 0.1560753583908081,
                        "Top": 0.13555361330509186,
                        "Left": -0.0952017530798912,
                        "Height": 0.6934483051300049
                    },
                    "Landmarks": [
                        {
                            "Y": 0.4013825058937073,
                            "X": -0.041750285774469376,
                            "Type": "eyeLeft"
                        },
                        {
```

```
                        "Y": 0.41695496439933777,
                        "X": 0.027979329228401184,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.6375303268432617,
                        "X": -0.04034662991762161,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.6497718691825867,
                        "X": 0.013960429467260838,
                        "Type": "mouthRight"
                    },
                    {
                        "Y": 0.5238034129142761,
                        "X": 0.008022055961191654,
                        "Type": "nose"
                    }
                ],
                "Pose": {
                    "Yaw": -58.07863998413086,
                    "Roll": 1.9384294748306274,
                    "Pitch": -24.66305160522461
                },
                "Quality": {
                    "Sharpness": 83.14741516113281,
                    "Brightness": 25.75942611694336
                },
                "Confidence": 87.7622299194336
            }
        }
    },
    {
        "Timestamp": 967,
        "FaceMatches": [
            {
                "Face": {
                    "BoundingBox": {
                        "Width": 0.12368900328874588,
                        "Top": 0.16007399559020996,
                        "Left": 0.5901259779930115,
                        "Height": 0.2514039874076843
                    },
```

```
                    "FaceId": "056a95fa-2060-4159-9cab-7ed4daa030fa",
                    "ExternalImageId": "image3.jpg",
                    "Confidence": 100.0,
                    "ImageId": "08f8a078-8929-37fd-8e8f-aadf690e8232"
                },
                "Similarity": 98.44476318359375
            }
        ],
        "Person": {
            "Index": 1,
            "Face": {
                "BoundingBox": {
                    "Width": 0.28559377789497375,
                    "Top": 0.19436298310756683,
                    "Left": 0.024553587660193443,
                    "Height": 0.7216082215309143
                },
                "Landmarks": [
                    {
                        "Y": 0.4650231599807739,
                        "X": 0.16269078850746155,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.4843238294124603,
                        "X": 0.2782580852508545,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.71530681848526,
                        "X": 0.1741468608379364,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.7310671210289001,
                        "X": 0.26857468485832214,
                        "Type": "mouthRight"
                    },
                    {
                        "Y": 0.582602322101593,
                        "X": 0.2566150426864624,
                        "Type": "nose"
                    }
                ],
```

```
                    "Pose": {
                        "Yaw": 11.487052917480469,
                        "Roll": 5.074230670928955,
                        "Pitch": 15.396159172058105
                    },
                    "Quality": {
                        "Sharpness": 73.32209777832031,
                        "Brightness": 54.96497344970703
                    },
                    "Confidence": 99.99998474121094
                }
            }
        }
    ],
    "NextToken": "5bkgcezyuaqhtWk3C8OTW6cjRghrwV9XDMivm5B3MXm+Lv6G+L+GejyFHPhoNa/
ldXIC4c/d",
    "JobStatus": "SUCCEEDED",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.970617294311523,
        "Codec": "h264",
        "DurationMillis": 6806,
        "FrameHeight": 1080,
        "FrameWidth": 1920
    }
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的搜索存储视频中的人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFaceSearch。

get-label-detection

以下代码示例演示了如何使用 get-label-detection。

Amazon CLI

获取物体和场景检测操作的结果

以下 get-label-detection 命令显示了您之前通过调用 start-label-detection 启动的对象和场景检测操作的结果。

```
aws rekognition get-label-detection  \
```

```
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "Labels": [
        {
            "Timestamp": 0,
            "Label": {
                "Instances": [],
                "Confidence": 50.19071578979492,
                "Parents": [
                    {
                        "Name": "Person"
                    },
                    {
                        "Name": "Crowd"
                    }
                ],
                "Name": "Audience"
            }
        },
        {
            "Timestamp": 0,
            "Label": {
                "Instances": [],
                "Confidence": 55.74115753173828,
                "Parents": [
                    {
                        "Name": "Room"
                    },
                    {
                        "Name": "Indoors"
                    },
                    {
                        "Name": "School"
                    }
                ],
                "Name": "Classroom"
            }
        }
    ],
    "JobStatus": "SUCCEEDED",
```

```
    "LabelModelVersion": "2.0",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.970617294311523,
        "Codec": "h264",
        "DurationMillis": 6806,
        "FrameHeight": 1080,
        "FrameWidth": 1920
    },
    "NextToken": "BMugzAi4L72IERzQdbpyMQuEFBsjlo5W0Yx3mfG+sR9mm98E1/
CpObenspRfs/5FBQFs4X7G"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测视频中的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLabelDetection。

get-person-tracking

以下代码示例演示了如何使用 get-person-tracking。

Amazon CLI

获取人物轨迹跟踪操作的结果

以下 get-person-tracking 命令显示了您之前通过调用 start-person-tracking 启动的人物轨迹跟踪操作的结果。

```
aws rekognition get-person-tracking  \
    --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

输出：

```
{
    "Persons": [
        {
            "Timestamp": 500,
            "Person": {
                "BoundingBox": {
                    "Width": 0.4151041805744171,
                    "Top": 0.07870370149612427,
                    "Left": 0.0,
```

```
                    "Height": 0.9212962985038757
                },
                "Index": 0
            }
        },
        {
            "Timestamp": 567,
            "Person": {
                "BoundingBox": {
                    "Width": 0.4755208194255829,
                    "Top": 0.07777778059244156,
                    "Left": 0.0,
                    "Height": 0.9194444417953491
                },
                "Index": 0
            }
        }
    ],
    "NextToken": "D/vRIYNyhG79ugdta3f+8cRg9oSRo
+HigGOuxRiYpTn0ExnqTi1CJektVAc4HrAXDv25eHYk",
    "JobStatus": "SUCCEEDED",
    "VideoMetadata": {
        "Format": "QuickTime / MOV",
        "FrameRate": 29.970617294311523,
        "Codec": "h264",
        "DurationMillis": 6806,
        "FrameHeight": 1080,
        "FrameWidth": 1920
    }
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的人物轨迹跟踪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPersonTracking。

index-faces

以下代码示例演示了如何使用 index-faces。

有关更多信息，请参阅将人脸添加到集合中。

Amazon CLI

将人脸添加到集合

以下 index-faces 命令将在图像中找到的人脸添加到指定的集合中。

```
aws rekognition index-faces \
    --image '{"S3Object":{"Bucket":"MyVideoS3Bucket","Name":"MyPicture.jpg"}}' \
    --collection-id MyCollection \
    --max-faces 1 \
    --quality-filter "AUTO" \
    --detection-attributes "ALL" \
    --external-image-id "MyPicture.jpg"
```

输出：

```
{
    "FaceRecords": [
        {
            "FaceDetail": {
                "Confidence": 99.993408203125,
                "Eyeglasses": {
                    "Confidence": 99.11750030517578,
                    "Value": false
                },
                "Sunglasses": {
                    "Confidence": 99.98249053955078,
                    "Value": false
                },
                "Gender": {
                    "Confidence": 99.92769622802734,
                    "Value": "Male"
                },
                "Landmarks": [
                    {
                        "Y": 0.26750367879867554,
                        "X": 0.6202793717384338,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.26642778515815735,
                        "X": 0.6787431836128235,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.31361380219459534,
                        "X": 0.6421601176261902,
```

```
            "Type": "nose"
        },
        {
            "Y": 0.3495299220085144,
            "X": 0.6216195225715637,
            "Type": "mouthLeft"
        },
        {
            "Y": 0.35194727778434753,
            "X": 0.669899046421051,
            "Type": "mouthRight"
        },
        {
            "Y": 0.26844894886016846,
            "X": 0.6210268139839172,
            "Type": "leftPupil"
        },
        {
            "Y": 0.26707562804222107,
            "X": 0.6817160844802856,
            "Type": "rightPupil"
        },
        {
            "Y": 0.24834522604942322,
            "X": 0.6018546223640442,
            "Type": "leftEyeBrowLeft"
        },
        {
            "Y": 0.24397172033786774,
            "X": 0.6172008514404297,
            "Type": "leftEyeBrowUp"
        },
        {
            "Y": 0.24677404761314392,
            "X": 0.6339119076728821,
            "Type": "leftEyeBrowRight"
        },
        {
            "Y": 0.24582654237747192,
            "X": 0.6619398593902588,
            "Type": "rightEyeBrowLeft"
        },
        {
            "Y": 0.23973053693771362,
```

```
                    "X": 0.6804757118225098,
                    "Type": "rightEyeBrowUp"
                },
                {
                    "Y": 0.24441994726657867,
                    "X": 0.6978968977928162,
                    "Type": "rightEyeBrowRight"
                },
                {
                    "Y": 0.2695908546447754,
                    "X": 0.6085202693939209,
                    "Type": "leftEyeLeft"
                },
                {
                    "Y": 0.26716896891593933,
                    "X": 0.6315826177597046,
                    "Type": "leftEyeRight"
                },
                {
                    "Y": 0.26289820671081543,
                    "X": 0.6202316880226135,
                    "Type": "leftEyeUp"
                },
                {
                    "Y": 0.27123287320137024,
                    "X": 0.6205548048019409,
                    "Type": "leftEyeDown"
                },
                {
                    "Y": 0.2668408751487732,
                    "X": 0.6663622260093689,
                    "Type": "rightEyeLeft"
                },
                {
                    "Y": 0.26741549372673035,
                    "X": 0.6910083889961243,
                    "Type": "rightEyeRight"
                },
                {
                    "Y": 0.2614026665687561,
                    "X": 0.6785826086997986,
                    "Type": "rightEyeUp"
                },
                {
```

```
                    "Y": 0.27075251936912537,
                    "X": 0.6789616942405701,
                    "Type": "rightEyeDown"
                },
                {
                    "Y": 0.3211299479007721,
                    "X": 0.6324167847633362,
                    "Type": "noseLeft"
                },
                {
                    "Y": 0.32276326417922974,
                    "X": 0.6558475494384766,
                    "Type": "noseRight"
                },
                {
                    "Y": 0.34385165572166443,
                    "X": 0.6444970965385437,
                    "Type": "mouthUp"
                },
                {
                    "Y": 0.3671635091304779,
                    "X": 0.6459195017814636,
                    "Type": "mouthDown"
                }
            ],
            "Pose": {
                "Yaw": -9.54541015625,
                "Roll": -0.5709401965141296,
                "Pitch": 0.6045494675636292
            },
            "Emotions": [
                {
                    "Confidence": 39.90074157714844,
                    "Type": "HAPPY"
                },
                {
                    "Confidence": 23.38753890991211,
                    "Type": "CALM"
                },
                {
                    "Confidence": 5.840933322906494,
                    "Type": "CONFUSED"
                }
            ],
```

```
                "AgeRange": {
                    "High": 63,
                    "Low": 45
                },
                "EyesOpen": {
                    "Confidence": 99.80887603759766,
                    "Value": true
                },
                "BoundingBox": {
                    "Width": 0.18562500178813934,
                    "Top": 0.1618015021085739,
                    "Left": 0.5575000047683716,
                    "Height": 0.24770642817020416
                },
                "Smile": {
                    "Confidence": 99.69740295410156,
                    "Value": false
                },
                "MouthOpen": {
                    "Confidence": 99.97393798828125,
                    "Value": false
                },
                "Quality": {
                    "Sharpness": 95.54405975341797,
                    "Brightness": 63.867706298828125
                },
                "Mustache": {
                    "Confidence": 97.05007934570312,
                    "Value": false
                },
                "Beard": {
                    "Confidence": 87.34505462646484,
                    "Value": false
                }
            },
            "Face": {
                "BoundingBox": {
                    "Width": 0.18562500178813934,
                    "Top": 0.1618015021085739,
                    "Left": 0.5575000047683716,
                    "Height": 0.24770642817020416
                },
                "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
                "ExternalImageId": "example-image.jpg",
```

```
                "Confidence": 99.993408203125,
                "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
            }
        }
    ],
    "UnindexedFaces": [],
    "FaceModelVersion": "3.0",
    "OrientationCorrection": "ROTATE_0"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的将人脸添加到集合中。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IndexFaces。

`list-collections`

以下代码示例演示了如何使用 `list-collections`。

有关更多信息，请参阅列出集合。

Amazon CLI

列出可用的集合

以下 `list-collections` 命令列出了 Amazon 账户中的可用集合。

```
aws rekognition list-collections
```

输出：

```
{
    "FaceModelVersions": [
        "2.0",
        "3.0",
        "3.0",
        "3.0",
        "4.0",
        "1.0",
        "3.0",
        "4.0",
        "4.0",
        "4.0"
    ],
```

```
    "CollectionIds": [
        "MyCollection1",
        "MyCollection2",
        "MyCollection3",
        "MyCollection4",
        "MyCollection5",
        "MyCollection6",
        "MyCollection7",
        "MyCollection8",
        "MyCollection9",
        "MyCollection10"
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的列出集合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListCollections。

`list-faces`

以下代码示例演示了如何使用 `list-faces`。

有关更多信息，请参阅列出集合中的人脸。

Amazon CLI

列出集合中的人脸

以下 `list-faces` 命令将列出指定集合中的人脸。

```
aws rekognition list-faces \
    --collection-id MyCollection
```

输出：

```
{
    "FaceModelVersion": "3.0",
    "Faces": [
        {
            "BoundingBox": {
                "Width": 0.5216310024261475,
                "Top": 0.3256250023841858,
                "Left": 0.13394300639629364,
```

```
                "Height": 0.3918749988079071
            },
            "FaceId": "0040279c-0178-436e-b70a-e61b074e96b0",
            "ExternalImageId": "image1.jpg",
            "Confidence": 100.0,
            "ImageId": "f976e487-3719-5e2d-be8b-ea2724c26991"
        },
        {
            "BoundingBox": {
                "Width": 0.5074880123138428,
                "Top": 0.3774999976158142,
                "Left": 0.18302799761295319,
                "Height": 0.3812499940395355
            },
            "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",
            "ExternalImageId": "image2.jpg",
            "Confidence": 99.99930572509766,
            "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"
        },
        {
            "BoundingBox": {
                "Width": 0.5574039816856384,
                "Top": 0.37187498807907104,
                "Left": 0.14559100568294525,
                "Height": 0.4181250035762787
            },
            "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",
            "ExternalImageId": "image3.jpg",
            "Confidence": 99.99960327148438,
            "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"
        },
        {
            "BoundingBox": {
                "Width": 0.18562500178813934,
                "Top": 0.1618019938468933,
                "Left": 0.5575000047683716,
                "Height": 0.24770599603652954
            },
            "FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
            "ExternalImageId": "image4.jpg",
            "Confidence": 99.99340057373047,
            "ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
        },
        {
```

```
            "BoundingBox": {
                "Width": 0.5307819843292236,
                "Top": 0.2862499952316284,
                "Left": 0.1564060002565384,
                "Height": 0.3987500071525574
            },
            "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
            "ExternalImageId": "image5.jpg",
            "Confidence": 99.99970245361328,
            "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
        },
        {
            "BoundingBox": {
                "Width": 0.5773710012435913,
                "Top": 0.34437501430511475,
                "Left": 0.12396000325679779,
                "Height": 0.4337500035762787
            },
            "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
            "ExternalImageId": "image6.jpg",
            "Confidence": 100.0,
            "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
        },
        {
            "BoundingBox": {
                "Width": 0.5349419713020325,
                "Top": 0.29124999046325684,
                "Left": 0.16389399766921997,
                "Height": 0.40187498927116394
            },
            "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
            "ExternalImageId": "image7.jpg",
            "Confidence": 99.99979400634766,
            "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
        },
        {
            "BoundingBox": {
                "Width": 0.41499999165534973,
                "Top": 0.09187500178813934,
                "Left": 0.28083300590515137,
                "Height": 0.3112500011920929
            },
            "FaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",
            "ExternalImageId": "image8.jpg",
```

```
            "Confidence": 99.99769592285156,
            "ImageId": "a294da46-2cb1-5cc4-9045-61d7ca567662"
        },
        {
            "BoundingBox": {
                "Width": 0.48166701197624207,
                "Top": 0.20999999344348907,
                "Left": 0.21250000596046448,
                "Height": 0.36125001311302185
            },
            "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",
            "ExternalImageId": "image9.jpg",
            "Confidence": 99.99949645996094,
            "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"
        },
        {
            "BoundingBox": {
                "Width": 0.18562500178813934,
                "Top": 0.1618019938468933,
                "Left": 0.5575000047683716,
                "Height": 0.24770599603652954
            },
            "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
            "ExternalImageId": "image10.jpg",
            "Confidence": 99.99340057373047,
            "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的列出集合中的人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFaces。

`list-stream-processors`

以下代码示例演示了如何使用 `list-stream-processors`。

Amazon CLI

获取您账户中的流处理器

以下 `list-stream-processors` 命令列出了您账户中的流处理器以及每个流处理器的状态。

```
aws rekognition list-stream-processors
```

输出：

```
{
    "StreamProcessors": [
        {
            "Status": "STOPPED",
            "Name": "my-stream-processor"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStreamProcessors。

recognize-celebrities

以下代码示例演示了如何使用 recognize-celebrities。

有关更多信息，请参阅识别图像中的名人。

Amazon CLI

识别图像中的名人

以下 recognize-celebrities 命令将识别存储在 Amazon S3 存储桶中的指定图像中的名人。

```
aws rekognition recognize-celebrities \
    --image "S3Object={Bucket=MyImageS3Bucket,Name=moviestars.jpg}"
```

输出：

```
{
    "UnrecognizedFaces": [
        {
            "BoundingBox": {
                "Width": 0.14416666328907013,
                "Top": 0.07777778059244156,
                "Left": 0.625,
                "Height": 0.2746031880378723
```

```
            },
            "Confidence": 99.9990234375,
            "Pose": {
                "Yaw": 10.80408763885498,
                "Roll": -12.761146545410156,
                "Pitch": 10.96889877319336
            },
            "Quality": {
                "Sharpness": 94.1185531616211,
                "Brightness": 79.18367004394531
            },
            "Landmarks": [
                {
                    "Y": 0.18220913410186768,
                    "X": 0.6702951788902283,
                    "Type": "eyeLeft"
                },
                {
                    "Y": 0.16337193548679352,
                    "X": 0.7188183665275574,
                    "Type": "eyeRight"
                },
                {
                    "Y": 0.20739148557186127,
                    "X": 0.7055801749229431,
                    "Type": "nose"
                },
                {
                    "Y": 0.2889308035373688,
                    "X": 0.687512218952179,
                    "Type": "mouthLeft"
                },
                {
                    "Y": 0.2706988751888275,
                    "X": 0.7250053286552429,
                    "Type": "mouthRight"
                }
            ]
        }
    ],
    "CelebrityFaces": [
        {
            "MatchConfidence": 100.0,
            "Face": {
```

```
                "BoundingBox": {
                    "Width": 0.14000000059604645,
                    "Top": 0.1190476194024086,
                    "Left": 0.82833331823349,
                    "Height": 0.2666666805744171
                },
                "Confidence": 99.99359130859375,
                "Pose": {
                    "Yaw": -10.509642601013184,
                    "Roll": -14.51749324798584,
                    "Pitch": 13.799399375915527
                },
                "Quality": {
                    "Sharpness": 78.74752044677734,
                    "Brightness": 42.201324462890625
                },
                "Landmarks": [
                    {
                        "Y": 0.2290833294391632,
                        "X": 0.8709492087364197,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.20639978349208832,
                        "X": 0.9153988361358643,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.2541764378547668 5,
                        "X": 0.8907724022865295,
                        "Type": "nose"
                    },
                    {
                        "Y": 0.32729196548461914,
                        "X": 0.8876466155052185,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.3115464746952057,
                        "X": 0.9238573312759399,
                        "Type": "mouthRight"
                    }
                ]
            },
```

```
            "Name": "Celeb A",
            "Urls": [
                "www.imdb.com/name/aaaaaaaaa"
            ],
            "Id": "1111111"
        },
        {
            "MatchConfidence": 97.0,
            "Face": {
                "BoundingBox": {
                    "Width": 0.13333334028720856,
                    "Top": 0.24920634925365448,
                    "Left": 0.4449999928474426,
                    "Height": 0.2539682686328888
                },
                "Confidence": 99.99979400634766,
                "Pose": {
                    "Yaw": 6.557040691375732,
                    "Roll": -7.316643714904785,
                    "Pitch": 9.272967338562012
                },
                "Quality": {
                    "Sharpness": 83.23492431640625,
                    "Brightness": 78.83267974853516
                },
                "Landmarks": [
                    {
                        "Y": 0.3625510632991791,
                        "X": 0.48898839950561523,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.35366007685661316,
                        "X": 0.5313721299171448,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.3894785940647125,
                        "X": 0.5173314809799194,
                        "Type": "nose"
                    },
                    {
                        "Y": 0.44889405369758606,
                        "X": 0.5020005702972412,
```

```
                    "Type": "mouthLeft"
                },
                {
                    "Y": 0.4408611059188843,
                    "X": 0.5351271629333496,
                    "Type": "mouthRight"
                }
            ]
        },
        "Name": "Celeb B",
        "Urls": [
            "www.imdb.com/name/bbbbbbbbb"
        ],
        "Id": "2222222"
    },
    {
        "MatchConfidence": 100.0,
        "Face": {
            "BoundingBox": {
                "Width": 0.12416666746139526,
                "Top": 0.2968254089355469,
                "Left": 0.2150000035762787,
                "Height": 0.23650793731212616
            },
            "Confidence": 99.99958801269531,
            "Pose": {
                "Yaw": 7.801797866821289,
                "Roll": -8.326810836791992,
                "Pitch": 7.844768047332764
            },
            "Quality": {
                "Sharpness": 86.93206024169922,
                "Brightness": 79.81291198730469
            },
            "Landmarks": [
                {
                    "Y": 0.4027804136276245,
                    "X": 0.2575301229953766,
                    "Type": "eyeLeft"
                },
                {
                    "Y": 0.3934555947780609,
                    "X": 0.2956969439983368,
                    "Type": "eyeRight"
```

```
                    },
                    {
                        "Y": 0.4309830069541931,
                        "X": 0.2837020754814148,
                        "Type": "nose"
                    },
                    {
                        "Y": 0.48186683654785156,
                        "X": 0.26812544465065,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.47338807582855225,
                        "X": 0.29905644059181213,
                        "Type": "mouthRight"
                    }
                ]
            },
            "Name": "Celeb C",
            "Urls": [
                "www.imdb.com/name/ccccccccc"
            ],
            "Id": "3333333"
        },
        {
            "MatchConfidence": 97.0,
            "Face": {
                "BoundingBox": {
                    "Width": 0.11916666477918625,
                    "Top": 0.3698412775993347,
                    "Left": 0.008333333767950535,
                    "Height": 0.22698412835597992
                },
                "Confidence": 99.99999237060547,
                "Pose": {
                    "Yaw": 16.38478660583496,
                    "Roll": -1.0260354280471802,
                    "Pitch": 5.975185394287109
                },
                "Quality": {
                    "Sharpness": 83.23492431640625,
                    "Brightness": 61.408443450927734
                },
                "Landmarks": [
```

```
                    {
                        "Y": 0.4632347822189331,
                        "X": 0.049406956881284714,
                        "Type": "eyeLeft"
                    },
                    {
                        "Y": 0.46388113498687744,
                        "X": 0.08722897619009018,
                        "Type": "eyeRight"
                    },
                    {
                        "Y": 0.5020678639411926,
                        "X": 0.0758260041475296,
                        "Type": "nose"
                    },
                    {
                        "Y": 0.544157862663269,
                        "X": 0.054029736667871475,
                        "Type": "mouthLeft"
                    },
                    {
                        "Y": 0.5463630557060242,
                        "X": 0.08464983850717545,
                        "Type": "mouthRight"
                    }
                ]
            },
            "Name": "Celeb D",
            "Urls": [
                "www.imdb.com/name/ddddddddd"
            ],
            "Id": "4444444"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的识别图像中的名人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RecognizeCelebrities。

`search-faces-by-image`

以下代码示例演示了如何使用 `search-faces-by-image`。

有关更多信息，请参阅搜索人脸（图像）。

Amazon CLI

搜索集合中与图像中最大人脸匹配的人脸。

以下 search-faces-by-image 命令将搜索集合中与指定图像中最大人脸相匹配的人脸。

```
aws rekognition search-faces-by-image \
    --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"ExamplePerson.jpg"}}' \
    --collection-id MyFaceImageCollection

{
    "SearchedFaceBoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618015021085739,
        "Left": 0.5575000047683716,
        "Height": 0.24770642817020416
    },
    "SearchedFaceConfidence": 99.993408203125,
    "FaceMatches": [
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.18562500178813934,
                    "Top": 0.1618019938468933,
                    "Left": 0.5575000047683716,
                    "Height": 0.24770599603652954
                },
                "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
                "ExternalImageId": "example-image.jpg",
                "Confidence": 99.99340057373047,
                "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
            },
            "Similarity": 99.97913360595703
        },
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.18562500178813934,
                    "Top": 0.1618019938468933,
                    "Left": 0.5575000047683716,
                    "Height": 0.24770599603652954
                },
```

```
            "FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
            "ExternalImageId": "image3.jpg",
            "Confidence": 99.99340057373047,
            "ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
        },
        "Similarity": 99.97913360595703
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.41499999165534973,
                "Top": 0.09187500178813934,
                "Left": 0.28083300590515137,
                "Height": 0.3112500011920929
            },
            "FaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",
            "ExternalImageId": "image2.jpg",
            "Confidence": 99.99769592285156,
            "ImageId": "a294da46-2cb1-5cc4-9045-61d7ca567662"
        },
        "Similarity": 99.18069458007812
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.48166701197624207,
                "Top": 0.20999999344348907,
                "Left": 0.21250000596046448,
                "Height": 0.36125001311302185
            },
            "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",
            "ExternalImageId": "image1.jpg",
            "Confidence": 99.99949645996094,
            "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"
        },
        "Similarity": 98.66607666015625
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5349419713020325,
                "Top": 0.29124999046325684,
                "Left": 0.16389399766921997,
                "Height": 0.40187498927116394
```

```
            },
            "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
            "ExternalImageId": "image9.jpg",
            "Confidence": 99.99979400634766,
            "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
        },
        "Similarity": 98.24278259277344
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5307819843292236,
                "Top": 0.2862499952316284,
                "Left": 0.1564060002565384,
                "Height": 0.3987500071525574
            },
            "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
            "ExternalImageId": "image10.jpg",
            "Confidence": 99.99970245361328,
            "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
        },
        "Similarity": 98.10665893554688
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5074880123138428,
                "Top": 0.3774999976158142,
                "Left": 0.18302799761295319,
                "Height": 0.3812499940395355
            },
            "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",
            "ExternalImageId": "image6.jpg",
            "Confidence": 99.99930572509766,
            "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"
        },
        "Similarity": 98.10526275634766
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5574039816856384,
                "Top": 0.37187498807907104,
                "Left": 0.14559100568294525,
```

```
                    "Height": 0.4181250035762787
                },
                "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",
                "ExternalImageId": "image5.jpg",
                "Confidence": 99.99960327148438,
                "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"
            },
            "Similarity": 97.94659423828125
        },
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.5773710012435913,
                    "Top": 0.34437501430511475,
                    "Left": 0.12396000325679779,
                    "Height": 0.4337500035762787
                },
                "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
                "ExternalImageId": "image8.jpg",
                "Confidence": 100.0,
                "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
            },
            "Similarity": 97.93476867675781
        }
    ],

 "FaceModelVersion": "3.0"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用图像搜索人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchFacesByImage。

search-faces

以下代码示例演示了如何使用 search-faces。

有关更多信息，请参阅搜索人脸（面容 ID）。

Amazon CLI

搜索集合中与人脸 ID 匹配的人脸

以下 search-faces 命令将搜索集合中与指定人脸 ID 相匹配的人脸。

```
aws rekognition search-faces \
    --face-id 8d3cfc70-4ba8-4b36-9644-90fba29c2dac \
    --collection-id MyCollection
```

输出：

```
{
    "SearchedFaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",
    "FaceModelVersion": "3.0",
    "FaceMatches": [
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.48166701197624207,
                    "Top": 0.20999999344348907,
                    "Left": 0.21250000596046448,
                    "Height": 0.36125001311302185
                },
                "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",
                "ExternalImageId": "image1.jpg",
                "Confidence": 99.99949645996094,
                "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"
            },
            "Similarity": 99.30997467041016
        },
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.18562500178813934,
                    "Top": 0.1618019938468933,
                    "Left": 0.5575000047683716,
                    "Height": 0.24770599603652954
                },
                "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
                "ExternalImageId": "example-image.jpg",
                "Confidence": 99.99340057373047,
                "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
            },
            "Similarity": 99.24862670898438
        },
        {
            "Face": {
                "BoundingBox": {
```

```
                "Width": 0.18562500178813934,
                "Top": 0.1618019938468933,
                "Left": 0.5575000047683716,
                "Height": 0.24770599603652954
            },
            "FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
            "ExternalImageId": "image3.jpg",
            "Confidence": 99.99340057373047,
            "ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
        },
        "Similarity": 99.24862670898438
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5349419713020325,
                "Top": 0.29124999046325684,
                "Left": 0.16389399766921997,
                "Height": 0.40187498927116394
            },
            "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
            "ExternalImageId": "image9.jpg",
            "Confidence": 99.99979400634766,
            "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
        },
        "Similarity": 96.73158264160156
    },
    {
        "Face": {
            "BoundingBox": {
                "Width": 0.5307819843292236,
                "Top": 0.2862499952316284,
                "Left": 0.1564060002565384,
                "Height": 0.3987500071525574
            },
            "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
            "ExternalImageId": "image10.jpg",
            "Confidence": 99.99970245361328,
            "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
        },
        "Similarity": 96.48291015625
    },
    {
        "Face": {
```

```
                "BoundingBox": {
                    "Width": 0.5074880123138428,
                    "Top": 0.3774999976158142,
                    "Left": 0.18302799761295319,
                    "Height": 0.3812499940395355
                },
                "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",
                "ExternalImageId": "image6.jpg",
                "Confidence": 99.99930572509766,
                "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"
            },
            "Similarity": 96.43287658691406
        },
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.5574039816856384,
                    "Top": 0.37187498807907104,
                    "Left": 0.14559100568294525,
                    "Height": 0.4181250035762787
                },
                "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",
                "ExternalImageId": "image5.jpg",
                "Confidence": 99.99960327148438,
                "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"
            },
            "Similarity": 95.25305938720703
        },
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.5773710012435913,
                    "Top": 0.34437501430511475,
                    "Left": 0.12396000325679779,
                    "Height": 0.4337500035762787
                },
                "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
                "ExternalImageId": "image8.jpg",
                "Confidence": 100.0,
                "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
            },
            "Similarity": 95.22837829589844
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用人脸 ID 搜索人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchFaces。

`start-celebrity-recognition`

以下代码示例演示了如何使用 `start-celebrity-recognition`。

Amazon CLI

开始识别存储视频中的名人

以下 `start-celebrity-recognition` 命令将启动作业，以在存储在 Amazon S3 存储桶中的指定视频文件中查找名人。

```
aws rekognition start-celebrity-recognition \
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的识别存储视频中的名人。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartCelebrityRecognition。

`start-content-moderation`

以下代码示例演示了如何使用 `start-content-moderation`。

Amazon CLI

开始识别存储视频中的不安全内容

以下 `start-content-moderation` 命令将启动作业，以在存储在 Amazon S3 存储桶中的指定视频中检测不安全的内容。

```
aws rekognition start-content-moderation \
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测不安全的存储视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartContentModeration。

start-face-detection

以下代码示例演示了如何使用 start-face-detection。

Amazon CLI

检测视频中的人脸

以下 start-face-detection 命令将启动作业，以在存储在 Amazon S3 存储桶中的指定视频文件中检测人脸。

```
aws rekognition start-face-detection
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的在存储视频中检测人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartFaceDetection。

start-face-search

以下代码示例演示了如何使用 start-face-search。

Amazon CLI

在集合中搜索与视频中检测到的人脸相匹配的人脸

以下 start-face-search 命令将启动作业，以在集合中搜索与在存储在 Amazon S3 存储桶中的指定视频文件中检测到的人脸相匹配的人脸。

```
aws rekognition start-face-search \
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}" \
    --collection-id collection
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的搜索存储视频中的人脸。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartFaceSearch。

start-label-detection

以下代码示例演示了如何使用 start-label-detection。

Amazon CLI

检测视频中的物体和场景

以下 start-label-detection 命令将启动作业，以在存储在 Amazon S3 存储桶中的指定视频文件中检测物体和场景。

```
aws rekognition start-label-detection \
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
```

```
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的检测视频中的标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartLabelDetection。

start-person-tracking

以下代码示例演示了如何使用 start-person-tracking。

Amazon CLI

在存储的视频中开始人物轨迹跟踪

以下 start-person-tracking 命令将启动作业，以在存储在 Amazon S3 存储桶中的指定视频文件中跟踪人物轨迹。

```
aws rekognition start-person-tracking \
    --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

输出：

```
{
    "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的人物轨迹跟踪。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartPersonTracking。

start-stream-processor

以下代码示例演示了如何使用 start-stream-processor。

Amazon CLI

启动流处理器

以下 start-stream-processor 命令将启动指定的视频流处理器。

```
aws rekognition start-stream-processor \
```

```
    --name my-stream-processor
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartStreamProcessor。

`stop-stream-processor`

以下代码示例演示了如何使用 `stop-stream-processor`。

Amazon CLI

停止正在运行的流处理器

以下 `stop-stream-processor` 命令将停止指定的正在运行的流处理器。

```
aws rekognition stop-stream-processor \
    --name my-stream-processor
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Rekognition 开发人员指南》中的使用流视频。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopStreamProcessor。

Amazon RAM使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon RAM 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

accept-resource-share-invitation

以下代码示例演示了如何使用 accept-resource-share-invitation。

Amazon CLI

接受资源共享邀请

以下 accept-resource-share-invitation 示例接受了指定的资源共享邀请。受邀账户中的主体可以立即开始使用共享的资源。

```
aws ram accept-resource-share-invitation \
    --resource-share-invitation-arn arn:aws:ram:us-west-2:111111111111:resource-share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE
```

输出：

```
{
    "resourceShareInvitation": {
        "resourceShareInvitationArn": "arn:aws:ram:us-west-2:111111111111:resource-share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE",
        "resourceShareName": "MyLicenseShare",
        "resourceShareArn": "arn:aws:ram:us-west-2:111111111111:resource-share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE",
        "senderAccountId": "111111111111",
        "receiverAccountId": "222222222222",
        "invitationTimestamp": "2021-09-22T15:07:35.620000-07:00",
        "status": "ACCEPTED"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptResourceShareInvitation。

associate-resource-share-permission

以下代码示例演示了如何使用 associate-resource-share-permission。

Amazon CLI

将 RAM 托管权限与资源共享关联

以下 associate-resource-share-permission 示例将相关资源类型的现有托管权限替换为指定的托管权限。对相关资源类型的所有资源的访问受新权限的约束。

```
aws ram associate-resource-share-permission \
    --permission-arn arn:aws:ram::aws:permission/
AWSRAMPermissionGlueDatabaseReadWrite \
    --replace \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE
```

输出：

```
{
    "returnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateResourceSharePermission。

associate-resource-share

以下代码示例演示了如何使用 associate-resource-share。

Amazon CLI

示例 1：将资源与资源共享关联

以下 associate-resource-share 示例将许可证配置添加到指定的资源共享。

```
aws ram associate-resource-share \
    --resource-share arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \
    --resource-arns arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE
```

输出：

```
{
    "resourceShareAssociations": [
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE",
```

```
            "associatedEntity": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
            "associationType": "RESOURCE",
            "status": "ASSOCIATING",
            "external": false
        }
    ]
}
```

示例 2：将主体与资源共享关联

以下 associate-resource-share 示例向指定组织单位中的所有账户授予访问指定资源共享的权限。

```
aws ram associate-resource-share \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \
    --principals arn:aws:organizations::123456789012:ou/o-63bEXAMPLE/ou-46xi-
rEXAMPLE
```

输出：

```
{
    "resourceShareAssociations": [
        {
            "status": "ASSOCIATING",
            "associationType": "PRINCIPAL",
            "associatedEntity": "arn:aws:organizations::123456789012:ou/
o-63bEXAMPLE/ou-46xi-rEXAMPLE",
            "external": false,
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateResourceShare。

create-resource-share

以下代码示例演示了如何使用 create-resource-share。

Amazon CLI

示例 1：创建资源共享

以下 create-resource-share 示例创建了一个使用指定名称的空资源共享。您必须分别向共享中添加资源、主体和权限。

```
aws ram create-resource-share \
    --name MyNewResourceShare
```

输出：

```
{
    "resourceShare": {
        "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/4476c27d-8feb-4b21-afe9-7de23EXAMPLE",
        "name": "MyNewResourceShare",
        "owningAccountId": "123456789012",
        "allowExternalPrincipals": true,
        "status": "ACTIVE",
        "creationTime": 1634586271.302,
        "lastUpdatedTime": 1634586271.302
    }
}
```

示例 2：创建以 Amazon 账户为主体的资源共享

以下 create-resource-share 示例创建了一个资源共享，并向指定的 Amazon 账户（222222222222）授予访问权限。如果指定的主体不属于同一个 Amazon 组织，则会发送邀请，并且必须在接受邀请后再授予访问权限。

```
aws ram create-resource-share \
    --name MyNewResourceShare \
    --principals 222222222222
```

示例 3：创建仅限于您的 Amazon 组织的资源共享

以下 create-resource-share 示例创建一个仅限于您的账户所属 Amazon 组织中账户的资源共享，并将指定的 OU 添加为主体。该 OU 中的所有账户都可以使用资源共享中的资源。

```
aws ram create-resource-share \
```

```
    --name MyNewResourceShare \
    --no-allow-external-principals \
    --principals arn:aws:organizations::123456789012:ou/o-63bEXAMPLE/ou-46xi-
rEXAMPLE
```

输出：

```
{
    "resourceShare": {
        "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
        "name": "MyNewResourceShare",
        "owningAccountId": "123456789012",
        "allowExternalPrincipals": false,
        "status": "ACTIVE",
        "creationTime": 1634587042.49,
        "lastUpdatedTime": 1634587042.49
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResourceShare。

delete-resource-share

以下代码示例演示了如何使用 delete-resource-share。

Amazon CLI

删除资源共享

以下 delete-resource-share 示例删除了指定的资源共享。

```
aws ram delete-resource-share \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE
```

以下输出代表成功：

```
{
    "returnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourceShare。

disassociate-resource-share-permission

以下代码示例演示了如何使用 disassociate-resource-share-permission。

Amazon CLI

从资源共享中移除某个资源类型的 RAM 托管权限

以下 disassociate-resource-share-permission 示例从指定的资源共享中移除对 Glue 数据库的 RAM 托管权限。

```
aws ram disassociate-resource-share-permission \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \
    --permission-arn arn:aws:ram::aws:permission/AWSRAMPermissionGlueDatabaseReadWrite
```

输出：

```
{
    "returnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateResourceSharePermission。

disassociate-resource-share

以下代码示例演示了如何使用 disassociate-resource-share。

Amazon CLI

从资源共享中移除资源

以下 disassociate-resource-share 示例从指定的资源共享中移除指定的资源（在本例中为 VPC 子网）。任何有权访问资源共享的主体都无法再对该资源执行操作。

```
aws ram disassociate-resource-share \
```

```
    --resource-arns arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0250c25a1fEXAMPLE \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE
```

输出：

```
{
    "resourceShareAssociations": [
        "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
        "associatedEntity": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0250c25a1fEXAMPLE",
        "associationType": "RESOURCE",
        "status": "DISASSOCIATING",
        "external": false
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateResourceShare。

enable-sharing-with-aws-organization

以下代码示例演示了如何使用 enable-sharing-with-aws-organization。

Amazon CLI

启用跨 Amazon 组织的资源共享

以下 enable-sharing-with-aws-organization 示例启用了跨组织和组织单位的资源共享。

```
aws ram enable-sharing-with-aws-organization
```

以下输出代表成功。

```
{
    "returnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableSharingWithAwsOrganization。

get-permission

以下代码示例演示了如何使用 get-permission。

Amazon CLI

检索 RAM 托管权限的详细信息

以下 get-permission 示例显示指定 RAM 托管权限的默认版本的详细信息。

```
aws ram get-permission \
    --permission-arn arn:aws:ram::aws:permission/
AWSRAMPermissionGlueTableReadWriteForDatabase
```

输出：

```
{
    "permission": {
        "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueTableReadWriteForDatabase",
        "version": "2",
        "defaultVersion": true,
        "name": "AWSRAMPermissionGlueTableReadWriteForDatabase",
        "resourceType": "glue:Database",
        "permission": "{\"Effect\":\"Allow\",\"Action\":[\"glue:GetTable
\", \"glue:UpdateTable\", \"glue:DeleteTable\", \"glue:BatchDeleteTable\",
 \"glue:BatchDeleteTableVersion\", \"glue:GetTableVersion\", \"glue:GetTableVersions
\", \"glue:GetPartition\", \"glue:GetPartitions\", \"glue:BatchGetPartition\",
 \"glue:BatchCreatePartition\", \"glue:CreatePartition\", \"glue:UpdatePartition
\", \"glue:BatchDeletePartition\", \"glue:DeletePartition\", \"glue:GetTables\",
 \"glue:SearchTables\"]}",
        "creationTime": 1624912434.431,
        "lastUpdatedTime": 1624912434.431,
        "isResourceTypeDefault": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPermission。

get-resource-policies

以下代码示例演示了如何使用 get-resource-policies。

Amazon CLI

获取资源的策略

以下 get-resource-policies 示例显示了与资源共享关联的指定资源的基于资源的权限策略。

```
aws ram get-resource-policies \
    --resource-arns arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0250c25a1fEXAMPLE
```

输出：

```
{
    "policies": [
        "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"RamStatement1\",
\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[]},\"Action\":[\"ec2:RunInstances
\",\"ec2:CreateNetworkInterface\",\"ec2:DescribeSubnets\"],\"Resource\":
\"arn:aws:ec2:us-west-2:123456789012:subnet/subnet-0250c25a1fEXAMPLE\"}]}"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourcePolicies。

get-resource-share-associations

以下代码示例演示了如何使用 get-resource-share-associations。

Amazon CLI

示例 1：列出所有资源类型的所有资源关联

以下 get-resource-share-associations 示例列出了所有资源共享中所有资源类型的资源关联。

```
aws ram get-resource-share-associations \
    --association-type RESOURCE
```

输出：

```
{
    "resourceShareAssociations": [
```

```
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
            "associatedEntity": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0250c25a1fEXAMPLE",
            "resourceShareName": "MySubnetShare",
            "associationType": "RESOURCE",
            "status": "ASSOCIATED",
            "creationTime": 1565303590.973,
            "lastUpdatedTime": 1565303591.695,
            "external": false
        },
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/8167bdfe-4480-4a01-8632-315e0EXAMPLE",
            "associatedEntity": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
            "resourceShareName": "MyLicenseShare",
            "associationType": "RESOURCE",
            "status": "ASSOCIATED",
            "creationTime": 1632342958.457,
            "lastUpdatedTime": 1632342958.907,
            "external": false
        }
    ]
}
```

示例 2：列出资源共享的主体关联

以下 get-resource-share-associations 示例仅列出指定资源共享的主体关联。

```
aws ram get-resource-share-associations \
   --resource-share-arns arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE \
   --association-type PRINCIPAL
```

输出：

```
{
    "resourceShareAssociations": [
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
```

```
            "resourceShareName": "MyNewResourceShare",
            "associatedEntity": "arn:aws:organizations::123456789012:ou/
o-63bEXAMPLE/ou-46xi-rEXAMPLE",
            "associationType": "PRINCIPAL",
            "status": "ASSOCIATED",
            "creationTime": 1634587042.49,
            "lastUpdatedTime": 1634587044.291,
            "external": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceShareAssociations。

get-resource-share-invitations

以下代码示例演示了如何使用 get-resource-share-invitations。

Amazon CLI

列出您的资源共享邀请

以下 get-resource-share-invitations 示例列出了您当前的资源共享邀请。

```
aws ram get-resource-share-invitations
```

输出：

```
{
    "resourceShareInvitations": [
        {
            "resourceShareInvitationArn": "arn:aws:ram:us-
west2-1:111111111111:resource-share-invitation/32b639f0-14b8-7e8f-55ea-
e6117EXAMPLE",
            "resourceShareName": "project-resource-share",
            "resourceShareArn": "arn:aws:ram:us-west-2:111111111111:resource-share/
fcb639f0-1449-4744-35bc-a983fEXAMPLE",
            "senderAccountId": "111111111111",
            "receiverAccountId": "222222222222",
            "invitationTimestamp": 1565312166.258,
            "status": "PENDING"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceShareInvitations。

get-resource-shares

以下代码示例演示了如何使用 get-resource-shares。

Amazon CLI

示例 1：列出您拥有并与其他人共享的资源共享

以下 get-resource-shares 示例列出了您创建并正在与其他人共享的资源共享。

```
aws ram get-resource-shares \
    --resource-owner SELF
```

输出：

```
{
    "resourceShares": [
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/3ab63985-99d9-1cd2-7d24-75e93EXAMPLE",
            "name": "my-resource-share",
            "owningAccountId": "123456789012",
            "allowExternalPrincipals": false,
            "status": "ACTIVE",
            "tags": [
                {
                    "key": "project",
                    "value": "lima"
                }
            ]
            "creationTime": 1565295733.282,
            "lastUpdatedTime": 1565295733.282
        },
        {
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
            "name": "my-resource-share",
```

```
            "owningAccountId": "123456789012",
            "allowExternalPrincipals": true,
            "status": "ACTIVE",
            "creationTime": 1565295733.282,
            "lastUpdatedTime": 1565295733.282
        }
    ]
}
```

示例 2：列出其他人拥有并与您共享的资源共享

以下 get-resource-shares 示例列出了其他人创建并与您共享的资源共享。在此示例中为“无”。

```
aws ram get-resource-shares \
    --resource-owner OTHER-ACCOUNTS
```

输出：

```
{
    "resourceShares": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourceShares。

list-pending-invitation-resources

以下代码示例演示了如何使用 list-pending-invitation-resources。

Amazon CLI

列出待处理资源共享中可用的资源

以下 list-pending-invitation-resources 示例列出了与指定邀请关联的资源共享中的所有资源。

```
aws ram list-pending-invitation-resources \
    --resource-share-invitation-arn arn:aws:ram:us-west-2:123456789012:resource-share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE
```

输出：

```
{
   "resources": [
        {
            "arn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-04a555b0e6EXAMPLE",
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
            "creationTime": 1634676051.269,
            "lastUpdatedTime": 1634676052.07,
            "status": "AVAILABLE",
            "type": "ec2:Subnet"
        },
        {
            "arn": "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
            "creationTime": 1624912434.431,
            "lastUpdatedTime": 1624912434.431,
            "status": "AVAILABLE",
            "type": "license-manager:LicenseConfiguration"
      }
   ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPendingInvitationResources。

list-permissions

以下代码示例演示了如何使用 list-permissions。

Amazon CLI

列出可用的 RAM 托管权限

以下 list-permissions 示例列出了仅适用于 Amazon Glue 数据库资源类型的所有 RAM 托管权限。

```
aws ram list-permissions \
    --resource-type glue:Database
```

输出：


```
{
    "permissions": [
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMDefaultPermissionGlueDatabase",
            "version": "1",
            "defaultVersion": true,
            "name": "AWSRAMDefaultPermissionGlueDatabase",
            "resourceType": "glue:Database",
            "creationTime": 1592007820.935,
            "lastUpdatedTime": 1592007820.935,
            "isResourceTypeDefault": true
        },
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueAllTablesReadWriteForDatabase",
            "version": "2",
            "defaultVersion": true,
            "name": "AWSRAMPermissionGlueAllTablesReadWriteForDatabase",
            "resourceType": "glue:Database",
            "creationTime": 1624912413.323,
            "lastUpdatedTime": 1624912413.323,
            "isResourceTypeDefault": false
        },
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueDatabaseReadWrite",
            "version": "2",
            "defaultVersion": true,
            "name": "AWSRAMPermissionGlueDatabaseReadWrite",
            "resourceType": "glue:Database",
            "creationTime": 1624912417.4,
            "lastUpdatedTime": 1624912417.4,
            "isResourceTypeDefault": false
        },
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueTableReadWriteForDatabase",
            "version": "2",
            "defaultVersion": true,
            "name": "AWSRAMPermissionGlueTableReadWriteForDatabase",
            "resourceType": "glue:Database",
```

```
            "creationTime": 1624912434.431,
            "lastUpdatedTime": 1624912434.431,
            "isResourceTypeDefault": false
        }
    ]
}
```

以下 `list-permissions` 示例显示了所有资源类型的可用 RAM 托管权限。

```
aws ram list-permissions
```

输出：

```
{
    "permissions": [
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMBlankEndEntityCertificateAPICSRPassthroughIssuanceCertificateAuthority",
            "version": "1",
            "defaultVersion": true,
            "name":
 "AWSRAMBlankEndEntityCertificateAPICSRPassthroughIssuanceCertificateAuthority",
            "resourceType": "acm-pca:CertificateAuthority",
            "creationTime": 1623264861.085,
            "lastUpdatedTime": 1623264861.085,
            "isResourceTypeDefault": false
        },
        {
            "arn": "arn:aws:ram::aws:permission/AWSRAMDefaultPermissionAppMesh",
            "version": "1",
            "defaultVersion": true,
            "name": "AWSRAMDefaultPermissionAppMesh",
            "resourceType": "appmesh:Mesh",
            "creationTime": 1589307188.584,
            "lastUpdatedTime": 1589307188.584,
            "isResourceTypeDefault": true
        },
        ...TRUNCATED FOR BREVITY...
        {
            "arn": "arn:aws:ram::aws:permission/
AWSRAMSubordinateCACertificatePathLen0IssuanceCertificateAuthority",
            "version": "1",
            "defaultVersion": true,
```

```
            "name":
 "AWSRAMSubordinateCACertificatePathLen0IssuanceCertificateAuthority",
            "resourceType": "acm-pca:CertificateAuthority",
            "creationTime": 1623264876.75,
            "lastUpdatedTime": 1623264876.75,
            "isResourceTypeDefault": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPermissions。

`list-principals`

以下代码示例演示了如何使用 `list-principals`。

Amazon CLI

列出有权访问资源的主体

以下 `list-principals` 示例显示了可以通过任何资源共享访问指定类型资源的主体列表。

```
aws ram list-principals \
    --resource-type ec2:Subnet
```

输出：

```
{
    "principals": [
        {
            "id": "arn:aws:organizations::123456789012:ou/o-gx7EXAMPLE/ou-29c5-
zEXAMPLE",
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
            "creationTime": 1565298209.737,
            "lastUpdatedTime": 1565298211.019,
            "external": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPrincipals。

`list-resource-share-permissions`

以下代码示例演示了如何使用 `list-resource-share-permissions`。

Amazon CLI

列出当前附加到资源共享的所有 RAM 托管权限

以下 `list-resource-share-permissions` 示例列出了附加到指定资源共享的所有 RAM 托管权限。

```
aws ram list-resource-share-permissions \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE
```

输出：

```
{
    "permissions": [
        {
            "arn": "arn:aws:ram::aws:permission/AWSRAMDefaultPermissionLicenseConfiguration",
            "version": "1",
            "resourceType": "license-manager:LicenseConfiguration",
            "status": "ASSOCIATED",
            "lastUpdatedTime": 1632342984.234
        },
        {
            "arn": "arn:aws:ram::aws:permission/AWSRAMPermissionGlueDatabaseReadWrite",
            "version": "2",
            "resourceType": "glue:Database",
            "status": "ASSOCIATED",
            "lastUpdatedTime": 1632512462.297
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceSharePermissions。

list-resource-types

以下代码示例演示了如何使用 list-resource-types。

Amazon CLI

列出 Amazon RAM 支持的资源类型

以下 list-resource-types 示例列出了 Amazon RAM 当前支持的所有资源类型。

```
aws ram list-resource-types
```

输出：

```
{
    "resourceTypes": [
        {
            "resourceType": "route53resolver:FirewallRuleGroup",
            "serviceName": "route53resolver"
        },
        {
            "resourceType": "ec2:LocalGatewayRouteTable",
            "serviceName": "ec2"
        },
        ...OUTPUT TRUNCATED FOR BREVITY...
        {
            "resourceType": "ec2:Subnet",
            "serviceName": "ec2"
        },
        {
            "resourceType": "ec2:TransitGatewayMulticastDomain",
            "serviceName": "ec2"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceTypes。

list-resources

以下代码示例演示了如何使用 list-resources。

Amazon CLI

列出与资源共享关联的资源

以下 `list-resources` 示例列出了指定资源共享中属于指定资源类型的所有资源。

```
aws ram list-resources \
    --resource-type ec2:Subnet \
    --resource-owner SELF \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE
```

输出：

```
{
    "resources": [
        {
            "arn": "aarn:aws:ec2:us-west-2:123456789012:subnet/subnet-0250c25a1f4e15235",
            "type": "ec2:Subnet",
            "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
            "creationTime": 1565301545.023,
            "lastUpdatedTime": 1565301545.947
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResources。

promote-resource-share-created-from-policy

以下代码示例演示了如何使用 `promote-resource-share-created-from-policy`。

Amazon CLI

在 Amazon RAM 中将基于资源策略的资源共享提升到全部功能

以下 `promote-resource-share-created-from-policy` 示例采用您通过附加基于资源的策略隐式创建的资源共享，并通过 Amazon RAM 控制台及其 CLI 和 API 操作将其转换为功能完善的资源共享。

```
aws ram promote-resource-share-created-from-policy \
    --resource-share-arn arn:aws:ram:us-east-1:123456789012:resource-
share/91fa8429-2d06-4032-909a-90909EXAMPLE
```

输出：

```
{
    "returnValue": true
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PromoteResourceShareCreatedFromPolicy。

reject-resource-share-invitation

以下代码示例演示了如何使用 reject-resource-share-invitation。

Amazon CLI

拒绝资源共享邀请

以下 reject-resource-share-invitation 示例拒绝了指定的资源共享邀请。

```
aws ram reject-resource-share-invitation \
    --resource-share-invitation-arn arn:aws:ram:us-west-2:111111111111:resource-
share-invitation/32b639f0-14b8-7e8f-55ea-e6117EXAMPLE
```

输出：

```
"resourceShareInvitations": [
    {
        "resourceShareInvitationArn": "arn:aws:ram:us-west2-1:111111111111:resource-
share-invitation/32b639f0-14b8-7e8f-55ea-e6117EXAMPLE",
        "resourceShareName": "project-resource-share",
        "resourceShareArn": "arn:aws:ram:us-west-2:111111111111:resource-share/
fcb639f0-1449-4744-35bc-a983fEXAMPLE",
        "senderAccountId": "111111111111",
        "receiverAccountId": "222222222222",
        "invitationTimestamp": 1565319592.463,
        "status": "REJECTED"
    }
```

```
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectResourceShareInvitation。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为资源共享添加标签

以下 tag-resource 示例将标签键 project 和关联的值 lima 添加到指定的资源共享。

```
aws ram tag-resource \
    --tags key=project,value=lima \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源共享中移除标签

以下 untag-resource 示例从指定的资源共享中移除 project 标签键及关联值。

```
aws ram untag-resource \
    --tag-keys project \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-resource-share

以下代码示例演示了如何使用 update-resource-share。

Amazon CLI

更新资源共享

以下 update-resource-share 示例更改了指定的资源共享，以支持使用不属于 Amazon 组织的外部主体。

```
aws ram update-resource-share \
    --allow-external-principals \
    --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE
```

输出：

```
{
    "resourceShare": {
        "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
        "name": "my-resource-share",
        "owningAccountId": "123456789012",
        "allowExternalPrincipals": true,
        "status": "ACTIVE",
        "creationTime": 1565295733.282,
        "lastUpdatedTime": 1565303080.023
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResourceShare。

使用 Amazon CLI 的资源管理器示例

以下代码示例演示如何通过将 Amazon Command Line Interface与资源管理器结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-default-view`

以下代码示例演示了如何使用 `associate-default-view`。

Amazon CLI

将资源管理器视图设置为其 Amazon 区域的默认值

以下 `associate-default-view` 示例将视图（由其 ARN 指定）设置为调用该操作的 Amazon 区域的默认视图。

```
aws resource-explorer-2 associate-default-view \
    --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

输出：

```
{
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的在 Amazon 区域中设置默认视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateDefaultView。

`batch-get-view`

以下代码示例演示了如何使用 `batch-get-view`。

Amazon CLI

检索有关多个资源管理器视图的详细信息

以下 batch-get-view 示例显示了两个视图（由其 ARN 指定）的详细信息。使用空格分隔 --view-arn 参数中的多个 ARN。

```
aws resource-explorer-2 batch-get-view \
    --view-arns arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222, \
                arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

输出：

```
{
    "Views": [
        {
            "Filters": {
                "FilterString": "service:ec2"
            },
            "IncludedProperties": [
                {
                    "Name": "tags"
                }
            ],
            "LastUpdatedAt": "2022-07-13T21:33:45.249000+00:00",
            "Owner": "123456789012",
            "Scope": "arn:aws:iam::123456789012:root",
            "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
        },
        {
            "Filters": {
                "FilterString": ""
            },
            "IncludedProperties": [
                {
                    "Name": "tags"
                }
            ],
            "LastUpdatedAt": "2022-07-13T20:34:11.314000+00:00",
            "Owner": "123456789012",
            "Scope": "arn:aws:iam::123456789012:root",
            "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
        }
```

```
    ]
    "Errors": []
}
```

有关视图的更多信息，请参阅《Amazon 资源管理器用户指南》中的关于资源管理器视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetView。

create-index

以下代码示例演示了如何使用 create-index。

Amazon CLI

通过创建索引在 Amazon 区域中启用资源管理器

以下 create-index 示例在调用操作的 Amazon 区域中创建了本地索引。如果您未指定值，Amazon CLI 会自动生成一个随机 client-token 参数值，并将其包含在对 Amazon 的调用中。

```
aws resource-explorer-2 create-index \
    --region us-east-1
```

输出：

```
{
    "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE22222c",
    "CreatedAt": "2022-11-01T20:00:59.149Z",
    "State": "CREATING"
}
```

创建本地索引后，您可以通过运行 update-index-type 命令将其转换为账户的聚合索引。

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的在 Amazon 区域中启用资源管理器以索引您的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIndex。

create-view

以下代码示例演示了如何使用 create-view。

Amazon CLI

示例 1：为 Amazon 区域中的索引创建未经筛选的视图

以下 `create-view` 示例在指定的 Amazon 区域中创建了一个视图，该视图不经过任何筛选即可返回该区域中的所有结果。该视图在返回的结果中包含可选的“标签”字段。由于此视图是在包含聚合器索引的区域中创建的，因此它可以包括账户中包含资源管理器索引的所有区域中的结果。

```
aws resource-explorer-2 create-view \
    --view-name My-Main-View \
    --included-properties Name=tags \
    --region us-east-1
```

输出：

```
{
    "View": {
        "Filters": {
            "FilterString": ""
        },
        "IncludedProperties": [
            {
                "Name": "tags"
            }
        ],
        "LastUpdatedAt": "2022-07-13T20:34:11.314000+00:00",
        "Owner": "123456789012",
        "Scope": "arn:aws:iam::123456789012:root",
        "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
    }
}
```

示例 2：创建仅返回与 Amazon EC2 关联的资源的视图

以下 `create-view` 在 Amazon 区域 `us-east-1` 中创建了一个视图，该视图仅返回该区域中与 Amazon EC2 服务关联的资源。该视图在返回的结果中包含可选的 `Tags` 字段。由于此视图是在包含聚合器索引的区域中创建的，因此它可以包括账户中包含资源管理器索引的所有区域中的结果。

```
aws resource-explorer-2 create-view \
    --view-name My-EC2-Only-View \
```

```
    --included-properties Name=tags \
    --filters FilterString="service:ec2" \
    --region us-east-1
```

输出：

```
{
    "View": {
        "Filters": {
            "FilterString": "service:ec2"
        },
        "IncludedProperties": [
            {
                "Name":"tags"
            }
        ],
        "LastUpdatedAt": "2022-07-13T21:35:09.059Z",
        "Owner": "123456789012",
        "Scope": "arn:aws:iam::123456789012:root",
        "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-
Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
    }
}
```

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的创建用于搜索的视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateView。

delete-index

以下代码示例演示了如何使用 delete-index。

Amazon CLI

关闭 Amazon 区域中的资源管理器

以下 delete-index 示例删除了您发出请求的 Amazon 区域中指定的资源管理器索引。

```
aws resource-explorer-2 delete-index \
    --arn arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE22222 \
    --region us-west-2
```

输出：

```
{
    "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222",
    "State": "DELETING"
}
```

有关删除索引的更多信息，请参阅《Amazon 资源管理器用户指南》中的在 Amazon 区域中关闭 Amazon 资源管理器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIndex。

delete-view

以下代码示例演示了如何使用 delete-view。

Amazon CLI

删除资源管理器视图

以下 delete-view 示例删除了一个视图（由其 ARN 指定）。

```
aws resource-explorer-2 delete-view \
    --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

输出：

```
{
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的删除视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteView。

disassociate-default-view

以下代码示例演示了如何使用 disassociate-default-view。

Amazon CLI

删除 Amazon 区域的默认资源管理器视图

以下 disassociate-default-view 删除了您调用操作的 Amazon 区域的默认资源管理器视图。执行此操作后，区域中的所有搜索操作都必须明确指定视图，否则操作将失败。

```
aws resource-explorer-2 disassociate-default-view
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的在 Amazon 区域中设置默认视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateDefaultView。

get-default-view

以下代码示例演示了如何使用 get-default-view。

Amazon CLI

检索资源管理器视图，该视图是其 Amazon 区域的默认视图

以下 get-default-view 示例将检索您调用操作的 Amazon 区域的默认视图的 ARN。

```
aws resource-explorer-2 get-default-view
```

输出：

```
{
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/default-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的在 Amazon 区域中设置默认视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDefaultView。

get-index

以下代码示例演示了如何使用 get-index。

Amazon CLI

示例 1：检索资源管理器聚合器索引的详细信息

以下 get-index 示例显示了指定 Amazon 区域中资源管理器索引的详细信息。由于指定的区域包含账户的聚合器索引，因此输出会列出将数据复制到该区域索引中的区域。

```
aws resource-explorer-2 get-index \
    --region us-east-1
```

输出：

```
{
    "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE11111",
    "CreatedAt": "2022-07-12T18:59:10.503000+00:00",
    "LastUpdatedAt": "2022-07-13T18:41:58.799000+00:00",
    "ReplicatingFrom": [
        "ap-south-1",
        "us-west-2"
    ],
    "State": "ACTIVE",
    "Tags": {},
    "Type": "AGGREGATOR"
}
```

示例 2：检索资源管理器本地索引的详细信息

以下 get-index 示例显示了指定 Amazon 区域中资源管理器索引的详细信息。由于指定的区域包含本地索引，因此输出会列出将该区域索引中的数据复制到的区域。

```
aws resource-explorer-2 get-index \
    --region us-west-2
```

输出：

```
{
    "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE22222",
    "CreatedAt": "2022-07-12T18:59:10.503000+00:00",
    "LastUpdatedAt": "2022-07-13T18:41:58.799000+00:00",
    "ReplicatingTo": [
```

```
        "us-west-2"
    ],
    "State": "ACTIVE",
    "Tags": {},
    "Type": "LOCAL"
}
```

有关索引的更多信息，请参阅《Amazon 资源管理器用户指南》中的检查哪些 Amazon 区域开启了资源管理器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIndex。

get-view

以下代码示例演示了如何使用 get-view。

Amazon CLI

检索有关资源管理器视图的详细信息

以下 get-view 示例显示了一个视图（由其 ARN 指定）的详细信息。

```
aws resource-explorer-2 get-view \
    --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

输出：

```
{
  "Tags" : {},
  "View" : {
        "Filters" : {
            "FilterString" : "service:ec2"
        },
        "IncludedProperties" : [
            {
                "Name" : "tags"
            }
        ],
        "LastUpdatedAt" : "2022-07-13T21:33:45.249Z",
        "Owner" : "123456789012",
        "Scope" : "arn:aws:iam::123456789012:root",
```

```
        "ViewArn" : "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-
Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
  }
}
```

有关视图的更多信息，请参阅《Amazon 资源管理器用户指南》中的关于资源管理器视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetView。

`list-indexes`

以下代码示例演示了如何使用 `list-indexes`。

Amazon CLI

列出资源管理器中包含索引的 Amazon 区域

以下 `list-indexes` 示例列出了资源管理器具有索引的所有区域的索引。响应指定了每个索引的类型、其 Amazon 区域及其 ARN。

```
aws resource-explorer-2 list-indexes
```

输出：

```
{
    "Indexes": [
        {
            "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111",
            "Region": "us-west-2",
            "Type": "AGGREGATOR"
        },
        {
            "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222",
            "Region": "us-east-1",
            "Type": "LOCAL"
        },
        {
            "Arn": "arn:aws:resource-explorer-2:us-east-2:123456789012:index/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE33333",
            "Region": "us-east-2",
```

```
            "Type": "LOCAL"
        },
        {
            "Arn": "arn:aws:resource-explorer-2:us-west-1:123456789012:index/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE44444",
            "Region": "us-west-1",
            "Type": "LOCAL"
        }
    ]
}
```

有关索引的更多信息，请参阅《Amazon 资源管理器用户指南》中的检查哪些 Amazon 区域开启了资源管理器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIndexes。

`list-supported-resource-types`

以下代码示例演示了如何使用 `list-supported-resource-types`。

Amazon CLI

列出资源管理器中包含索引的 Amazon 区域

以下 `list-supported-resource-types` 示例列出了 &AREXlong; 当前支持的所有资源类型。示例响应中包含一个 `NextToken` 值，该值表示有更多输出可供通过其他调用进行检索。

```
aws resource-explorer-2 list-supported-resource-types \
    --max-items 10
```

输出：

```
{
    "ResourceTypes": [
        {
            "ResourceType": "cloudfront:cache-policy",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudfront:distribution",
            "Service": "cloudfront"
        },
```

```
        {
            "ResourceType": "cloudfront:function",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudfront:origin-access-identity",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudfront:origin-request-policy",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudfront:realtime-log-config",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudfront:response-headers-policy",
            "Service": "cloudfront"
        },
        {
            "ResourceType": "cloudwatch:alarm",
            "Service": "cloudwatch"
        },
        {
            "ResourceType": "cloudwatch:dashboard",
            "Service": "cloudwatch"
        },
        {
            "ResourceType": "cloudwatch:insight-rule",
            "Service": "cloudwatch"
        }
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0="
}
```

要获取输出的下一部分，请再次调用该操作，并将上一个调用的 NextToken 响应值作为 --starting-token 的值传递。重复此操作，直到响应中不存在 NextToken。

```
aws resource-explorer-2 list-supported-resource-types \
    --max-items 10  \
    --starting-
token eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0=
```

输出：

```
{
    "ResourceTypes": [
        {
            "ResourceType": "cloudwatch:metric-stream",
            "Service": "cloudwatch"
        },
        {
            "ResourceType": "dynamodb:table",
            "Service": "dynamodb"
        },
        {
            "ResourceType": "ec2:capacity-reservation",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:capacity-reservation-fleet",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:client-vpn-endpoint",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:customer-gateway",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:dedicated-host",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:dhcp-options",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:egress-only-internet-gateway",
            "Service": "ec2"
        },
        {
            "ResourceType": "ec2:elastic-gpu",
            "Service": "ec2"
        }
```

```
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyMH0="
}
```

有关索引的更多信息，请参阅《Amazon 资源管理器用户指南》中的检查哪些 Amazon 区域开启了资源管理器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSupportedResourceTypes。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出附加到资源管理器视图或索引的标签

以下 `list-tags-for-resource` 示例列出了附加到具有指定 ARN 的视图的标签键和值对。您必须从包含资源的 Amazon 区域调用操作。

```
aws resource-explorer-2 list-tags-for-resource \
    --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

输出：

```
{
    "Tags": {
        "application": "MainCorpApp",
        "department": "1234"
    }
}
```

有关为视图添加标签的更多信息，请参阅《Amazon 资源管理器用户指南》中的为视图添加标签以实现访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-views`

以下代码示例演示了如何使用 `list-views`。

Amazon CLI

列出 Amazon 区域中可用的资源管理器视图

以下 `list-views` 示例列出了您在其中调用操作的区域中可用的所有视图。

```
aws resource-explorer-2 list-views
```

输出：

```
{
    "Views": [
        "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111",
        "arn:aws:resource-explorer-2:us-east-1:123456789012:view/Default-All-
Resources-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222",
        "arn:aws:resource-explorer-2:us-east-1:123456789012:view/Production-Only-
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE33333"
    ]
}
```

有关视图的更多信息，请参阅《Amazon 资源管理器用户指南》中的关于资源管理器视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListViews。

`search`

以下代码示例演示了如何使用 `search`。

Amazon CLI

示例 1：使用默认视图进行搜索

以下 `search` 示例显示了与服务关联的指定视图中的所有资源。该搜索使用该区域的默认视图。示例响应中包含一个 `NextToken` 值，该值表示有更多输出可供通过其他调用进行检索。

```
aws resource-explorer-2 search \
    --query-string "service:iam"
```

输出：

```
{
```

```
    "Count": {
        "Complete": true,
        "TotalResources": 55
    },
    "NextToken":
 "AG9VOEF1KLEXAMPLEOhJHVwo5chEXAMPLER5XiEpNrgsEXAMPLE...b0CmOFOryHEXAMPLE",
    "Resources": [{
        "Arn": "arn:aws:iam::123456789012:policy/service-role/Some-Policy-For-A-
Service-Role",
        "LastReportedAt": "2022-07-21T12:34:42Z",
        "OwningAccountId": "123456789012",
        "Properties": [],
        "Region": "global",
        "ResourceType": "iam:policy",
        "Service": "iam"
    }, {
        "Arn": "arn:aws:iam::123456789012:policy/service-role/Another-Policy-For-A-
Service-Role",
        "LastReportedAt": "2022-07-21T12:34:42Z",
        "OwningAccountId": "123456789012",
        "Properties": [],
        "Region": "global",
        "ResourceType": "iam:policy",
        "Service": "iam"
    }, {
       ... TRUNCATED FOR BREVITY ...
    }],
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/my-default-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
```

示例 2：使用指定视图进行搜索

以下 search 示例搜索显示了指定 Amazon 区域中通过指定视图可见的所有资源（“*”）。由于视图附带筛选条件，结果仅包含与 Amazon EC2 关联的资源。

```
aws resource-explorer-2 search \
    -- query-string "*" \
    -- view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-view/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222
```

输出：

```
HTTP/1.1 200 OK
Date: Tue, 01 Nov 2022 20:00:59 GMT
Content-Type: application/json
Content-Length: <PayloadSizeBytes>

    {
        "Count": {
            "Complete": true,
            "TotalResources": 67
        },
        "Resources": [{
            "Arn": "arn:aws:ec2:us-east-1:123456789012:network-acl/acl-1a2b3c4d",
            "LastReportedAt": "2022-07-21T18:52:02Z",
            "OwningAccountId": "123456789012",
            "Properties": [{
                "Data": [{
                    "Key": "Department",
                    "Value": "AppDevelopment"
                }, {
                    "Key": "Environment",
                    "Value": "Production"
                }],
                "LastReportedAt": "2021-11-15T14:48:29Z",
                "Name": "tags"
            }],
            "Region": "us-east-1",
            "ResourceType": "ec2:network-acl",
            "Service": "ec2"
        }, {
            "Arn": "arn:aws:ec2:us-east-1:123456789012:subnet/subnet-1a2b3c4d",
            "LastReportedAt": "2022-07-21T21:22:23Z",
            "OwningAccountId": "123456789012",
            "Properties": [{
                "Data": [{
                    "Key": "Department",
                    "Value": "AppDevelopment"
                }, {
                    "Key": "Environment",
                    "Value": "Production"
                }],
                "LastReportedAt": "2021-07-29T19:02:39Z",
                "Name": "tags"
            }],
```

```
            "Region": "us-east-1",
            "ResourceType": "ec2:subnet",
            "Service": "ec2"
        }, {
            "Arn": "arn:aws:ec2:us-east-1:123456789012:dhcp-options/dopt-1a2b3c4d",
            "LastReportedAt": "2022-07-21T06:08:53Z",
            "OwningAccountId": "123456789012",
            "Properties": [{
                "Data": [{
                    "Key": "Department",
                    "Value": "AppDevelopment"
                }, {
                    "Key": "Environment",
                    "Value": "Production"
                }],
                "LastReportedAt": "2021-11-15T15:11:05Z",
                "Name": "tags"
            }],
            "Region": "us-east-1",
            "ResourceType": "ec2:dhcpoptions",
            "Service": "ec2"
        }, {
            ... TRUNCATED FOR BREVITY ...
        }],
        "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
    }
```

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的使用 Amazon 资源管理器搜索资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Search。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

为资源管理器视图添加标签

以下 `tag-resource` 示例将值为“production”的标签键“environment”添加到具有指定 ARN 的视图中。

```
aws resource-explorer-2 tag-resource \
    --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View//
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \
    --tags environment=production
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的为视图添加标签以实现访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源管理器视图中删除标签

以下 untag-resource 示例从具有指定 ARN 的视图中删除键名为“environment”的所有标签。

```
aws resource-explorer-2 untag-resource \
    --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View//
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \
    --tag-keys environment
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 资源管理器用户指南》中的为视图添加标签以实现访问控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-index-type

以下代码示例演示了如何使用 update-index-type。

Amazon CLI

更改资源管理器索引的类型

以下 update-index-type 示例将指定的索引从类型 local 转换为类型 aggregator，以便能够在该账户的所有 Amazon 区域中搜索资源。您必须将请求发送到包含要更新的索引的 Amazon 区域。

```
aws resource-explorer-2 update-index-type \
    --arn arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \
    --type aggregator \
    --region us-east-1
```

输出：

```
{
    "Arn":"arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111",
    "LastUpdatedAt":"2022-07-13T18:41:58.799Z",
    "State":"updating",
    "Type":"aggregator"
}
```

有关更改索引类型的更多信息，请参阅《Amazon 资源管理器用户指南》中的通过创建聚合器索引启用跨区域搜索。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIndexType。

update-view

以下代码示例演示了如何使用 update-view。

Amazon CLI

示例 1：更新资源管理器视图的 IncludedProperties 字段

以下 update-view 示例通过将 `tags` 添加到可选 `IncludedProperties` 来更新指定的视图。运行此操作后，使用此视图的搜索操作将包含有关结果中显示的资源所附标签的信息。

```
aws resource-explorer-2 update-view \
    --included-properties Name=tags \
    --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222
```

输出：

```
{
```

```
    "View": {
        "Filters": {
            "FilterString": ""
        },
        "IncludedProperties": [
            {
                "Name": "tags"
            }
        ],
        "LastUpdatedAt": "2022-07-19T17:41:21.710000+00:00",
        "Owner": "123456789012",
        "Scope": "arn:aws:iam::123456789012:root",
        "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-
Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
    }
}
```

示例 2：更新附加到视图的筛选条件

以下 update-view 示例将指定视图更新为使用筛选条件，该筛选条件将结果仅限于与 Amazon EC2 服务关联的资源类型。

```
aws resource-explorer-2 update-view \
    --filters FilterString="service:ec2" \
    --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222
```

输出：

```
{
    "View": {
    "Filters": {
        "FilterString": "service:ec2"
    },
    "IncludedProperties": [],
    "LastUpdatedAt": "2022-07-19T17:41:21.710000+00:00",
        "Owner": "123456789012",
        "Scope": "arn:aws:iam::123456789012:root",
        "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
    }
}
```

有关视图的更多信息，请参阅《Amazon 资源管理器用户指南》中的关于资源管理器视图。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateView。

使用 Amazon CLI 的资源组示例

以下代码示例演示如何通过将 Amazon Command Line Interface与资源组结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-group`

以下代码示例演示了如何使用 `create-group`。

Amazon CLI

示例 1：创建基于标签的资源组

以下 `create-group` 示例在当前区域中创建 Amazon EC2 实例的基于标签的资源组。它基于使用键 `Name` 和值 `WebServers` 标记的资源的查询。日志组为 `tbq-WebServer`。该查询在传递给命令的单独 JSON 文件中。

```
aws resource-groups create-group \
    --name tbq-WebServer \
    --resource-query file://query.json
```

的内容`query.json`：

```
{
    "Type": "TAG_FILTERS_1_0",
```

```
    "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":
[{\"Key\":\"Name\", \"Values\":[\"WebServers\"]}]}"
}
```

输出：

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer",
        "Name": "tbq-WebServer"
    },
    "ResourceQuery": {
        "Type": "TAG_FILTERS_1_0",
        "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":
[{\"Key\":\"Name\", \"Values\":[\"WebServers\"]}]}"
    }
}
```

示例 2：创建基于 CloudFormation 堆栈的资源组

以下 `create-group` 示例创建了一个名为 `sampleCFNstackgroup` 的基于 Amazon CloudFormation 堆栈的资源组。该查询包括 Amazon Resource Groups 支持的指定 CloudFormation 堆栈中的所有资源。

```
aws resource-groups create-group \
    --name cbq-CFNstackgroup \
    --resource-query file://query.json
```

的内容`query.json`：

```
{
    "Type": "CLOUDFORMATION_STACK_1_0",
    "Query": "{\"ResourceTypeFilters\":[\"AWS::AllSupported\"],\"StackIdentifier
\":\"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-
z39z-11z8-97z5-500z212zz6fz\"}"
}
```

输出：

```
{
```

```
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-
CFNstackgroup",
        "Name": "cbq-CFNstackgroup"
    },
    "ResourceQuery": {
        "Type": "CLOUDFORMATION_STACK_1_0",
        "Query": "{\"ResourceTypeFilters\":[\"AWS::AllSupported\"],\"StackIdentifier
\":\"arn:aws:cloudformation:us-east-2:123456789012:stack/MyCFNStack/1415z9z0-
z39z-11z8-97z5-500z212zz6fz\"}"}'
    }
}
```

有关更多信息，请参阅《Amazon 用户指南》中的创建组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

更新资源组的描述

以下 delete-group 示例更新了指定资源组。

```
aws resource-groups delete-group \
    --group-name tbq-WebServer
```

输出：

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-west-2:1234567890:group/tbq-
WebServer",
        "Name": "tbq-WebServer"
    }
}
```

有关更多信息，请参阅《Amazon 用户指南》中的删除组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

get-group-query

以下代码示例演示了如何使用 get-group-query。

Amazon CLI

将查询附加到资源组

以下 get-group-query 示例显示了附加到指定资源组的查询。

```
aws resource-groups get-group-query \
    --group-name tbq-WebServer
```

输出：

```
{
    "GroupQuery": {
        "GroupName": "tbq-WebServer",
        "ResourceQuery": {
            "Type": "TAG_FILTERS_1_0",
            "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters
\":[{\"Key\":\"Name\", \"Values\":[\"WebServers\"]}]}"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroupQuery。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

获取有关资源组的信息

以下 get-group 示例显示了有关指定资源组的详细信息。要将查询附加到群组，请使用 get-group-query。

```
aws resource-groups get-group \
    --group-name tbq-WebServer
```

输出：

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer",
        "Name": "tbq-WebServer",
        "Description": "A tag-based query resource group of WebServers."
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

get-tags

以下代码示例演示了如何使用 get-tags。

Amazon CLI

检索附加到资源组的标签

以下 get-tags 示例显示了附加到指定资源组（组本身，而不是其成员）的标签键和值对。

```
aws resource-groups get-tags \
    --arn arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer
```

输出：

```
{
    "Arn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",
    "Tags": {
        "QueryType": "tags",
        "QueryResources": "ec2-instances"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTags。

list-group-resources

以下代码示例演示了如何使用 list-group-resources。

Amazon CLI

列出资源组中的所有资源

示例 1：以下 list-resource-groups 示例列出了属于指定资源组的所有资源。

```
aws resource-groups list-group-resources \
    --group-name tbq-WebServer
```

输出：

```
{
    "ResourceIdentifiers": [
        {
            "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-09f77fa38c12345ab",
            "ResourceType": "AWS::EC2::Instance"
        }
    ]
}
```

示例 2：以下示例列出了组中“resource-type”也为“Amazon::EC2::Instance”的所有资源。

aws resource-groups list-group-resources --group-name tbq-WebServer --filters Name=resource-type,Values=Amazon::EC2::Instance

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupResources。

list-groups

以下代码示例演示了如何使用 list-groups。

Amazon CLI

列出可用的资源组

以下 list-groups 示例显示了所有资源组的列表。

```
aws resource-groups list-groups
```

输出：

```
{
    "GroupIdentifiers": [
        {
            "GroupName": "tbq-WebServer",
            "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer3"
        },
        {
            "GroupName": "cbq-CFNStackQuery",
            "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-
CFNStackQuery"
        }
    ],
    "Groups": [
        {
            "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer",
            "Name": "tbq-WebServer"
        },
        {
            "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-
CFNStackQuery",
            "Name": "cbq-CFNStackQuery"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

list-resource-groups

以下代码示例演示了如何使用 list-resource-groups。

Amazon CLI

列出资源组中的所有资源

以下 list-resource-groups 示例列出了属于指定资源组的所有资源。

```
aws resource-groups list-group-resources \
    --group-name tbq-WebServer
```

输出：

```
{
    "ResourceIdentifiers": [
        {
            "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-09f77fa38c12345ab",
            "ResourceType": "AWS::EC2::Instance"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceGroups。

put-group-configuration

以下代码示例演示了如何使用 put-group-configuration。

Amazon CLI

将服务配置附加到资源组

示例 1：以下 put-group-configuration 示例指定资源组仅包含 C5 或 M5 系列中实例的 Amazon EC2 容量预留。

```
aws resource-groups put-group-configuration \
    --group MyTestGroup \
    --configuration file://config.json
```

的内容config.json：

```
[
    {
        "Type": "AWS::EC2::HostManagement",
        "Parameters": [
            {
                "Name": "allowed-host-families",
```

```
                "Values": [ "c5", "m5" ]
            },
                {
                    "Name": "any-host-based-license-configuration",
                    "Values": [ "true" ]
                }
        ]
    },
    {
        "Type": "AWS::ResourceGroups::Generic",
        "Parameters": [
            {
                "Name": "allowed-resource-types",
                "Values": [ "AWS::EC2::Host" ]
            },
            {
                "Name": "deletion-protection",
                "Values": [ "UNLESS_EMPTY" ]
            }
        ]
    }
]
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《资源组 API 参考指南》中的资源组的服务配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutGroupConfiguration。

`search-resources`

以下代码示例演示了如何使用 `search-resources`。

Amazon CLI

查找与查询相匹配的资源

以下 `search-resources` 示例检索与指定查询匹配的所有 Amazon 资源的列表。

```
aws resource-groups search-resources \
    --resource-query file://query.json
```

的内容`query.json`：

```
{
    "Type": "TAG_FILTERS_1_0",
    "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":
[{\"Key\":\"Patch Group\", \"Values\":[\"Dev\"]}]}"
}
```

输出：

```
{
    "ResourceIdentifiers": [
        {
            "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-01a23bc45d67890ef",
            "ResourceType": "AWS::EC2::Instance"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchResources。

tag

以下代码示例演示了如何使用 tag。

Amazon CLI

将标签附加到资源组

以下 tag 示例将指定的标签键和值对附加到指定的资源组（组本身，而不是其成员）。

```
aws resource-groups tag \
    --tags QueryType=tags,QueryResources=ec2-instances \
    --arn arn:aws:resource-groups:us-west-2:128716708097:group/tbq-WebServer
```

输出：

```
{
    "Arn": "arn:aws:resource-groups:us-west-2:128716708097:group/tbq-WebServer",
    "Tags": {
        "QueryType": "tags",
        "QueryResources": "ec2-instances"
```

```
    }
}
```

有关更多信息，请参阅《Amazon 资源组用户指南》中的管理标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Tag。

untag

以下代码示例演示了如何使用 untag。

Amazon CLI

从资源中删除标签

以下 untags 示例从资源组（而非其成员）中删除具有指定键的所有标签。

```
aws resource-groups untag \
    --arn arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer \
    --keys QueryType
```

输出：

```
{
    "Arn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",
    "Keys": [
        "QueryType"
    ]
}
```

有关更多信息，请参阅《Amazon 资源组用户指南》中的管理标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Untag。

update-group-query

以下代码示例演示了如何使用 update-group-query。

Amazon CLI

示例 1：更新基于标签的资源组的查询

以下 update-group-query 示例更新了附加到基于指定标签的资源组的查询。

```
aws resource-groups update-group-query \
    --group-name tbq-WebServer \
    --resource-query '{"Type":"TAG_FILTERS_1_0", "Query":"{\"ResourceTypeFilters\":
[\"AWS::EC2::Instance\"],\"TagFilters\":[{\"Key\":\"Name\", \"Values\":[\"WebServers
\"]}]}"}'
```

输出：

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-east-2:123456789012:group/tbq-
WebServer",
        "Name": "tbq-WebServer"
    },
    "ResourceQuery": {
        "Type": "TAG_FILTERS_1_0",
        "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":
[{\"Key\":\"Name\", \"Values\":[\"WebServers\"]}]}"
    }
}
```

有关更多信息，请参阅《Amazon 资源组用户指南》中的更新组。

示例 2：更新基于 CloudFormation 堆栈的资源组的查询

以下 update-group-query 示例更新了附加到基于指定 Amazon CloudFormation 堆栈的资源组的查询。

```
aws resource-groups update-group-query \
    --group-name cbq-CFNstackgroup \
    --resource-query '{"Type": "CLOUDFORMATION_STACK_1_0", "Query":
 "{\"ResourceTypeFilters\":[\"AWS::AllSupported\"],\"StackIdentifier\":
\"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-
z39z-11z8-97z5-500z212zz6fz\"}"}'
```

输出：

```
{
    "Group": {
```

```
        "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-
CFNstackgroup",
        "Name": "cbq-CFNstackgroup"
    },
    "ResourceQuery": {
        "Type": "CLOUDFORMATION_STACK_1_0",
        "Query": "{\"ResourceTypeFilters\":[\"AWS::AllSupported\"],\"StackIdentifier
\":\"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-
z39z-11z8-97z5-500z212zz6fz\"}"
    }
}
```

有关更多信息，请参阅《Amazon 资源组用户指南》中的更新组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroupQuery。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

更新资源组的描述

以下 update-group 示例更新指定资源组的描述。

```
aws resource-groups update-group \
    --group-name tbq-WebServer \
    --description "Resource group for all web server resources."
```

输出：

```
{
    "Group": {
        "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer",
        "Name": "tbq-WebServer"
        "Description": "Resource group for all web server resources."
    }
}
```

有关更多信息，请参阅《Amazon 资源组用户指南》中的更新组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

使用 Amazon CLI 的资源组标记 API

以下代码示例演示如何通过将 Amazon Command Line Interface与资源组标记 API 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`get-resources`

以下代码示例演示了如何使用 `get-resources`。

Amazon CLI

获取已标记资源的列表

以下 `get-resources` 示例显示账户中使用指定键名和值标记的资源的列表。

```
aws resourcegroupstaggingapi get-resources \
    --tag-filters Key=Environment,Values=Production \
    --tags-per-page 100
```

输出：

```
{
    "ResourceTagMappingList": [
        {
            "ResourceARN": " arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX/template/0-7sbz2Kz0",
            "Tags": [
```

```
                {
                    "Key": "Environment",
                    "Value": "Production"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《资源组标记 API Reference》中的 GetResources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResources。

get-tag-keys

以下代码示例演示了如何使用 get-tag-keys。

Amazon CLI

获取所有标签键的列表

以下 get-tag-keys 示例检索账户中资源使用的所有标签键名的列表。

```
aws resourcegroupstaggingapi get-tag-keys
```

输出：

```
{
    "TagKeys": [
        "Environment",
        "CostCenter",
        "Department"
    ]
}
```

有关更多信息，请参阅《资源组标记 API Reference》中的 GetTagKeys。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTagKeys。

get-tag-values

以下代码示例演示了如何使用 get-tag-values。

Amazon CLI

获取所有标签值的列表

以下 `get-tag-values` 示例显示了用于账户中所有资源的指定键的所有值

```
aws resourcegroupstaggingapi get-tag-values \
    --key=Environment
```

输出：

```
{
    "TagValues": [
        "Alpha",
        "Gamma",
        "Production"
    ]
}
```

有关更多信息，请参阅《资源组标记 API Reference》中的 GetTagValues。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTagValues。

`tag-resources`

以下代码示例演示了如何使用 `tag-resources`。

Amazon CLI

将标签添加到资源中

以下 `tag-resources` 示例向指定资源添加带有键名和值的标签。

```
aws resourcegroupstaggingapi tag-resources \
    --resource-arn-list arn:aws:s3:::MyProductionBucket \
    --tags Environment=Production,CostCenter=1234
```

输出：

```
{
    "FailedResourcesMap": {}
}
```

有关更多信息，请参阅《资源组标记 API Reference》中的 TagResources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResources。

`untag-resources`

以下代码示例演示了如何使用 `untag-resources`。

Amazon CLI

从资源中删除标签

以下 `untag-resources` 示例将从指定资源中删除指定的标签键及其关联的值。

```
aws resourcegroupstaggingapi untag-resources \
    --resource-arn-list arn:aws:s3:::amzn-s3-demo-bucket \
    --tag-keys Environment CostCenter
```

输出：

```
{
    "FailedResourcesMap": {}
}
```

有关更多信息，请参阅《资源组标记 API Reference》中的 UntagResources。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResources。

使用 Amazon CLI 的 Route 53 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Route 53 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

change-resource-record-sets

以下代码示例演示了如何使用 change-resource-record-sets。

Amazon CLI

创建、更新或删除资源记录集

以下 change-resource-record-sets 命令使用 hosted-zone-id Z1R8UBAEXAMPLE 以及 C:\awscli\route53\change-resource-record-sets.json 文件中的 JSON 格式配置创建资源记录集：

```
aws route53 change-resource-record-sets --hosted-zone-id Z1R8UBAEXAMPLE --change-batch file://C:\awscli\route53\change-resource-record-sets.json
```

有关更多信息，请参阅《Amazon Route 53 API Reference》中的“POST ChangeResourceRecordSets”。

JSON 文件中的配置取决于要创建的资源记录集的类型：

BasicWeightedAliasWeighted AliasLatencyLatency AliasFailoverFailover Alias

基本语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
```

```
        ]
      }
    },
    {...}
  ]
}
```

加权语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Weight": value between 0 and 255,
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ],
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

别名语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
```

```
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
 S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
 bucket, Elastic Load Balancing load balancer, or another resource record set in
 this hosted zone",
          "EvaluateTargetHealth": true|false
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

加权别名语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Weight": value between 0 and 255,
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
 S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
 bucket, Elastic Load Balancing load balancer, or another resource record set in
 this hosted zone",
          "EvaluateTargetHealth": true|false
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

延迟语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Region": "Amazon EC2 region name",
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ],
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

延迟别名语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Region": "Amazon EC2 region name",
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
 S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
 bucket, Elastic Load Balancing load balancer, or another resource record set in
 this hosted zone",
          "EvaluateTargetHealth": true|false
```

```
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

失效转移语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Failover": "PRIMARY" | "SECONDARY",
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ],
        "HealthCheckId": "ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

失效转移别名语法：

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
```

```
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Failover": "PRIMARY" | "SECONDARY",
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
 S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
 bucket, Elastic Load Balancing load balancer, or another resource record set in
 this hosted zone",
          "EvaluateTargetHealth": true|false
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangeResourceRecordSets。

change-tags-for-resource

以下代码示例演示了如何使用 change-tags-for-resource。

Amazon CLI

以下命令将名为 owner 的标签添加到由 ID 指定的运行状况检查资源：

```
aws route53 change-tags-for-resource --resource-type healthcheck --resource-
id 6233434j-18c1-34433-ba8e-3443434 --add-tags Key=owner,Value=myboss
```

以下命令将从由 ID 指定的托管区资源中移除名为 owner 的标签：

```
aws route53 change-tags-for-resource --resource-type hostedzone --resource-
id Z1523434445 --remove-tag-keys owner
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangeTagsForResource。

create-health-check

以下代码示例演示了如何使用 create-health-check。

Amazon CLI

创建运行状况检查

以下 create-health-check 命令使用调用方参考 2014-04-01-18:47 和文件 C:\awscli\route53\create-health-check.json 中的 JSON 格式配置创建运行状况检查：

```
aws route53 create-health-check --caller-reference 2014-04-01-18:47 --health-check-config file://C:\awscli\route53\create-health-check.json
```

JSON 语法：

```
{
  "IPAddress": "IP address of the endpoint to check",
  "Port": port on the endpoint to check--required when Type is "TCP",
  "Type": "HTTP"|"HTTPS"|"HTTP_STR_MATCH"|"HTTPS_STR_MATCH"|"TCP",
  "ResourcePath": "path of the file that you want Amazon Route 53 to request--all
 Types except TCP",
  "FullyQualifiedDomainName": "domain name of the endpoint to check--all Types
 except TCP",
  "SearchString": "if Type is HTTP_STR_MATCH or HTTPS_STR_MATCH, the string to
 search for in the response body from the specified resource",
  "RequestInterval": 10 | 30,
  "FailureThreshold": integer between 1 and 10
}
```

要将运行状况检查添加到 Route 53 资源记录集，请使用 change-resource-record-sets 命令。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的“Amazon Route 53 运行状况检查和 DNS 故障转移”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHealthCheck。

create-hosted-zone

以下代码示例演示了如何使用 create-hosted-zone。

Amazon CLI

创建托管区

以下 create-hosted-zone 命令使用调用方参考 2014-04-01-18:47 添加名为 example.com 的托管区。可选注释中包含一个空格，因此必须用引号括起来：

```
aws route53 create-hosted-zone --name example.com --caller-
reference 2014-04-01-18:47 --hosted-zone-config Comment="command-line version"
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的“使用托管区”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateHostedZone。

delete-health-check

以下代码示例演示了如何使用 delete-health-check。

Amazon CLI

删除运行状况检查

以下 delete-health-check 命令将删除 health-check-id 为 e75b48d9-547a-4c3d-88a5-ae4002397608 的运行状况检查：

```
aws route53 delete-health-check --health-check-id e75b48d9-547a-4c3d-88a5-
ae4002397608
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHealthCheck。

delete-hosted-zone

以下代码示例演示了如何使用 delete-hosted-zone。

Amazon CLI

删除托管区

以下 delete-hosted-zone 命令将删除 id 为 Z36KTIQEXAMPLE 的托管区：

```
aws route53 delete-hosted-zone --id Z36KTIQEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteHostedZone。

get-change

以下代码示例演示了如何使用 get-change。

Amazon CLI

获取资源记录集更改的状态

以下 get-change 命令获取 Id 为 /change/CWPIK4URU2I5S 的 change-resource-record-sets 请求的状态和其他信息：

```
aws route53 get-change --id /change/CWPIK4URU2I5S
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetChange。

get-health-check

以下代码示例演示了如何使用 get-health-check。

Amazon CLI

获取有关运行状况检查的信息

以下 get-health-check 命令获取有关 health-check-id 为 02ec8401-9879-4259-91fa-04e66d094674 的运行状况检查的信息：

```
aws route53 get-health-check --health-check-id 02ec8401-9879-4259-91fa-04e66d094674
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetHealthCheck。

get-hosted-zone

以下代码示例演示了如何使用 get-hosted-zone。

Amazon CLI

获取有关托管区的信息

以下 get-hosted-zone 命令获取 id 为 Z1R8UBAEXAMPLE 的托管区的相关信息：

```
aws route53 get-hosted-zone --id Z1R8UBAEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetHostedZone。

list-health-checks

以下代码示例演示了如何使用 list-health-checks。

Amazon CLI

列出与当前 Amazon 账户关联的运行状况检查

以下 list-health-checks 命令列出与当前 Amazon 账户关联的前 100 个运行状况检查的详细信息：

```
aws route53 list-health-checks
```

如果您有超过 100 个运行状况检查，或者想要将它们按小于 100 的组列出，请包含 --maxitems 参数。例如，要一次列出一个运行状况检查，请使用以下命令：

```
aws route53 list-health-checks --max-items 1
```

要查看下一个运行状况检查，请从上一个命令的响应中获取 NextToken 的值，并将其包含在 --starting-token 参数中，例如：

```
aws route53 list-health-checks --max-items 1 --starting-token Z3M3LMPEXAMPLE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHealthChecks。

list-hosted-zones-by-name

以下代码示例演示了如何使用 list-hosted-zones-by-name。

Amazon CLI

以下命令最多列出 100 个按域名排序的托管区：

```
aws route53 list-hosted-zones-by-name
```

输出：

```
{
  "HostedZones": [
      {
          "ResourceRecordSetCount": 2,
          "CallerReference": "test20150527-2",
          "Config": {
              "Comment": "test2",
              "PrivateZone": false
          },
          "Id": "/hostedzone/Z119WBBTVP5WFX",
          "Name": "2.example.com."
      },
      {
          "ResourceRecordSetCount": 2,
          "CallerReference": "test20150527-1",
          "Config": {
              "Comment": "test",
              "PrivateZone": false
          },
          "Id": "/hostedzone/Z3P5QSUBK4POTI",
          "Name": "www.example.com."
      }
  ],
  "IsTruncated": false,
  "MaxItems": "100"
}
```

以下命令列出按名称排序的托管区，开头为 www.example.com：

```
aws route53 list-hosted-zones-by-name --dns-name www.example.com
```

输出：

```
{
  "HostedZones": [
      {
          "ResourceRecordSetCount": 2,
          "CallerReference": "mwunderl20150527-1",
          "Config": {
              "Comment": "test",
```

```
                "PrivateZone": false
            },
            "Id": "/hostedzone/Z3P5QSUBK4POTI",
            "Name": "www.example.com."
        }
    ],
    "DNSName": "www.example.com",
    "IsTruncated": false,
    "MaxItems": "100"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListHostedZonesByName。

list-hosted-zones

以下代码示例演示了如何使用 list-hosted-zones。

Amazon CLI

列出与当前 Amazon 账户关联的托管区

如下 list-hosted-zones 命令列出与当前 Amazon 账户关联的前 100 个托管区的摘要信息：

```
aws route53 list-hosted-zones
```

如果您有超过 100 个托管区，或者想要将它们按小于 100 的数量分组列出，请包含 --max-items 参数。例如，要一次列出一个托管区，请使用以下命令：

```
aws route53 list-hosted-zones --max-items 1
```

要查看有关下一个托管区的信息，请从上一个命令的响应中获取 NextToken 的值，并将其包含在 --starting-token 参数中，例如：

```
aws route53 list-hosted-zones --max-items 1 --starting-token Z3M3LMPEXAMPLE
```

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考中的 ListHostedZones。

list-query-logging-configs

以下代码示例演示了如何使用 list-query-logging-configs。

Amazon CLI

列出查询日志记录配置

以下 `list-query-logging-configs` 示例针对托管区 `Z1OX3WQEXAMPLE`，列出了 Amazon 账户中前 100 个查询日志配置的相关信息。

```
aws route53 list-query-logging-configs \
    --hosted-zone-id Z1OX3WQEXAMPLE
```

输出：

```
{
    "QueryLoggingConfigs": [
        {
            "Id": "964ff34e-ae03-4f06-80a2-9683cexample",
            "HostedZoneId": "Z1OX3WQEXAMPLE",
            "CloudWatchLogsLogGroupArn": "arn:aws:logs:us-east-1:111122223333:log-
group:/aws/route53/example.com:*"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的记录 DNS 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueryLoggingConfigs。

list-resource-record-sets

以下代码示例演示了如何使用 `list-resource-record-sets`。

Amazon CLI

列出托管区中的资源记录集

以下 `list-resource-record-sets` 命令列出了有关指定托管区中前 100 个资源记录集的摘要信息。

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE
```

如果托管区中包含的资源记录集超过 100 个，或者想要将它们按小于 100 的数量分组列出，请包含 `--maxitems` 参数。例如，要一次列出一个资源记录集，请使用以下命令：

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --max-items 1
```

要查看有关托管区中下一个资源记录的信息，请从上一个命令的响应中获取 NextToken 的值，并将其包含在 --starting-token 参数中，例如：

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --max-items 1
 --starting-token Z3M3LMPEXAMPLE
```

要查看具有特定名称的所有资源记录集，请使用 --query 参数将其过滤掉。例如：

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --
query "ResourceRecordSets[?Name == 'example.domain.']"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceRecordSets。

使用Amazon CLI的 Route 53 域注册示例

以下代码示例演示如何通过将Amazon Command Line Interface与 Route 53 域注册结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`check-domain-availability`

以下代码示例演示了如何使用 check-domain-availability。

Amazon CLI

确定是否可以在 Route 53 上注册域名

以下 check-domain-availability 命令返回有关域名 example.com 是否可以使用 Route 53 进行注册的信息。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains check-domain-availability \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "Availability": "UNAVAILABLE"
}
```

Route 53 支持大量顶级域（TLD），例如 .com 和 .jp，但我们并不支持所有可用的 TLD。如果您查看某个域是否可用，而 Route 53 不支持该顶级域，check-domain-availability 会返回以下消息。

```
An error occurred (UnsupportedTLD) when calling the CheckDomainAvailability
 operation: <top-level domain> tld is not supported.
```

有关可在将域注册到 Route 53 时使用的 TLD 的列表，请参阅《Amazon Route 53 开发人员指南》中的可向 Amazon Route 53 注册的域。有关使用 Amazon Route 53 注册域的更多信息，请参阅《Amazon Route 53 开发人员指南》中的注册新域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckDomainAvailability。

check-domain-transferability

以下代码示例演示了如何使用 check-domain-transferability。

Amazon CLI

确定域是否可以传输到 Route 53

以下 check-domain-transferability 命令返回有关是否可以将域名 example.com 传输到 Route 53 的信息。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains check-domain-transferability \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "Transferability": {
        "Transferable": "UNTRANSFERABLE"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的将域注册转移到 Amazon Route 53。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckDomainTransferability。

delete-tags-for-domain

以下代码示例演示了如何使用 delete-tags-for-domain。

Amazon CLI

删除域的标签

以下 delete-tags-for-domain 命令从指定域中删除三个标签。请注意，您只能指定标签键，不能指定标签值。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains delete-tags-for-domain \
    --region us-east-1 \
    --domain-name example.com \
    --tags-to-delete accounting-key hr-key engineering-key
```

此命令不生成任何输出。

要确认标签已删除，则可以运行 list-tags-for-domain。有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的为 Amazon Route 53 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTagsForDomain。

disable-domain-auto-renew

以下代码示例演示了如何使用 disable-domain-auto-renew。

Amazon CLI

禁用域的自动续订

以下 disable-domain-auto-renew 命令将 Route 53 配置为在域名注册到期之前不自动续订域 example.com。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains disable-domain-auto-renew \
    --region us-east-1 \
    --domain-name example.com
```

此命令不生成任何输出。

要确认设置已更改，您可以运行 get-domain-detail。如果禁用了自动续订，则 AutoRenew 的值为 False。有关自动续订的更多信息，请参阅《Amazon Route 53 开发者指南》中的“续订域名注册”<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableDomainAutoRenew。

disable-domain-transfer-lock

以下代码示例演示了如何使用 disable-domain-transfer-lock。

Amazon CLI

禁用域名的转移锁定

以下 disable-domain-transfer-lock 命令可移除域 example.com 的转移锁定，以便可以将域转移到其他注册商。此命令将更改 clientTransferProhibited 状态。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains disable-domain-transfer-lock \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "OperationId": "3f28e0ac-126a-4113-9048-cc930example"
}
```

要确认转移锁定已更改，您可以运行 get-domain-detail。禁用转移锁定时，StatusList 的值不包括 clientTransferProhibited。

有关转移过程的更多信息，请参阅《Amazon Route 53 开发人员指南》中的将域从 Amazon Route 53 转移到另一注册商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableDomainTransferLock。

enable-domain-auto-renew

以下代码示例演示了如何使用 enable-domain-auto-renew。

Amazon CLI

启用域的自动续订

以下 enable-domain-auto-renew 命令将 Route 53 配置为在域注册到期之前自动续订域 example.com。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains enable-domain-auto-renew \
    --region us-east-1 \
    --domain-name example.com
```

此命令不生成任何输出。要确认设置已更改，您可以运行 get-domain-detail。如果启用了自动续订，AutoRenew 的值为 True。

有关自动续订的更多信息，请参阅《Amazon Route 53 开发者指南》中的“续订域名注册”<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableDomainAutoRenew。

enable-domain-transfer-lock

以下代码示例演示了如何使用 enable-domain-transfer-lock。

Amazon CLI

在域名上启用转移锁定

以下 enable-domain-transfer-lock 命令可锁定指定域，以便可以转移到其他注册商。此命令将更改 clientTransferProhibited 状态。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains enable-domain-transfer-lock \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "OperationId": "3f28e0ac-126a-4113-9048-cc930example"
}
```

要确认转移锁定已更改，您可以运行 get-domain-detail。启用转移锁定后，StatusList 的值包括 clientTransferProhibited。

有关转移过程的更多信息，请参阅《Amazon Route 53 开发人员指南》中的将域从 Amazon Route 53 转移到另一注册商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableDomainTransferLock。

get-contact-reachability-status

以下代码示例演示了如何使用 get-contact-reachability-status。

Amazon CLI

确定注册人联系人是否已回复确认电子邮件

以下 get-contact-reachability-status 命令返回有关指定域的注册人联系人是否已回复确认电子邮件的信息。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains get-contact-reachability-status \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "domainName": "example.com",
    "status": "DONE"
}
```

有关更多信息，请参阅《Amazon Route 53 开发者指南》中的重新发送授权和确认电子邮件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetContactReachabilityStatus。

get-domain-detail

以下代码示例演示了如何使用 get-domain-detail。

Amazon CLI

获取有关指定域的详细信息

以下 get-domain-detail 命令显示有关指定域的详细信息。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains get-domain-detail \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "DomainName": "example.com",
    "Nameservers": [
        {
            "Name": "ns-2048.awsdns-64.com",
            "GlueIps": []
        },
        {
            "Name": "ns-2049.awsdns-65.net",
            "GlueIps": []
        },
        {
            "Name": "ns-2050.awsdns-66.org",
            "GlueIps": []
        },
        {
            "Name": "ns-2051.awsdns-67.co.uk",
            "GlueIps": []
        }
    ],
    "AutoRenew": true,
    "AdminContact": {
        "FirstName": "Saanvi",
        "LastName": "Sarkar",
        "ContactType": "COMPANY",
        "OrganizationName": "Example",
        "AddressLine1": "123 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "ssarkar@example.com",
        "ExtraParams": []
    },
    "RegistrantContact": {
        "FirstName": "Alejandro",
        "LastName": "Rosalez",
        "ContactType": "COMPANY",
        "OrganizationName": "Example",
        "AddressLine1": "123 Main Street",
        "City": "Anytown",
```

```
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "arosalez@example.com",
        "ExtraParams": []
    },
    "TechContact": {
        "FirstName": "Wang",
        "LastName": "Xiulan",
        "ContactType": "COMPANY",
        "OrganizationName": "Example",
        "AddressLine1": "123 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "wxiulan@example.com",
        "ExtraParams": []
    },
    "AdminPrivacy": true,
    "RegistrantPrivacy": true,
    "TechPrivacy": true,
    "RegistrarName": "Amazon Registrar, Inc.",
    "WhoIsServer": "whois.registrar.amazon",
    "RegistrarUrl": "http://registrar.amazon.com",
    "AbuseContactEmail": "abuse@registrar.amazon.com",
    "AbuseContactPhone": "+1.2062661000",
    "CreationDate": 1444934889.601,
    "ExpirationDate": 1602787689.0,
    "StatusList": [
        "clientTransferProhibited"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainDetail。

get-domain-suggestions

以下代码示例演示了如何使用 get-domain-suggestions。

Amazon CLI

获取建议的域名列表

以下 get-domain-suggestions 命令根据域名 example.com 显示建议域名列表。响应仅包含可用域名。此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains get-domain-suggestions \
    --region us-east-1 \
    --domain-name example.com \
    --suggestion-count 10 \
    --only-available
```

输出：

```
{
    "SuggestionsList": [
        {
            "DomainName": "egzaampal.com",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "examplelaw.com",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "examplehouse.net",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "homeexample.net",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "examplelist.com",
            "Availability": "AVAILABLE"
       },
        {
            "DomainName": "examplenews.net",
            "Availability": "AVAILABLE"
        },
        {
```

```
            "DomainName": "officeexample.com",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "exampleworld.com",
            "Availability": "AVAILABLE"
        },
        {
            "DomainName": "exampleart.com",
            "Availability": "AVAILABLE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDomainSuggestions。

get-operation-detail

以下代码示例演示了如何使用 get-operation-detail。

Amazon CLI

获取操作的当前状态

某些域注册操作是异步运行的，并在完成之前返回响应。这些操作会返回一个操作 ID，您可以使用它来获取当前状态。以下 get-operation-detail 命令返回指定操作的状态。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains get-operation-detail \
    --region us-east-1 \
    --operation-id edbd8d63-7fe7-4343-9bc5-54033example
```

输出：

```
{
    "OperationId": "edbd8d63-7fe7-4343-9bc5-54033example",
    "Status": "SUCCESSFUL",
    "DomainName": "example.com",
    "Type": "DOMAIN_LOCK",
    "SubmittedDate": 1573749367.864
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOperationDetail。

`list-domains`

以下代码示例演示了如何使用 `list-domains`。

Amazon CLI

列出使用当前 Amazon 账户注册的域

以下 `list-domains` 命令列出了与使用当前 Amazon 账户注册的域相关的摘要信息。

此命令仅在 `us-east-1` 区域中运行。如果您的默认区域设置为 `us-east-1`，则可以省略 `region` 参数。

```
aws route53domains list-domains
    --region us-east-1
```

输出：

```
{
    "Domains": [
        {
            "DomainName": "example.com",
            "AutoRenew": true,
            "TransferLock": true,
            "Expiry": 1602712345.0
        },
        {
            "DomainName": "example.net",
            "AutoRenew": true,
            "TransferLock": true,
            "Expiry": 1602723456.0
        },
        {
            "DomainName": "example.org",
            "AutoRenew": true,
            "TransferLock": true,
            "Expiry": 1602734567.0
        }
    ]
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDomains。

list-operations

以下代码示例演示了如何使用 list-operations。

Amazon CLI

列出返回操作 ID 的操作的状态

某些域注册操作是异步运行的，并在完成之前返回响应。这些操作会返回一个操作 ID，您可以使用它来获取当前状态。以下 list-operations 命令列出了有关当前域注册操作的状态等摘要信息。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains list-operations
    --region us-east-1
```

输出：

```
{
    "Operations": [
        {
            "OperationId": "aab9822f-1da0-4bf3-8a15-fd4e0example",
            "Status": "SUCCESSFUL",
            "Type": "DOMAIN_LOCK",
            "SubmittedDate": 1455321739.986
        },
        {
            "OperationId": "c24379ed-76be-42f8-bdad-9379bexample",
            "Status": "SUCCESSFUL",
            "Type": "UPDATE_NAMESERVER",
            "SubmittedDate": 1468960475.109
        },
        {
            "OperationId": "f47e1297-ef9e-4c2b-ae1e-a5fcbexample",
            "Status": "SUCCESSFUL",
            "Type": "RENEW_DOMAIN",
```

```
            "SubmittedDate": 1473561835.943
        },
        {
            "OperationId": "75584f23-b15f-459e-aed7-dc6f5example",
            "Status": "SUCCESSFUL",
            "Type": "UPDATE_DOMAIN_CONTACT",
            "SubmittedDate": 1547501003.41
        }
    ]
}
```

输出包括返回操作 ID 的所有操作，以及您在使用当前 Amazon 账户注册的所有域中执行的操作。如果只想获取在指定日期之后提交的操作，可以包含 `submitted-since` 参数并以 Unix 格式和协调世界时（UTC）指定日期。以下命令可获取在协调世界时间 2020 年 1 月 1 日凌晨 12:00 之后提交的所有操作的状态。

```
aws route53domains list-operations \
    --submitted-since 1577836800
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOperations。

`list-tags-for-domain`

以下代码示例演示了如何使用 `list-tags-for-domain`。

Amazon CLI

列出域的标签

以下 `list-tags-for-domain` 命令列出当前与指定域关联的标签。

此命令仅在 `us-east-1` 区域中运行。如果您的默认区域设置为 `us-east-1`，则可以省略 `region` 参数。

```
aws route53domains list-tags-for-domain \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "TagList": [
```

```
        {
            "Key": "key1",
            "Value": "value1"
        },
        {
            "Key": "key2",
            "Value": "value2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的为 Amazon Route 53 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForDomain。

register-domain

以下代码示例演示了如何使用 register-domain。

Amazon CLI

注册域

以下 register-domain 命令注册一个域，并从 JSON 格式的文件中检索所有参数值。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains register-domain \
    --region us-east-1 \
    --cli-input-json file://register-domain.json
```

的内容register-domain.json：

```
{
    "DomainName": "example.com",
    "DurationInYears": 1,
    "AutoRenew": true,
    "AdminContact": {
        "FirstName": "Martha",
        "LastName": "Rivera",
        "ContactType": "PERSON",
```

```
        "OrganizationName": "Example",
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "mrivera@example.com"
    },
    "RegistrantContact": {
        "FirstName": "Li",
        "LastName": "Juan",
        "ContactType": "PERSON",
        "OrganizationName": "Example",
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "ljuan@example.com"
    },
    "TechContact": {
        "FirstName": "Mateo",
        "LastName": "Jackson",
        "ContactType": "PERSON",
        "OrganizationName": "Example",
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "mjackson@example.com"
    },
    "PrivacyProtectAdminContact": true,
    "PrivacyProtectRegistrantContact": true,
    "PrivacyProtectTechContact": true
}
```

输出：

```
{
```

```
    "OperationId": "b114c44a-9330-47d1-a6e8-a0b11example"
}
```

要确认操作成功，可以运行 get-operation-detail。有关更多信息，请参阅 get-operation-detail。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的注册新域。

有关哪些顶级域（TLD）需要 ExtraParams 值以及何为有效值的信息，请参阅《Amazon Route 53 API Reference》中的 ExtraParam。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterDomain。

renew-domain

以下代码示例演示了如何使用 renew-domain。

Amazon CLI

续订域

以下 renew-domain 命令将续订指定域五年。要获取 current-expiry-year 值，请使用 get-domain-detail 命令，然后转换 ExpirationDate 值的 Unix 格式。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains renew-domain \
    --region us-east-1 \
    --domain-name example.com \
    --duration-in-years 5 \
    --current-expiry-year 2020
```

输出：

```
{
    "OperationId": "3f28e0ac-126a-4113-9048-cc930example"
}
```

要确认操作成功，可以运行 get-operation-detail。有关更多信息，请参阅 get-operation-detail。

每个顶级域（TLD）（例如.com 或.org）的注册管理机构控制您可以续订域的最大年数。要获取域的最长续订时间，请参阅《Amazon Route 53 开发人员指南》中可向 Amazon Route 53 注册的域中“注册和续订期”部分。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的续订域注册。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RenewDomain。

`resend-contact-reachability-email`

以下代码示例演示了如何使用 `resend-contact-reachability-email`。

Amazon CLI

向注册人联系人的电子邮件地址发送确认电子邮件

以下 `resend-contact-reachability-email` 命令将确认电子邮件重新发送到 example.com 域的注册商联系人的当前电子邮件地址。

此命令仅在 `us-east-1` 区域中运行。如果您的默认区域设置为 `us-east-1`，则可以省略 `region` 参数。

```
aws route53domains resend-contact-reachability-email \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "domainName": "example.com",
    "emailAddress": "moliveira@example.com",
    "isAlreadyVerified": true
}
```

如果 `isAlreadyVerified` 值为 `true`，如本例所示，则注册人联系人已经确认可以访问指定的电子邮件地址。

有关更多信息，请参阅《Amazon Route 53 开发者指南》中的重新发送授权和确认电子邮件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResendContactReachabilityEmail。

retrieve-domain-auth-code

以下代码示例演示了如何使用 retrieve-domain-auth-code。

Amazon CLI

获取域的授权码，以将域转移到其他注册商

以下 retrieve-domain-auth-code 命令将获取 example.com 域的当前授权码。当您想将域转移给其他域注册商时，可以将此值提供给该注册商。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains retrieve-domain-auth-code \
    --region us-east-1 \
    --domain-name example.com
```

输出：

```
{
    "AuthCode": ")o!v3dJeXampLe"
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的将域从 Amazon Route 53 转移到另一注册商。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RetrieveDomainAuthCode。

transfer-domain

以下代码示例演示了如何使用 transfer-domain。

Amazon CLI

将域转移到 Amazon Route 53

以下 transfer-domain 命令使用由 JSON 格式文件 C:\temp\transfer-domain.json 提供的参数将域转移到 Route 53。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains transfer-domain \
    --region us-east-1 \
    --cli-input-json file://C:\temp\transfer-domain.json
```

的内容transfer-domain.json：

```
{
    "DomainName": "example.com",
    "DurationInYears": 1,
    "Nameservers": [
        {
            "Name": "ns-2048.awsdns-64.com"
        },
        {
            "Name": "ns-2049.awsdns-65.net"
        },
        {
            "Name": "ns-2050.awsdns-66.org"
        },
        {
            "Name": "ns-2051.awsdns-67.co.uk"
        }
    ],
    "AuthCode": ")o!v3dJeXampLe",
    "AutoRenew": true,
    "AdminContact": {
        "FirstName": "Martha",
        "LastName": "Rivera",
        "ContactType": "PERSON",
        "OrganizationName": "Example",
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "mrivera@example.com"
    },
    "RegistrantContact": {
        "FirstName": "Li",
        "LastName": "Juan",
        "ContactType": "PERSON",
        "OrganizationName": "Example",
```

```
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "ljuan@example.com"
    },
    "TechContact": {
        "FirstName": "Mateo",
        "LastName": "Jackson",
        "ContactType": "PERSON",
        "OrganizationName": "Example",
        "AddressLine1": "1 Main Street",
        "City": "Anytown",
        "State": "WA",
        "CountryCode": "US",
        "ZipCode": "98101",
        "PhoneNumber": "+1.8005551212",
        "Email": "mjackson@example.com"
    },
    "PrivacyProtectAdminContact": true,
    "PrivacyProtectRegistrantContact": true,
    "PrivacyProtectTechContact": true
}
```

输出：

```
{
    "OperationId": "b114c44a-9330-47d1-a6e8-a0b11example"
}
```

要确认操作成功，可以运行 `get-operation-detail`。有关更多信息，请参阅 get-operation-detail。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的将域注册转移到 Amazon Route 53。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TransferDomain。

update-domain-contact-privacy

以下代码示例演示了如何使用 `update-domain-contact-privacy`。

Amazon CLI

更新域联系人的隐私设置

以下 update-domain-contact-privacy 命令关闭了 example.com 域的管理员联系人的隐私保护。此命令仅在 us-east-1 区域中运行。

如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains update-domain-contact-privacy \
    --region us-east-1 \
    --domain-name example.com \
    --no-admin-privacy
```

输出：

```
{
    "OperationId": "b3a219e9-d801-4244-b533-b7256example"
}
```

要确认操作成功，可以运行 get-operation-detail。有关更多信息，请参阅 get-operation-detail。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的为域的联系信息启用或禁用隐私保护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainContactPrivacy。

update-domain-contact

以下代码示例演示了如何使用 update-domain-contact。

Amazon CLI

更新域的联系信息

以下 update-domain-contact 命令更新域的联系人信息，从 JSON 格式的文件 C:\temp\update-domain-contact.json 中获取参数。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains update-domain-contact \
    --region us-east-1 \
    --cli-input-json file://C:\temp\update-domain-contact.json
```

的内容update-domain-contact.json：

```
{
    "AdminContact": {
        "AddressLine1": "101 Main Street",
        "AddressLine2": "Suite 1a",
        "City": "Seattle",
        "ContactType": "COMPANY",
        "CountryCode": "US",
        "Email": "w.xiulan@example.com",
        "FirstName": "Wang",
        "LastName": "Xiulan",
        "OrganizationName": "Example",
        "PhoneNumber": "+1.8005551212",
        "State": "WA",
        "ZipCode": "98101"
    },
    "DomainName": "example.com",
    "RegistrantContact": {
        "AddressLine1": "101 Main Street",
        "AddressLine2": "Suite 1a",
        "City": "Seattle",
        "ContactType": "COMPANY",
        "CountryCode": "US",
        "Email": "w.xiulan@example.com",
        "FirstName": "Wang",
        "LastName": "Xiulan",
        "OrganizationName": "Example",
        "PhoneNumber": "+1.8005551212",
        "State": "WA",
        "ZipCode": "98101"
    },
    "TechContact": {
        "AddressLine1": "101 Main Street",
        "AddressLine2": "Suite 1a",
        "City": "Seattle",
        "ContactType": "COMPANY",
        "CountryCode": "US",
        "Email": "w.xiulan@example.com",
```

```
        "FirstName": "Wang",
        "LastName": "Xiulan",
        "OrganizationName": "Example",
        "PhoneNumber": "+1.8005551212",
        "State": "WA",
        "ZipCode": "98101"
    }
}
```

输出：

```
{
    "OperationId": "b3a219e9-d801-4244-b533-b7256example"
}
```

要确认操作是否成功，您可以运行 get-domain-detail。有关更多信息，请参阅《Amazon Route 53 开发者指南》中的更新域的联系人信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainContact。

update-domain-nameservers

以下代码示例演示了如何使用 update-domain-nameservers。

Amazon CLI

更新域的名称服务器

以下 update-domain-nameservers 命令更新了域的名称服务器。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains update-domain-nameservers \
    --region us-east-1 \
    --domain-name example.com \
    --
nameservers Name=ns-1.awsdns-01.org Name=ns-2.awsdns-02.co.uk Name=ns-3.awsdns-03.net Name=n
```

输出：

```
{
```

```
    "OperationId": "f1691ec4-0e7a-489e-82e0-b19d3example"
}
```

要确认操作是否成功，您可以运行 get-domain-detail。

有关更多信息，请参阅《Amazon Route 53 开发人员》中的为域添加或更改名称服务器和粘附记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDomainNameservers。

update-tags-for-domain

以下代码示例演示了如何使用 update-tags-for-domain。

Amazon CLI

为域名添加或更新标签

以下 update-tags-for-domain 命令添加或更新 example.com 域的两个键和相应值。要更新键值，只需包含键和新值。一次只能在一个域中添加或更新标签。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains update-tags-for-domain \
    --region us-east-1 \
    --domain-name example.com \
    --tags-to-update "Key=key1,Value=value1" "Key=key2,Value=value2"
```

此命令不生成任何输出。要确认标签是否已添加或更新，您可以运行 list-tags-for-domain。

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的为 Amazon Route 53 资源添加标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTagsForDomain。

view-billing

以下代码示例演示了如何使用 view-billing。

Amazon CLI

获取当前 Amazon 账户的域注册费用的账单信息

以下 view-billing 命令返回当前账户自 2018 年 1 月 1 日（Unix 时间为 1514764800）至 2019 年 12 月 31 日午夜（Unix 时间为 1577836800）期间的所有与域相关的账单记录。

此命令仅在 us-east-1 区域中运行。如果您的默认区域设置为 us-east-1，则可以省略 region 参数。

```
aws route53domains view-billing \
    --region us-east-1 \
    --start-time 1514764800 \
    --end-time 1577836800
```

输出：

```
{
    "BillingRecords": [
        {
            "DomainName": "example.com",
            "Operation": "RENEW_DOMAIN",
            "InvoiceId": "149962827",
            "BillDate": 1536618063.181,
            "Price": 12.0
        },
        {
            "DomainName": "example.com",
            "Operation": "RENEW_DOMAIN",
            "InvoiceId": "290913289",
            "BillDate": 1568162630.884,
            "Price": 12.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 API Reference》中的 ViewBilling。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ViewBilling。

使用 Amazon CLI 的 Route 53 配置文件示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Route 53 配置文件结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-profile`

以下代码示例演示了如何使用 `associate-profile`。

Amazon CLI

关联配置文件

以下 `associate-profile` 示例将配置文件关联到 VPC。

```
aws route53profiles associate-profile \
    --name test-association \
    --profile-id rp-4987774726example \
    --resource-id vpc-0af3b96b3example
```

输出：

```
{
    "ProfileAssociation": {
        "CreationTime": 1710851336.527,
        "Id": "rpassoc-489ce212fexample",
        "ModificationTime": 1710851336.527,
        "Name": "test-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceId": "vpc-0af3b96b3example",
        "Status": "CREATING",
        "StatusMessage": "Creating Profile Association"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的使用配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateProfile。

associate-resource-to-profile

以下代码示例演示了如何使用 associate-resource-to-profile。

Amazon CLI

将资源与配置文件关联

以下 associate-resource-to-profile 示例将优先级为 102 的 DNS 防火墙规则组与配置文件相关联。

```
aws route53profiles associate-resource-to-profile \
    --name test-resource-association \
    --profile-id rp-4987774726example \
    --resource-arn arn:aws:route53resolver:us-east-1:123456789012:firewall-rule-
group/rslvr-frg-cfe7f72example \
    --resource-properties "{\"priority\": 102}"
```

输出：

```
{
    "ProfileResourceAssociation": {
        "CreationTime": 1710851216.613,
        "Id": "rpr-001913120a7example",
        "ModificationTime": 1710851216.613,
        "Name": "test-resource-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceArn": "arn:aws:route53resolver:us-east-1:123456789012:firewall-
rule-group/rslvr-frg-cfe7f72example",
        "ResourceProperties": "{\"priority\":102}",
        "ResourceType": "FIREWALL_RULE_GROUP",
        "Status": "UPDATING",
        "StatusMessage": "Updating the Profile to DNS Firewall rule group
 association"
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateResourceToProfile。

create-profile

以下代码示例演示了如何使用 create-profile。

Amazon CLI

创建配置文件

以下 create-profile 示例创建了一个配置文件。

```
aws route53profiles create-profile \
    --name test
```

输出：

```
{
    "Profile": {
        "Arn": "arn:aws:route53profiles:us-east-1:123456789012:profile/
rp-6ffe47d5example",
        "ClientToken": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
        "CreationTime": 1710850903.578,
        "Id": "rp-6ffe47d5example",
        "ModificationTime": 1710850903.578,
        "Name": "test",
        "OwnerId": "123456789012",
        "ShareStatus": "NOT_SHARED",
        "Status": "COMPLETE",
        "StatusMessage": "Created Profile"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProfile。

delete-profile

以下代码示例演示了如何使用 delete-profile。

Amazon CLI

删除配置文件

以下 delete-profile 示例删除了配置文件。

```
aws route53profiles delete-profile \
    --profile-id rp-6ffe47d5example
```

输出：

```
{
    "Profile": {
        "Arn": "arn:aws:route53profiles:us-east-1:123456789012:profile/
rp-6ffe47d5example",
        "ClientToken": "0a15fec0-05d9-4f78-bec0-EXAMPLE11111",
        "CreationTime": 1710850903.578,
        "Id": "rp-6ffe47d5example",
        "ModificationTime": 1710850903.578,
        "Name": "test",
        "OwnerId": "123456789012",
        "ShareStatus": "NOT_SHARED",
        "Status": "DELETED",
        "StatusMessage": "Deleted Profile"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProfile。

disassociate-profile

以下代码示例演示了如何使用 disassociate-profile。

Amazon CLI

取消与配置文件的关联

以下 disassociate-profile 示例取消了配置文件与 VPC 的关联。

```
aws route53profiles disassociate-profile \
    --profile-id rp-4987774726example \
```

```
    --resource-id vpc-0af3b96b3example
```

输出：

```
{
    "ProfileAssociation": {
        "CreationTime": 1710851336.527,
        "Id": "rpassoc-489ce212fexample",
        "ModificationTime": 1710851401.362,
        "Name": "test-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceId": "vpc-0af3b96b3example",
        "Status": "DELETING",
        "StatusMessage": "Deleting Profile Association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateProfile。

disassociate-resource-from-profile

以下代码示例演示了如何使用 disassociate-resource-from-profile。

Amazon CLI

取消资源与配置文件的关联

以下 disassociate-resource-from-profile 示例取消了 DNS 防火墙规则组与配置文件的关联。

```
aws route53profiles disassociate-resource-from-profile \
    --profile-id rp-4987774726example \
    --resource-arn arn:aws:route53resolver:us-east-1:123456789012:firewall-rule-group/rslvr-frg-cfe7f72example
```

输出：

```
{
    "ProfileResourceAssociation": {
```

```
        "CreationTime": 1710851216.613,
        "Id": "rpr-001913120a7example",
        "ModificationTime": 1710852624.36,
        "Name": "test-resource-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceArn": "arn:aws:route53resolver:us-east-1:123456789012:firewall-
rule-group/rslvr-frg-cfe7f72example",
        "ResourceProperties": "{\"priority\":105}",
        "ResourceType": "FIREWALL_RULE_GROUP",
        "Status": "DELETING",
        "StatusMessage": "Deleting the Profile to DNS Firewall rule group
 association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateResourceFromProfile。

get-profile-association

以下代码示例演示了如何使用 get-profile-association。

Amazon CLI

获取有关配置文件关联的信息

以下 get-profile-association 返回有关指定配置文件关联的信息。

```
aws route53profiles get-profile-association \
    --profile-association-id rpassoc-489ce212fexample
```

输出：

```
{
    "ProfileAssociation": {
        "CreationTime": 1709338817.148,
        "Id": "rrpassoc-489ce212fexample",
        "ModificationTime": 1709338974.772,
        "Name": "test-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceId": "vpc-0af3b96b3example",
```

```
        "Status": "COMPLETE",
        "StatusMessage": "Created Profile Association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProfileAssociation。

get-profile-resource-association

以下代码示例演示了如何使用 get-profile-resource-association。

Amazon CLI

获取与配置文件关联的资源的相关信息

以下 get-profile-resource-association 返回有关指定资源与配置文件的关联的信息。

```
aws route53profiles get-profile-resource-association \
    --profile-resource-association-id rpr-001913120a7example
```

输出：

```
{
    "ProfileResourceAssociation": {
        "CreationTime": 1710851216.613,
        "Id": "rpr-001913120a7example",
        "ModificationTime": 1710852303.798,
        "Name": "test-resource-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceArn": "arn:aws:route53resolver:us-east-1:123456789012:firewall-
rule-group/rslvr-frg-cfe7f72example",
        "ResourceProperties": "{\"priority\":105}",
        "ResourceType": "FIREWALL_RULE_GROUP",
        "Status": "COMPLETE",
        "StatusMessage": "Completed creation of Profile to DNS Firewall rule group
 association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProfileResourceAssociation。

get-profile

以下代码示例演示了如何使用 get-profile。

Amazon CLI

获取有关配置文件的信息

以下 get-profile 返回有关指定配置文件的信息。

```
aws route53profiles get-profile \
    --profile-id rp-4987774726example
```

输出：

```
{
    "Profile": {
        "Arn": "arn:aws:route53profiles:us-east-1:123456789012:profile/
rp-4987774726example",
        "ClientToken": "0cbc5ae7-4921-4204-bea9-EXAMPLE11111",
        "CreationTime": 1710851044.288,
        "Id": "rp-4987774726example",
        "ModificationTime": 1710851044.288,
        "Name": "test",
        "OwnerId": "123456789012",
        "ShareStatus": "NOT_SHARED",
        "Status": "COMPLETE",
        "StatusMessage": "Created Profile"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetProfile。

list-profile-associations

以下代码示例演示了如何使用 list-profile-associations。

Amazon CLI

列出配置文件关联

以下 list-profile-associations 列出了您的 Amazon 账户中的配置文件关联。

```
aws route53profiles list-profile-associations
```

输出：

```
{
    "ProfileAssociations": [
        {
            "CreationTime": 1709338817.148,
            "Id": "rpassoc-489ce212fexample",
            "ModificationTime": 1709338974.772,
            "Name": "test-association",
            "OwnerId": "123456789012",
            "ProfileId": "rp-4987774726example",
            "ResourceId": "vpc-0af3b96b3example",
            "Status": "COMPLETE",
            "StatusMessage": "Created Profile Association"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProfileAssociations。

list-profile-resource-associations

以下代码示例演示了如何使用 list-profile-resource-associations。

Amazon CLI

列出配置文件资源关联

以下 list-profile-resource-associations 列出了指定配置文件的配置文件资源关联。

```
aws route53profiles list-profile-resource-associations \
    --profile-id rp-4987774726example
```

输出：

```
{
     "ProfileResourceAssociations": [
         {
             "CreationTime": 1710851216.613,
```

```
            "Id": "rpr-001913120a7example",
            "ModificationTime": 1710851216.613,
            "Name": "test-resource-association",
            "OwnerId": "123456789012",
            "ProfileId": "rp-4987774726example",
            "ResourceArn": "arn:aws:route53resolver:us-
east-1:123456789012:firewall-rule-group/rslvr-frg-cfe7f72example",
            "ResourceProperties": "{\"priority\":102}",
            "ResourceType": "FIREWALL_RULE_GROUP",
            "Status": "COMPLETE",
            "StatusMessage": "Completed creation of Profile to DNS Firewall rule
 group association"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProfileResourceAssociations。

list-profiles

以下代码示例演示了如何使用 `list-profiles`。

Amazon CLI

列出配置文件

以下 `list-profiles` 列出了您的 Amazon 账户中的配置文件并显示了有关这些配置文件的其他信息。

```
aws route53profiles list-profiles
```

输出：

```
{
    "ProfileSummaries": [
        {
            "Arn": "arn:aws:route53profiles:us-east-1:123456789012:profile/
rp-4987774726example",
            "Id": "rp-4987774726example",
            "Name": "test",
            "ShareStatus": "NOT_SHARED"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProfiles。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出资源标签

以下 `list-tags-for-resource` 列出指定资源的标签。

```
aws route53profiles list-tags-for-resource \
    --resource-arn arn:aws:route53profiles:us-east-1:123456789012:profile/
rp-4987774726example
```

输出：

```
{
    "Tags": {
        "my-key-2": "my-value-2",
        "my-key-1": "my-value-1"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`update-profile-resource-association`

以下代码示例演示了如何使用 `update-profile-resource-association`。

Amazon CLI

更新与配置文件关联的资源

以下 `update-profile-resource-association` 更新了与配置文件关联的 DNS 防火墙规则组的优先级。

```
aws route53profiles update-profile-resource-association \
    --profile-resource-association-id rpr-001913120a7example \
    --resource-properties "{\"priority\": 105}"
```

输出：

```
{
    "ProfileResourceAssociation": {
        "CreationTime": 1710851216.613,
        "Id": "rpr-001913120a7example",
        "ModificationTime": 1710852303.798,
        "Name": "test-resource-association",
        "OwnerId": "123456789012",
        "ProfileId": "rp-4987774726example",
        "ResourceArn": "arn:aws:route53resolver:us-east-1:123456789012:firewall-
rule-group/rslvr-frg-cfe7f72example",
        "ResourceProperties": "{\"priority\":105}",
        "ResourceType": "FIREWALL_RULE_GROUP",
        "Status": "UPDATING",
        "StatusMessage": "Updating the Profile to DNS Firewall rule group
 association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProfileResourceAssociation。

使用 Amazon CLI 的 Route 53 Resolver 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Route 53 Resolver 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-firewall-rule-group`

以下代码示例演示了如何使用 `associate-firewall-rule-group`。

Amazon CLI

将防火墙规则组与 VPC 关联

以下 `associate-firewall-rule-group` 示例将 DNS 防火墙规则组与 Amazon VPC 相关联。

```
aws route53resolver associate-firewall-rule-group \
    --name test-association \
    --firewall-rule-group-id rslvr-frg-47f93271fexample \
    --vpc-id vpc-31e92222 \
    --priority 101
```

输出：

```
{
    "FirewallRuleGroupAssociation": {
        "Id": "rslvr-frgassoc-57e8873d7example",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-
association/rslvr-frgassoc-57e8873d7example",
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "VpcId": "vpc-31e92222",
        "Name": "test-association",
        "Priority": 101,
        "MutationProtection": "DISABLED",
        "Status": "UPDATING",
        "StatusMessage": "Creating Firewall Rule Group Association",
        "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:47:48.755768Z",
        "ModificationTime": "2021-05-25T21:47:48.755768Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 VPC 与 Route 53 Resolver DNS Firewall 规则组之间的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateFirewallRuleGroup。

associate-resolver-endpoint-ip-address

以下代码示例演示了如何使用 associate-resolver-endpoint-ip-address。

Amazon CLI

将另一个 IP 地址与 Resolver 端点相关联

以下 associate-resolver-endpoint-ip-address 示例将另一个 IP 地址与入站 Resolver 端点相关联。如果您仅指定子网 ID，而 --ip-address 参数中省略了 IP 地址，Resolver 会从指定子网中的可用 IP 地址中选择一个 IP 地址。

```
aws route53resolver associate-resolver-endpoint-ip-address \
    --resolver-endpoint-id rslvr-in-497098ad5example \
    --ip-address="SubnetId=subnet-12d8exam,Ip=192.0.2.118"
```

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-in-497098ad5example",
        "CreatorRequestId": "AWSConsole.25.0123456789",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-497098ad5example",
        "Name": "my-inbound-endpoint",
        "SecurityGroupIds": [
            "sg-05cd7b25d6example"
        ],
        "Direction": "INBOUND",
        "IpAddressCount": 3,
        "HostVPCId": "vpc-304bexam",
        "Status": "UPDATING",
        "StatusMessage": "Updating the Resolver Endpoint",
        "CreationTime": "2020-01-02T23:25:45.538Z",
        "ModificationTime": "2020-01-02T23:25:45.538Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的创建或编辑入站端点时指定的值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateResolverEndpointIpAddress。

associate-resolver-rule

以下代码示例演示了如何使用 associate-resolver-rule。

Amazon CLI

将 Resolver 规则与 VPC 关联

以下 associate-resolver-rule 示例将 Resolver 规则组与 Amazon VPC 相关联。运行命令后，Resolver 开始根据规则中的设置（例如所转发查询的域名）将 DNS 查询转发到您的网络。

```
aws route53resolver associate-resolver-rule \
    --name my-resolver-rule-association \
    --resolver-rule-id rslvr-rr-42b60677c0example \
    --vpc-id vpc-304bexam
```

输出：

```
{
    "ResolverRuleAssociation": {
        "Id": "rslvr-rrassoc-d61cbb2c8bexample",
        "ResolverRuleId": "rslvr-rr-42b60677c0example",
        "Name": "my-resolver-rule-association",
        "VPCId": "vpc-304bexam",
        "Status": "CREATING",
        "StatusMessage": "[Trace id: 1-5dc5a8fa-ec2cc480d2ef07617example] Creating
 the association."
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的将出站 DNS 查询转发到您的网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateResolverRule。

create-firewall-domain-list

以下代码示例演示了如何使用 create-firewall-domain-list。

Amazon CLI

创建 Route 53 Resolver DNS Firewall 域列表

以下 create-firewall-domain-list 示例在您的 Amazon 账户中创建名为 test 的 Route 53 Resolver DNS Firewall 域列表。

```
aws route53resolver create-firewall-domain-list \
    --creator-request-id my-request-id \
    --name test
```

输出：

```
{
    "FirewallDomainList": {
        "Id": "rslvr-fdl-d61cbb2cbexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/
rslvr-fdl-d61cbb2cbexample",
        "Name": "test",
        "DomainCount": 0,
        "Status": "COMPLETE",
        "StatusMessage": "Created Firewall Domain List",
        "CreatorRequestId": "my-request-id",
        "CreationTime": "2021-05-25T15:55:51.115365Z",
        "ModificationTime": "2021-05-25T15:55:51.115365Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFirewallDomainList。

create-firewall-rule-group

以下代码示例演示了如何使用 create-firewall-rule-group。

Amazon CLI

创建防火墙规则组

以下 create-firewall-rule-group 示例创建了 DNS 防火墙规则组。

```
aws route53resolver create-firewall-rule-group \
    --creator-request-id my-request-id \
    --name test
```

输出：

```
{
    "FirewallRuleGroup": {
        "Id": "rslvr-frg-47f93271fexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/
rslvr-frg-47f93271fexample",
        "Name": "test",
        "RuleCount": 0,
        "Status": "COMPLETE",
        "StatusMessage": "Created Firewall Rule Group",
        "OwnerId": "123456789012",
        "CreatorRequestId": "my-request-id",
        "ShareStatus": "NOT_SHARED",
        "CreationTime": "2021-05-25T18:59:26.490017Z",
        "ModificationTime": "2021-05-25T18:59:26.490017Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFirewallRuleGroup。

create-firewall-rule

以下代码示例演示了如何使用 create-firewall-rule。

Amazon CLI

创建防火墙规则

以下 create-firewall-rule 示例在 DNS 防火墙规则中为 DNS 防火墙域列表中列出的域创建了防火墙规则。

```
aws route53resolver create-firewall-rule \
    --name allow-rule \
    --firewall-rule-group-id rslvr-frg-47f93271fexample \
    --firewall-domain-list-id rslvr-fdl-9e956e9ffexample \
    --priority 101 \
    --action ALLOW
```

输出：

```
{
    "FirewallRule": {
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
        "Name": "allow-rule",
        "Priority": 101,
        "Action": "ALLOW",
        "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:44:00.346093Z",
        "ModificationTime": "2021-05-25T21:44:00.346093Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFirewallRule。

`create-resolver-endpoint`

以下代码示例演示了如何使用 `create-resolver-endpoint`。

Amazon CLI

创建入站 Resolver 端点

以下 `create-resolver-endpoint` 示例创建了入站 Resolver 端点。您可以使用相同的命令来创建入站和出站端点。

aws route53resolver create-resolver-endpoint --name my-inbound-endpoint --creator-request-id 2020-01-01-18:47 --security-group-ids "sg-f62bexam" --direction INBOUND --ip-addresses SubnetId=subnet-ba47exam,Ip=192.0.2.255 SubnetId=subnet-12d8exam,Ip=192.0.2.254

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-in-f9ab8a03f1example",
        "CreatorRequestId": "2020-01-01-18:47",
```

```
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-f9ab8a03f1example",
        "Name": "my-inbound-endpoint",
        "SecurityGroupIds": [
            "sg-f62bexam"
        ],
        "Direction": "INBOUND",
        "IpAddressCount": 2,
        "HostVPCId": "vpc-304examp",
        "Status": "CREATING",
        "StatusMessage": "[Trace id: 1-5dc1ff84-f3477826e4a190025example] Creating
 the Resolver Endpoint",
        "CreationTime": "2020-01-01T23:02:29.583Z",
        "ModificationTime": "2020-01-01T23:02:29.583Z"
    }
}
```

创建出站 Resolver 端点

以下 `create-resolver-endpoint` 示例使用 JSON 格式文档 `create-outbound-resolver-endpoint.json` 中的值创建出站 Resolver 端点。

```
aws route53resolver create-resolver-endpoint \
    --cli-input-json file://c:\temp\create-outbound-resolver-endpoint.json
```

的内容`create-outbound-resolver-endpoint.json`：

```
{
   "CreatorRequestId": "2020-01-01-18:47",
   "Direction": "OUTBOUND",
   "IpAddresses": [
      {
         "Ip": "192.0.2.255",
         "SubnetId": "subnet-ba47exam"
      },
      {
         "Ip": "192.0.2.254",
         "SubnetId": "subnet-12d8exam"
      }
   ],
   "Name": "my-outbound-endpoint",
   "SecurityGroupIds": [ "sg-05cd7b25d6example" ],
```

```
   "Tags": [
      {
         "Key": "my-key-name",
         "Value": "my-key-value"
      }
   ]
}
```

有关更多信息，请参阅 Amazon Route 53 Developer Guide 中解析 VPC 与您的网络之间的 DNS 查询。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResolverEndpoint。

create-resolver-rule

以下代码示例演示了如何使用 create-resolver-rule。

Amazon CLI

创建 Resolver 规则

以下 create-resolver-rule 示例创建了 Resolver 转发规则。该规则使用出站端点 rslvr-out-d5e5920e37example 将 example.com 的 DNS 查询转发到 IP 地址 10.24.8.75 和 10.24.8.156。

```
aws route53resolver create-resolver-rule \
    --creator-request-id 2020-01-02-18:47 \
    --domain-name example.com \
    --name my-rule \
    --resolver-endpoint-id rslvr-out-d5e5920e37example \
    --rule-type FORWARD \
    --target-ips "Ip=10.24.8.75" "Ip=10.24.8.156"
```

输出：

```
{
    "ResolverRule": {
        "Status": "COMPLETE",
        "RuleType": "FORWARD",
        "ResolverEndpointId": "rslvr-out-d5e5920e37example",
        "Name": "my-rule",
        "DomainName": "example.com.",
        "CreationTime": "2022-05-10T21:35:30.923187Z",
```

```
        "TargetIps": [
            {
                "Ip": "10.24.8.75",
                "Port": 53
            },
            {
                "Ip": "10.24.8.156",
                "Port": 53
            }
        ],
        "CreatorRequestId": "2022-05-10-16:33",
        "ModificationTime": "2022-05-10T21:35:30.923187Z",
        "ShareStatus": "NOT_SHARED",
        "Arn": "arn:aws:route53resolver:us-east-1:111117012054:resolver-rule/rslvr-
rr-b1e0b905e93611111",
        "OwnerId": "111111111111",
        "Id": "rslvr-rr-rslvr-rr-b1e0b905e93611111",
        "StatusMessage": "[Trace id: 1-22222222-3e56afcc71a3724664f22e24]
 Successfully created Resolver Rule."
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResolverRule。

delete-firewall-domain-list

以下代码示例演示了如何使用 delete-firewall-domain-list。

Amazon CLI

删除 Route 53 Resolver DNS Firewall 域列表

以下 delete-firewall-domain-list 示例在您的 Amazon 账户中删除了名为 test 的 Route 53 Resolver DNS Firewall 域列表。

```
aws route53resolver delete-firewall-domain-list \
    --firewall-domain-list-id rslvr-fdl-9e956e9ffexample
```

输出：

```
{
```

```
    "FirewallDomainList": {
        "Id": "rslvr-fdl-9e956e9ffexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/
rslvr-fdl-9e956e9ffexample",
        "Name": "test",
        "DomainCount": 6,
        "Status": "DELETING",
        "StatusMessage": "Deleting the Firewall Domain List",
        "CreatorRequestId": "my-request-id",
        "CreationTime": "2021-05-25T15:55:51.115365Z",
        "ModificationTime": "2021-05-25T18:58:05.588024Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFirewallDomainList。

delete-firewall-rule-group

以下代码示例演示了如何使用 delete-firewall-rule-group。

Amazon CLI

删除防火墙规则组

以下 delete-firewall-rule-group 示例创建了防火墙规则组。

```
aws route53resolver delete-firewall-rule-group \
    --firewall-rule-group-id rslvr-frg-47f93271fexample
```

输出：

```
{
    "FirewallRuleGroup": {
        "Id": "rslvr-frg-47f93271fexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/
rslvr-frg-47f93271fexample",
        "Name": "test",
        "RuleCount": 0,
        "Status": "UPDATING",
        "StatusMessage": "Updating Firewall Rule Group",
```

```
        "OwnerId": "123456789012",
        "CreatorRequestId": "my-request-id",
        "ShareStatus": "NOT_SHARED",
        "CreationTime": "2021-05-25T18:59:26.490017Z",
        "ModificationTime": "2021-05-25T21:51:53.028688Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFirewallRuleGroup。

delete-firewall-rule

以下代码示例演示了如何使用 delete-firewall-rule。

Amazon CLI

删除防火墙规则

以下 delete-firewall-rule 示例删除了防火墙规则组。

```
aws route53resolver delete-firewall-rule \
    --firewall-rule-group-id rslvr-frg-47f93271fexample \
    --firewall-domain-list-id rslvr-fdl-9e956e9ffexample
```

输出：

```
{
    "FirewallRule": {
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
        "Name": "allow-rule",
        "Priority": 102,
        "Action": "ALLOW",
        "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:44:00.346093Z",
        "ModificationTime": "2021-05-25T21:45:59.611600Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFirewallRule。

delete-resolver-endpoint

以下代码示例演示了如何使用 delete-resolver-endpoint。

Amazon CLI

删除 Resolver 端点

以下 delete-resolver-endpoint 示例删除了指定的端点。

重要提示：如果您删除入站端点，来自您网络的 DNS 查询将不再转发到在该端点中指定的 VPC 中的 Resolver。如果您删除了出站端点，对于指定了所删除端点的规则，Resolver 会停止将 DNS 查询从您的 VPC 转发到您的网络。

```
aws route53resolver delete-resolver-endpoint \
    --resolver-endpoint-id rslvr-in-497098ad59example
```

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-in-497098ad59example",
        "CreatorRequestId": "AWSConsole.25.157290example",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-497098ad59example",
        "Name": "my-inbound-endpoint",
        "SecurityGroupIds": [
            "sg-05cd7b25d6example"
        ],
        "Direction": "INBOUND",
        "IpAddressCount": 5,
        "HostVPCId": "vpc-304bexam",
        "Status": "DELETING",
        "StatusMessage": "[Trace id: 1-5dc5b658-811b5be0922bbc382example] Deleting
 ResolverEndpoint.",
        "CreationTime": "2020-01-01T23:25:45.538Z",
        "ModificationTime": "2020-01-02T23:25:45.538Z"
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResolverEndpoint。

delete-resolver-rule

以下代码示例演示了如何使用 delete-resolver-rule。

Amazon CLI

删除 Resolver 规则

以下 delete-resolver-rule 示例删除了指定规则。

注意：如果某个规则与任意 VPC 关联，您必须先从 VPC 上解除规则关联，然后才能删除规则。

```
aws route53resolver delete-resolver-rule \
    --resolver-rule-id rslvr-rr-5b3809426bexample
```

输出：

```
{
    "ResolverRule": {
        "Id": "rslvr-rr-5b3809426bexample",
        "CreatorRequestId": "2020-01-03-18:47",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-
rr-5b3809426bexample",
        "DomainName": "zenith.example.com.",
        "Status": "DELETING",
        "StatusMessage": "[Trace id: 1-5dc5e05b-602e67b052cb74f05example] Deleting
 Resolver Rule.",
        "RuleType": "FORWARD",
        "Name": "my-resolver-rule",
        "TargetIps": [
            {
                "Ip": "192.0.2.50",
                "Port": 53
            }
        ],
        "ResolverEndpointId": "rslvr-out-d5e5920e3example",
        "OwnerId": "111122223333",
        "ShareStatus": "NOT_SHARED"
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResolverRule。

`disassociate-firewall-rule-group`

以下代码示例演示了如何使用 `disassociate-firewall-rule-group`。

Amazon CLI

取消防火墙规则组与 VPC 的关联

以下 `disassociate-firewall-rule-group` 示例取消了 DNS 防火墙规则组与 Amazon VPC 的关联。

```
aws route53resolver disassociate-firewall-rule-group \
    --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example
```

输出：

```
{
    "FirewallRuleGroupAssociation": {
        "Id": "rslvr-frgassoc-57e8873d7example",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-
association/rslvr-frgassoc-57e8873d7example",
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "VpcId": "vpc-31e92222",
        "Name": "test-association",
        "Priority": 103,
        "MutationProtection": "DISABLED",
        "Status": "DELETING",
        "StatusMessage": "Deleting the Firewall Rule Group Association",
        "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:47:48.755768Z",
        "ModificationTime": "2021-05-25T21:51:02.377887Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 VPC 与 Route 53 Resolver DNS Firewall 规则组之间的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateFirewallRuleGroup。

disassociate-resolver-endpoint-ip-address

以下代码示例演示了如何使用 disassociate-resolver-endpoint-ip-address。

Amazon CLI

取消 IP 地址与 Resolver 端点的关联

以下 disassociate-resolver-endpoint-ip-address 示例从指定的 Resolver 入站或出站端点中删除了 IP 地址。

注意：一个端点必须具有至少两个 IP 地址。如果一个端点当前只有两个 IP 地址，并且您希望将一个地址替换为另一个地址，则必须先使用 associate-resolver-endpoint-ip-address 来关联新的 IP 地址。然后，您可以取消其中一个原始 IP 地址与端点的关联。

```
aws route53resolver disassociate-resolver-endpoint-ip-address \
    --resolver-endpoint-id rslvr-in-f9ab8a03f1example \
    --ip-address="SubnetId=subnet-12d8a459,Ip=172.31.40.121"
```

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-in-f9ab8a03f1example",
        "CreatorRequestId": "2020-01-01-18:47",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-f9ab8a03f1example",
        "Name": "my-inbound-endpoint",
        "SecurityGroupIds": [
            "sg-f62bexam"
        ],
        "Direction": "INBOUND",
        "IpAddressCount": 3,
        "HostVPCId": "vpc-304bexam",
        "Status": "UPDATING",
        "StatusMessage": "Updating the Resolver Endpoint",
        "CreationTime": "2020-01-01T23:02:29.583Z",
        "ModificationTime": "2020-01-05T23:02:29.583Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateResolverEndpointIpAddress。

disassociate-resolver-rule

以下代码示例演示了如何使用 disassociate-resolver-rule。

Amazon CLI

取消 Resolver 规则与 Amazon VPC 的关联

以下 disassociate-resolver-rule 示例删除了指定的 Resolver 规则与指定的 VPC 之间的关联。在下列情况下，您可以取消规则与 VPC 的关联：

对于源自此 VPC 中的 DNS 查询，您希望 Resolver 停止将对规则中指定域名的查询转发到您的网络。如果规则当前与一个或多个 VPC 关联，您必须先从所有 VPC 上解除规则的关联，然后才能删除规则。

```
aws route53resolver disassociate-resolver-rule \
    --resolver-rule-id rslvr-rr-4955cb98ceexample \
    --vpc-id vpc-304bexam
```

输出：

```
{
    "ResolverRuleAssociation": {
        "Id": "rslvr-rrassoc-322f4e8b9cexample",
        "ResolverRuleId": "rslvr-rr-4955cb98ceexample",
        "Name": "my-resolver-rule-association",
        "VPCId": "vpc-304bexam",
        "Status": "DELETING",
        "StatusMessage": "[Trace id: 1-5dc5ffa2-a26c38004c1f94006example] Deleting
 Association"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateResolverRule。

get-firewall-config

以下代码示例演示了如何使用 get-firewall-config。

Amazon CLI

获取 VPC 的防火墙配置

以下 get-firewall-config 示例检索指定 VPC 的 DNS 防火墙行为。

```
aws route53resolver get-firewall-config \
    --resource-id vpc-31e92222
```

输出：

```
{
    "FirewallConfig": {
        "Id": "rslvr-fc-86016850cexample",
        "ResourceId": "vpc-31e9222",
        "OwnerId": "123456789012",
        "FirewallFailOpen": "DISABLED"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 DNS 防火墙 VPC 配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFirewallConfig。

get-firewall-domain-list

以下代码示例演示了如何使用 get-firewall-domain-list。

Amazon CLI

获取 Route 53 Resolver DNS Firewall 域列表

以下 get-firewall-domain-list 示例使用您指定 ID 检索域列表。

```
aws route53resolver get-firewall-domain-list \
    --firewall-domain-list-id rslvr-fdl-42b60677cexample
```

输出：

```
{
    "FirewallDomainList": {
        "Id": "rslvr-fdl-9e956e9ffexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123457689012:firewall-domain-list/
rslvr-fdl-42b60677cexample",
        "Name": "test",
        "DomainCount": 0,
```

```
        "Status": "COMPLETE",
        "StatusMessage": "Created Firewall Domain List",
        "CreatorRequestId": "my-request-id",
        "CreationTime": "2021-05-25T15:55:51.115365Z",
        "ModificationTime": "2021-05-25T15:55:51.115365Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFirewallDomainList。

get-firewall-rule-group-association

以下代码示例演示了如何使用 get-firewall-rule-group-association。

Amazon CLI

获取防火墙规则组关联

以下 get-firewall-rule-group-association 示例检索防火墙规则组关联。

```
aws route53resolver get-firewall-rule-group-association \
    --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example
```

输出：

```
{
    "FirewallRuleGroupAssociation": {
        "Id": "rslvr-frgassoc-57e8873d7example",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-
association/rslvr-frgassoc-57e8873d7example",
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "VpcId": "vpc-31e92222",
        "Name": "test-association",
        "Priority": 101,
        "MutationProtection": "DISABLED",
        "Status": "COMPLETE",
        "StatusMessage": "Finished rule group association update",
        "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:47:48.755768Z",
        "ModificationTime": "2021-05-25T21:47:48.755768Z"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 VPC 与 Route 53 Resolver DNS Firewall 规则组之间的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFirewallRuleGroupAssociation。

get-firewall-rule-group-policy

以下代码示例演示了如何使用 get-firewall-rule-group-policy。

Amazon CLI

获取 Amazon IAM 策略

以下 get-firewall-rule-group-policy 示例获取了用于共享指定规则组的 Amazon Identity and Access Management（Amazon IAM）策略。

```
aws route53resolver get-firewall-rule-group-policy \
    --arn arn:aws:route53resolver:us-west-2:AWS_ACCOUNT_ID:firewall-rule-group/
rslvr-frg-47f93271fexample
```

输出：

```
{
    "FirewallRuleGroupPolicy": "{\"Version\":\"2012-10-17\",
\"Statement\":[{\"Sid\":\"test\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"arn:aws:iam::AWS_ACCOUNT_ID:root\"},\"Action\":
[\"route53resolver:GetFirewallRuleGroup\",\"route53resolver:ListFirewallRuleGroups
\"],\"Resource\":\"arn:aws:route53resolver:us-east-1:AWS_ACCOUNT_ID:firewall-rule-
group/rslvr-frg-47f93271fexample\"}]}"
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFirewallRuleGroupPolicy。

get-firewall-rule-group

以下代码示例演示了如何使用 get-firewall-rule-group。

Amazon CLI

获取防火墙规则组

以下 get-firewall-rule-group 示例使用您提供的 ID 检索有关 DNS 防火墙规则组的信息。

```
aws route53resolver get-firewall-rule-group \
    --firewall-rule-group-id rslvr-frg-47f93271fexample
```

输出：

```
{
    "FirewallRuleGroup": {
        "Id": "rslvr-frg-47f93271fexample",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/
rslvr-frg-47f93271fexample",
        "Name": "test",
        "RuleCount": 0,
        "Status": "COMPLETE",
        "StatusMessage": "Created Firewall Rule Group",
        "OwnerId": "123456789012",
        "CreatorRequestId": "my-request-id",
        "ShareStatus": "NOT_SHARED",
        "CreationTime": "2021-05-25T18:59:26.490017Z",
        "ModificationTime": "2021-05-25T18:59:26.490017Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFirewallRuleGroup。

get-resolver-endpoint

以下代码示例演示了如何使用 get-resolver-endpoint。

Amazon CLI

获取有关 Resolver 端点的信息

以下 get-resolver-endpoint 示例显示了指定出站端点的详细信息。通过指定适用的端点 ID，可同时对入站和出站端点使用 get-resolver-endpoint。

```
aws route53resolver get-resolver-endpoint \
    --resolver-endpoint-id rslvr-out-d5e5920e37example
```

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-out-d5e5920e37example",
        "CreatorRequestId": "2020-01-01-18:47",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-out-d5e5920e37example",
        "Name": "my-outbound-endpoint",
        "SecurityGroupIds": [
            "sg-05cd7b25d6example"
        ],
        "Direction": "OUTBOUND",
        "IpAddressCount": 2,
        "HostVPCId": "vpc-304bexam",
        "Status": "OPERATIONAL",
        "StatusMessage": "This Resolver Endpoint is operational.",
        "CreationTime": "2020-01-01T23:50:50.979Z",
        "ModificationTime": "2020-01-02T23:50:50.979Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的创建或编辑入站端点时指定的值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResolverEndpoint。

get-resolver-rule-association

以下代码示例演示了如何使用 get-resolver-rule-association。

Amazon CLI

获取有关 Resolver 规则与 VPC 之间的关联的信息

以下 get-resolver-rule-association 示例显示有关指定 Resolver 规则与 VPC 之间的关联的详细信息。您可以使用 associate-resolver-rule 将 Resolver 规则关联到 VPC。

```
aws route53resolver get-resolver-rule-association \
```

```
    --resolver-rule-association-id rslvr-rrassoc-d61cbb2c8bexample
```

输出：

```
{
    "ResolverRuleAssociation": {
        "Id": "rslvr-rrassoc-d61cbb2c8bexample",
        "ResolverRuleId": "rslvr-rr-42b60677c0example",
        "Name": "my-resolver-rule-association",
        "VPCId": "vpc-304bexam",
        "Status": "COMPLETE",
        "StatusMessage": ""
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResolverRuleAssociation。

get-resolver-rule

以下代码示例演示了如何使用 get-resolver-rule。

Amazon CLI

获取有关 Resolver 规则的信息

以下 get-resolver-rule 示例显示有关指定 Resolver 规则的详细信息，例如，规则为其转发 DNS 查询的域名以及与规则关联的 Resolver 出站端点的 ID。

```
aws route53resolver get-resolver-rule \
    --resolver-rule-id rslvr-rr-42b60677c0example
```

输出：

```
{
    "ResolverRule": {
        "Id": "rslvr-rr-42b60677c0example",
        "CreatorRequestId": "2020-01-01-18:47",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-
rr-42b60677c0example",
        "DomainName": "example.com.",
        "Status": "COMPLETE",
```

```
        "StatusMessage": "[Trace id: 1-5dc4b177-ff1d9d001a0f80005example]
 Successfully created Resolver Rule.",
        "RuleType": "FORWARD",
        "Name": "my-rule",
        "TargetIps": [
            {
                "Ip": "192.0.2.45",
                "Port": 53
            }
        ],
        "ResolverEndpointId": "rslvr-out-d5e5920e37example",
        "OwnerId": "111122223333",
        "ShareStatus": "NOT_SHARED"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的创建或编辑规则时指定的值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResolverRule。

import-firewall-domains

以下代码示例演示了如何使用 import-firewall-domains。

Amazon CLI

将域导入域列表

以下 import-firewall-domains 示例将一组域从文件导入到您指定的 DNS 防火墙域列表中。

```
aws route53resolver import-firewall-domains \
    --firewall-domain-list-id rslvr-fdl-d61cbb2cbexample \
    --operation REPLACE \
    --domain-file-url s3://PATH/TO/YOUR/FILE
```

输出：

```
{
    "Id": "rslvr-fdl-d61cbb2cbexample",
    "Name": "test",
    "Status": "IMPORTING",
    "StatusMessage": "Importing domains from provided file."
```

```
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportFirewallDomains。

list-firewall-configs

以下代码示例演示了如何使用 list-firewall-configs。

Amazon CLI

列出防火墙配置

以下 list-firewall-configs 示例列出您的 DNS 防火墙配置。

```
aws route53resolver list-firewall-configs
```

输出：

```
{
    "FirewallConfigs": [
        {
            "Id": "rslvr-fc-86016850cexample",
            "ResourceId": "vpc-31e92222",
            "OwnerId": "123456789012",
            "FirewallFailOpen": "DISABLED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 DNS 防火墙 VPC 配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallConfigs。

list-firewall-domain-lists

以下代码示例演示了如何使用 list-firewall-domain-lists。

Amazon CLI

列出所有 Route 53 Resolver DNS Firewall 域列表

以下 list-firewall-domain-lists 示例列出了所有域列表。

```
aws route53resolver list-firewall-domain-lists
```

输出：

```
{
    "FirewallDomainLists": [
        {
            "Id": "rslvr-fdl-2c46f2ecfexample",
            "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-
list/rslvr-fdl-2c46f2ecfexample",
            "Name": "AWSManagedDomainsMalwareDomainList",
            "CreatorRequestId": "AWSManagedDomainsMalwareDomainList",
            "ManagedOwnerName": "Route 53 Resolver DNS Firewall"
        },
        {
            "Id": "rslvr-fdl-aa970e9e1example",
            "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-
list/rslvr-fdl-aa970e9e1example",
            "Name": "AWSManagedDomainsBotnetCommandandControl",
            "CreatorRequestId": "AWSManagedDomainsBotnetCommandandControl",
            "ManagedOwnerName": "Route 53 Resolver DNS Firewall"
        },
        {
            "Id": "rslvr-fdl-42b60677cexample",
            "Arn": "arn:aws:route53resolver:us-west-2:123456789111:firewall-domain-
list/rslvr-fdl-42b60677cexample",
            "Name": "test",
            "CreatorRequestId": "my-request-id"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 Route 53 Resolver DNS Firewall 域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallDomainLists。

list-firewall-domains

以下代码示例演示了如何使用 list-firewall-domains。

Amazon CLI

在域列表中列出域

以下 `list-firewall-domains` 示例列出了您指定的 DNS 防火墙域列表中的域。

```
aws route53resolver list-firewall-domains \
    --firewall-domain-list-id rslvr-fdl-d61cbb2cbexample
```

输出：

```
{
    "Domains": [
        "test1.com.",
        "test2.com.",
        "test3.com."
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallDomains。

`list-firewall-rule-group-associations`

以下代码示例演示了如何使用 `list-firewall-rule-group-associations`。

Amazon CLI

列出 DNS 防火墙规则组关联

以下 `list-firewall-rule-group-associations` 示例列出了 DNS 防火墙规则组与 Amazon VPC 的关联。

```
aws route53resolver list-firewall-rule-group-associations
```

输出：

```
{
    "FirewallRuleGroupAssociations": [
```

```
        {
            "Id": "rslvr-frgassoc-57e8873d7example",
            "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-
group-association/rslvr-frgassoc-57e8873d7example",
            "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
            "VpcId": "vpc-31e92222",
            "Name": "test-association",
            "Priority": 101,
            "MutationProtection": "DISABLED",
            "Status": "UPDATING",
            "StatusMessage": "Creating Firewall Rule Group Association",
            "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
            "CreationTime": "2021-05-25T21:47:48.755768Z",
            "ModificationTime": "2021-05-25T21:47:48.755768Z"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 VPC 与 Route 53 Resolver DNS Firewall 规则组之间的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallRuleGroupAssociations。

`list-firewall-rule-groups`

以下代码示例演示了如何使用 `list-firewall-rule-groups`。

Amazon CLI

获取您的防火墙规则组列表

以下 `list-firewall-rule-groups` 示例列出了 DNS 防火墙规则组。

```
aws route53resolver list-firewall-rule-groups
```

输出：

```
{
    "FirewallRuleGroups": [
        {
            "Id": "rslvr-frg-47f93271fexample",
```

```
            "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-
group/rslvr-frg-47f93271fexample",
            "Name": "test",
            "OwnerId": "123456789012",
            "CreatorRequestId": "my-request-id",
            "ShareStatus": "NOT_SHARED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallRuleGroups。

list-firewall-rules

以下代码示例演示了如何使用 list-firewall-rules。

Amazon CLI

列出防火墙规则

以下 list-firewall-rules 示例列出了防火墙规则组中的所有 DNS 防火墙规则。

```
aws route53resolver list-firewall-rules \
    --firewall-rule-group-id rslvr-frg-47f93271fexample
```

输出：

```
{
    "FirewallRules": [
        {
            "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
            "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
            "Name": "allow-rule",
            "Priority": 101,
            "Action": "ALLOW",
            "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
            "CreationTime": "2021-05-25T21:44:00.346093Z",
            "ModificationTime": "2021-05-25T21:44:00.346093Z"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFirewallRules。

`list-resolver-endpoint-ip-addresses`

以下代码示例演示了如何使用 `list-resolver-endpoint-ip-addresses`。

Amazon CLI

列出指定入站或出站端点的 IP 地址

以下 `list-resolver-endpoint-ip-addresses` 示例列出了与入站端点 `rslvr-in-f9ab8a03f1example` 关联的 IP 地址的相关信息。也可以通过指定适用的端点 ID，对出站端点使用 `list-resolver-endpoint-ip-addresses`。

```
aws route53resolver list-resolver-endpoint-ip-addresses \
    --resolver-endpoint-id rslvr-in-f9ab8a03f1example
```

输出：

```
{
    "MaxResults": 10,
    "IpAddresses": [
        {
            "IpId": "rni-1de60cdbfeexample",
            "SubnetId": "subnet-ba47exam",
            "Ip": "192.0.2.44",
            "Status": "ATTACHED",
            "StatusMessage": "This IP address is operational.",
            "CreationTime": "2020-01-03T23:02:29.587Z",
            "ModificationTime": "2020-01-03T23:03:05.555Z"
        },
        {
            "IpId": "rni-aac7085e38example",
            "SubnetId": "subnet-12d8exam",
            "Ip": "192.0.2.45",
            "Status": "ATTACHED",
            "StatusMessage": "This IP address is operational.",
```

```
            "CreationTime": "2020-01-03T23:02:29.593Z",
            "ModificationTime": "2020-01-03T23:02:55.060Z"
        }
    ]
}
```

有关输出中值的更多信息，请参阅《Amazon Route 53 开发者指南》中的创建或编辑入站端点时指定的值或创建或编辑出站端点时指定的值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResolverEndpointIpAddresses。

`list-resolver-endpoints`

以下代码示例演示了如何使用 `list-resolver-endpoints`。

Amazon CLI

列出 Amazon 区域中的 Resolver 端点

以下 `list-resolver-endpoints` 示例列出了当前账户中存在的入站和出站 Resolver 端点。

```
aws route53resolver list-resolver-endpoints
```

输出：

```
{
    "MaxResults": 10,
    "ResolverEndpoints": [
        {
            "Id": "rslvr-in-497098ad59example",
            "CreatorRequestId": "2020-01-01-18:47",
            "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-
endpoint/rslvr-in-497098ad59example",
            "Name": "my-inbound-endpoint",
            "SecurityGroupIds": [
                "sg-05cd7b25d6example"
            ],
            "Direction": "INBOUND",
            "IpAddressCount": 2,
            "HostVPCId": "vpc-304bexam",
            "Status": "OPERATIONAL",
            "StatusMessage": "This Resolver Endpoint is operational.",
            "CreationTime": "2020-01-01T23:25:45.538Z",
```

```
            "ModificationTime": "2020-01-01T23:25:45.538Z"
        },
        {
            "Id": "rslvr-out-d5e5920e37example",
            "CreatorRequestId": "2020-01-01-18:48",
            "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-
endpoint/rslvr-out-d5e5920e37example",
            "Name": "my-outbound-endpoint",
            "SecurityGroupIds": [
                "sg-05cd7b25d6example"
            ],
            "Direction": "OUTBOUND",
            "IpAddressCount": 2,
            "HostVPCId": "vpc-304bexam",
            "Status": "OPERATIONAL",
            "StatusMessage": "This Resolver Endpoint is operational.",
            "CreationTime": "2020-01-01T23:50:50.979Z",
            "ModificationTime": "2020-01-01T23:50:50.979Z"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResolverEndpoints。

`list-resolver-rule-associations`

以下代码示例演示了如何使用 `list-resolver-rule-associations`。

Amazon CLI

列出 Resolver 规则与 VPC 之间的所有关联

以下 `list-resolver-rule-associations` 示例列出了当前 Amazon 账户中的 Resolver 规则与 VPC 之间的关联。

```
aws route53resolver list-resolver-rule-associations
```

输出：

```
{
    "MaxResults": 30,
    "ResolverRuleAssociations": [
```

```
        {
            "Id": "rslvr-autodefined-assoc-vpc-304bexam-internet-resolver",
            "ResolverRuleId": "rslvr-autodefined-rr-internet-resolver",
            "Name": "System Rule Association",
            "VPCId": "vpc-304bexam",
            "Status": "COMPLETE",
            "StatusMessage": ""
        },
        {
            "Id": "rslvr-rrassoc-d61cbb2c8bexample",
            "ResolverRuleId": "rslvr-rr-42b60677c0example",
            "Name": "my-resolver-rule-association",
            "VPCId": "vpc-304bexam",
            "Status": "COMPLETE",
            "StatusMessage": ""
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 Route 53 Resolver 如何将 DNS 查询从您的 VPC 转发到您的网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResolverRuleAssociations。

`list-resolver-rules`

以下代码示例演示了如何使用 `list-resolver-rules`。

Amazon CLI

列出 Resolver 规则

以下 `list-resolver-rules` 示例列出了当前 Amazon 账户中的所有 Resolver 规则。

```
aws route53resolver list-resolver-rules
```

输出：

```
{
    "MaxResults": 30,
    "ResolverRules": [
        {
```

```
            "Id": "rslvr-autodefined-rr-internet-resolver",
            "CreatorRequestId": "",
            "Arn": "arn:aws:route53resolver:us-west-2::autodefined-rule/rslvr-
autodefined-rr-internet-resolver",
            "DomainName": ".",
            "Status": "COMPLETE",
            "RuleType": "RECURSIVE",
            "Name": "Internet Resolver",
            "OwnerId": "Route 53 Resolver",
            "ShareStatus": "NOT_SHARED"
        },
        {
            "Id": "rslvr-rr-42b60677c0example",
            "CreatorRequestId": "2020-01-01-18:47",
            "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0bc4e299",
            "DomainName": "example.com.",
            "Status": "COMPLETE",
            "StatusMessage": "[Trace id: 1-5dc4b177-ff1d9d001a0f80005example]
 Successfully created Resolver Rule.",
            "RuleType": "FORWARD",
            "Name": "my-rule",
            "TargetIps": [
                {
                    "Ip": "192.0.2.45",
                    "Port": 53
                }
            ],
            "ResolverEndpointId": "rslvr-out-d5e5920e37example",
            "OwnerId": "111122223333",
            "ShareStatus": "NOT_SHARED"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 Route 53 Resolver 如何将 DNS 查询从您的 VPC 转发到您的网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResolverRules。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出 Resolver 资源的标签

以下 `list-tags-for-resource` 示例列出分配给指定 Resolver 规则的标签。

```
aws route53resolver list-tags-for-resource \
    --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0example"
```

输出：

```
{
    "Tags": [
        {
            "Key": "my-key-1",
            "Value": "my-value-1"
        },
        {
            "Key": "my-key-2",
            "Value": "my-value-2"
        }
    ]
}
```

有关将标签用于成本分配的信息，请参阅《Amazon 账单和成本管理用户指南》中的使用成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

put-firewall-rule-group-policy

以下代码示例演示了如何使用 `put-firewall-rule-group-policy`。

Amazon CLI

附加 Amazon IAM 策略以共享防火墙规则组策略

以下 `put-firewall-rule-group-policy` 示例附加用于共享指定规则组的 Amazon Identity and Access Management（Amazon IAM）策略。

```
aws route53resolver put-firewall-rule-group-policy \
```

```
    --firewall-rule-group-policy "{\"Version\":\"2012-10-17\",
\"Statement\":[{\"Sid\":\"test\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"arn:aws:iam::AWS_ACCOUNT_ID:root\"},\"Action\":
[\"route53resolver:GetFirewallRuleGroup\",\"route53resolver:ListFirewallRuleGroups
\"],\"Resource\":\"arn:aws:route53resolver:us-east-1:AWS_ACCOUNT_ID:firewall-rule-
group/rslvr-frg-47f93271fexample\"}]}"
```

输出：

```
{
    "ReturnValue": true
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutFirewallRuleGroupPolicy。

put-resolver-rule-policy

以下代码示例演示了如何使用 put-resolver-rule-policy。

Amazon CLI

与其他 Amazon 账户共享 Resolver 规则

以下 put-resolver-rule-policy 示例指定要与其他 Amazon 账户共享的 Resolver 规则、要与之共享规则的账户，以及您希望该账户能够对规则执行的规则相关操作。

注意：您必须使用创建规则的同一账户的凭证运行此命令。

```
aws route53resolver put-resolver-rule-policy \
    --region us-east-1 \
    --arn "arn:aws:route53resolver:us-east-1:111122223333:resolver-rule/rslvr-
rr-42b60677c0example" \
    --resolver-rule-policy "{\"Version\": \"2012-10-17\", \
        \"Statement\": [ { \
        \"Effect\" : \"Allow\", \
        \"Principal\" : {\"AWS\" : \"444455556666\" }, \
        \"Action\" : [ \
            \"route53resolver:GetResolverRule\", \
            \"route53resolver:AssociateResolverRule\", \
```

```
            \"route53resolver:DisassociateResolverRule\", \
            \"route53resolver:ListResolverRules\", \
            \"route53resolver:ListResolverRuleAssociations\" ], \
        \"Resource\" : [ \"arn:aws:route53resolver:us-east-1:111122223333:resolver-
rule/rslvr-rr-42b60677c0example\" ] } ] }"
```

输出：

```
{
    "ReturnValue": true
}
```

运行 put-resolver-rule-policy 后，您可以运行以下两个 Resource Access Manager（RAM）命令。您必须使用要与之共享规则的账户：

get-resource-share-invitations 会返回值 resourceShareInvitationArn。您需要此值才能接受使用共享规则的邀请。accept-resource-share-invitation 将接受使用共享规则的邀请。

有关更多信息，请参阅以下文档：

get-resource-share-invitationsaccept-resource-share-invitations《Amazon Route 53 开发人员指南》中的与其他 Amazon 账户共享转发规则并使用共享规则

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutResolverRulePolicy。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签与 Resolver 资源相关联

以下 tag-resource 示例将两个标签键/值对与指定的 Resolver 规则相关联。

```
aws route53resolver tag-resource \
    --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0example" \
    --tags "Key=my-key-1,Value=my-value-1" "Key=my-key-2,Value=my-value-2"
```

此命令不生成任何输出。

有关将标签用于成本分配的信息，请参阅《Amazon 账单和成本管理用户指南》中的使用成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从 Resolver 资源中删除标签

以下 `untag-resource` 示例从指定的 Resolver 规则中删除了两个标签。

```
aws route53resolver untag-resource \
    --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0example" \
    --tag-keys my-key-1 my-key-2
```

此命令不生成任何输出。要确认标签已被删除，可以使用 list-tags-for-resource。

有关将标签用于成本分配的信息，请参阅《Amazon 账单和成本管理用户指南》中的使用成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

`update-firewall-config`

以下代码示例演示了如何使用 `update-firewall-config`。

Amazon CLI

更新防火墙配置

以下 `update-firewall-config` 示例更新了 DNS 防火墙配置。

```
aws route53resolver update-firewall-config \
    --resource-id vpc-31e92222 \
    --firewall-fail-open DISABLED
```

输出：

```
{
    "FirewallConfig": {
        "Id": "rslvr-fc-86016850cexample",
        "ResourceId": "vpc-31e92222",
        "OwnerId": "123456789012",
        "FirewallFailOpen": "DISABLED"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的 DNS 防火墙 VPC 配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFirewallConfig。

update-firewall-domains

以下代码示例演示了如何使用 update-firewall-domains。

Amazon CLI

更新域列表

以下 update-firewall-domains 示例使用您提供的 ID 将域添加到域列表中。

```
aws route53resolver update-firewall-domains \
    --firewall-domain-list-id rslvr-fdl-42b60677cexampleb \
    --operation ADD \
    --domains test1.com test2.com test3.com
```

输出：

```
{
    "Id": "rslvr-fdl-42b60677cexample",
    "Name": "test",
    "Status": "UPDATING",
    "StatusMessage": "Updating the Firewall Domain List"
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理您自己的域列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFirewallDomains。

update-firewall-rule-group-association

以下代码示例演示了如何使用 update-firewall-rule-group-association。

Amazon CLI

更新防火墙规则组关联

以下 update-firewall-rule-group-association 示例更新了防火墙规则组关联。

```
aws route53resolver update-firewall-rule-group-association \
    --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example \
    --priority 103
```

输出：

```
{
    "FirewallRuleGroupAssociation": {
        "Id": "rslvr-frgassoc-57e8873d7example",
        "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-
association/rslvr-frgassoc-57e8873d7example",
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "VpcId": "vpc-31e92222",
        "Name": "test-association",
        "Priority": 103,
        "MutationProtection": "DISABLED",
        "Status": "UPDATING",
        "StatusMessage": "Updating the Firewall Rule Group Association Attributes",
        "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:47:48.755768Z",
        "ModificationTime": "2021-05-25T21:50:09.272569Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 VPC 与 Route 53 Resolver DNS Firewall 规则组之间的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFirewallRuleGroupAssociation。

update-firewall-rule

以下代码示例演示了如何使用 update-firewall-rule。

Amazon CLI

更新防火墙规则

以下 update-firewall-rule 示例使用您指定的参数更新了防火墙规则。

```
aws route53resolver update-firewall-rule \
    --firewall-rule-group-id rslvr-frg-47f93271fexample \
    --firewall-domain-list-id rslvr-fdl-9e956e9ffexample \
    --priority 102
```

输出：

```
{
    "FirewallRule": {
        "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
        "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
        "Name": "allow-rule",
        "Priority": 102,
        "Action": "ALLOW",
        "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
        "CreationTime": "2021-05-25T21:44:00.346093Z",
        "ModificationTime": "2021-05-25T21:45:59.611600Z"
    }
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的管理 DNS 防火墙中的规则组和规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFirewallRule。

update-resolver-endpoint

以下代码示例演示了如何使用 update-resolver-endpoint。

Amazon CLI

更新 Resolver 端点的名称

以下 update-resolver-endpoint 示例更新了 Resolver 端点的名称。不支持更新其他值。

```
aws route53resolver update-resolver-endpoint \
```

```
    --resolver-endpoint-id rslvr-in-b5d45e32bdc445f09 \
    --name my-renamed-inbound-endpoint
```

输出：

```
{
    "ResolverEndpoint": {
        "Id": "rslvr-in-b5d45e32bdexample",
        "CreatorRequestId": "2020-01-02-18:48",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-b5d45e32bdexample",
        "Name": "my-renamed-inbound-endpoint",
        "SecurityGroupIds": [
            "sg-f62bexam"
        ],
        "Direction": "INBOUND",
        "IpAddressCount": 2,
        "HostVPCId": "vpc-304bexam",
        "Status": "OPERATIONAL",
        "StatusMessage": "This Resolver Endpoint is operational.",
        "CreationTime": "2020-01-01T18:33:59.265Z",
        "ModificationTime": "2020-01-08T18:33:59.265Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResolverEndpoint。

update-resolver-rule

以下代码示例演示了如何使用 update-resolver-rule。

Amazon CLI

示例 1：更新 Resolver 端点的设置

以下 update-resolver-rule 示例更新了规则的名称、将 DNS 查询转发到的本地网络上的 IP 地址以及用于将查询转发到网络的 Resolver 出站端点的 ID。

注意：TargetIps 的现有值会被覆盖，因此您必须指定在更新后希望规则具有的所有 IP 地址。

```
aws route53resolver update-resolver-rule \
```

```
    --resolver-rule-id rslvr-rr-1247fa64f3example \
    --config Name="my-2nd-rule",TargetIps=[{Ip=192.0.2.45,Port=53},
{Ip=192.0.2.46,Port=53}],ResolverEndpointId=rslvr-out-7b89ed0d25example
```

输出：

```
{
    "ResolverRule": {
        "Id": "rslvr-rr-1247fa64f3example",
        "CreatorRequestId": "2020-01-02-18:47",
        "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-
rr-1247fa64f3example",
        "DomainName": "www.example.com.",
        "Status": "COMPLETE",
        "StatusMessage": "[Trace id: 1-5dcc90b9-8a8ee860aba1ebd89example]
 Successfully updated Resolver Rule.",
        "RuleType": "FORWARD",
        "Name": "my-2nd-rule",
        "TargetIps": [
            {
                "Ip": "192.0.2.45",
                "Port": 53
            },
            {
                "Ip": "192.0.2.46",
                "Port": 53
            }
        ],
        "ResolverEndpointId": "rslvr-out-7b89ed0d25example",
        "OwnerId": "111122223333",
        "ShareStatus": "NOT_SHARED"
    }
}
```

示例 2：使用用于“config”设置的文件更新 Resolver 端点的设置

或者，您可以将 config 设置包含在 JSON 文件中，然后在调用 update-resolver-rule 时指定该文件。

```
aws route53resolver update-resolver-rule \
    --resolver-rule-id rslvr-rr-1247fa64f3example \
    --config file://c:\temp\update-resolver-rule.json
```

update-resolver-rule.json 的内容。

```
{
    "Name": "my-2nd-rule",
    "TargetIps": [
        {
            "Ip": "192.0.2.45",
            "Port": 53
        },
        {
            "Ip": "192.0.2.46",
            "Port": 53
        }
    ],
    "ResolverEndpointId": "rslvr-out-7b89ed0d25example"
}
```

有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的创建或编辑规则时指定的值。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResolverRule。

使用Amazon CLI的 Amazon S3 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon S3 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

abort-multipart-upload

以下代码示例演示了如何使用 abort-multipart-upload。

Amazon CLI

中止指定的分段上传

以下 abort-multipart-upload 命令中止存储桶 amzn-s3-demo-bucket 中键 multipart/01 的分段上传：

```
aws s3api abort-multipart-upload \
    --bucket amzn-s3-demo-bucket \
    --key multipart/01 \
    --upload-
id dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3UR
```

此命令所需的上传 ID 由 create-multipart-upload 输出，也可以使用 list-multipart-uploads 进行检索。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortMultipartUpload。

complete-multipart-upload

以下代码示例演示了如何使用 complete-multipart-upload。

Amazon CLI

以下命令完成存储桶 amzn-s3-demo-bucket 中密钥 multipart/01 的分段上传：

```
aws s3api complete-multipart-upload --multipart-upload file://
mpustruct --bucket amzn-s3-demo-bucket --key 'multipart/01' --upload-
id dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3UR
```

此命令所需的上传 ID 由 create-multipart-upload 输出，也可以使用 list-multipart-uploads 进行检索。

上述命令中的分段上传选项采用 JSON 结构，用于描述分段上传中应重新组合成完整文件的各个部分。在此示例中，file:// 前缀用于从名为 mpustruct 的本地文件夹中的文件加载 JSON 结构。

mpustruct：

```
{
  "Parts": [
```

```
    {
      "ETag": "e868e0f4719e394144ef36531ee6824c",
      "PartNumber": 1
    },
    {
      "ETag": "6bb2b12753d66fe86da4998aa33fffb0",
      "PartNumber": 2
    },
    {
      "ETag": "d0a0112e841abec9c9ec83406f0159c8",
      "PartNumber": 3
    }
  ]
}
```

每次使用 upload-part 命令上传分段时，都会输出每个分段的 ETag 值，也可以通过调用 list-parts 来检索，或者通过对每个分段进行 MD5 校验和执行计算。

输出：

```
{
    "ETag": "\"3944a9f7a4faab7f78788ff6210f63f0-3\"",
    "Bucket": "amzn-s3-demo-bucket",
    "Location": "https://amzn-s3-demo-bucket.s3.amazonaws.com/multipart%2F01",
    "Key": "multipart/01"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CompleteMultipartUpload。

copy-object

以下代码示例演示了如何使用 copy-object。

Amazon CLI

以下命令将对象从 bucket-1 复制到 bucket-2：

```
aws s3api copy-object --copy-source bucket-1/test.txt --key test.txt --
bucket bucket-2
```

输出：

```
{
    "CopyObjectResult": {
        "LastModified": "2015-11-10T01:07:25.000Z",
        "ETag": "\"589c8b79c230a6ecd5a7e1d040a9a030\""
    },
    "VersionId": "YdnYvTCVDqRRFA.NFJjy36p0hxifMlkA"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyObject。

cp

以下代码示例演示了如何使用 cp。

Amazon CLI

示例 1：将本地文件复制到 S3

以下 cp 命令会将单个文件复制到指定的存储桶和密钥：

```
aws s3 cp test.txt s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
upload: test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 2：将本地文件复制到 S3，并设置到期日期

以下 cp 命令会将单个文件复制到指定的存储桶和密钥，并在指定的 ISO 8601 时间戳到期：

```
aws s3 cp test.txt s3://amzn-s3-demo-bucket/test2.txt \
    --expires 2014-10-01T20:30:00Z
```

输出：

```
upload: test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 3：将文件从 S3 复制到 S3

以下 cp 命令会将单个 S3 对象复制到指定的存储桶和密钥：

```
aws s3 cp s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
copy: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 4：将 S3 对象复制到本地文件

以下 cp 命令会将单个对象复制到指定的本地文件：

```
aws s3 cp s3://amzn-s3-demo-bucket/test.txt test2.txt
```

输出：

```
download: s3://amzn-s3-demo-bucket/test.txt to test2.txt
```

示例 5：将 S3 对象从一个存储桶复制到另一个存储桶

以下 cp 命令会将单个对象复制到指定的存储桶，同时保留其原始名称：

```
aws s3 cp s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket2/
```

输出：

```
copy: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket2/test.txt
```

示例 6：将 S3 对象递归复制到本地目录

当与参数 `--recursive` 一起传递时，以下 cp 命令会将指定前缀和存储桶下的所有对象递归复制到指定目录。在此示例中，存储桶 `amzn-s3-demo-bucket` 中包含对象 `test1.txt` 和 `test2.txt`：

```
aws s3 cp s3://amzn-s3-demo-bucket . \
    --recursive
```

输出：

```
download: s3://amzn-s3-demo-bucket/test1.txt to test1.txt
download: s3://amzn-s3-demo-bucket/test2.txt to test2.txt
```

示例 7：将本地文件递归复制到 S3

当与参数 --recursive 一起传递时，以下 cp 命令会将指定目录下的所有文件递归复制到指定的存储桶和前缀，同时使用 --exclude 参数将某些文件排除在外。在此示例中，目录 myDir 中包含文件 test1.txt 和 test2.jpg：

```
aws s3 cp myDir s3://amzn-s3-demo-bucket/ \
    --recursive \
    --exclude "*.jpg"
```

输出：

```
upload: myDir/test1.txt to s3://amzn-s3-demo-bucket/test1.txt
```

示例 8：将 S3 对象递归复制到另一个存储桶

当与参数 --recursive 一起传递时，以下 cp 命令会将指定存储桶下的所有对象递归复制到另一个存储桶，同时使用 --exclude 参数将某些对象排除在外。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 another/test1.txt：

```
aws s3 cp s3://amzn-s3-demo-bucket/ s3://amzn-s3-demo-bucket2/ \
    --recursive \
    --exclude "another/*"
```

输出：

```
copy: s3://amzn-s3-demo-bucket/test1.txt to s3://amzn-s3-demo-bucket2/test1.txt
```

您可以结合 --exclude 和 --include 选项，仅复制与模式匹配的对象，不复制其他对象：

```
aws s3 cp s3://amzn-s3-demo-bucket/logs/ s3://amzn-s3-demo-bucket2/logs/ \
    --recursive \
    --exclude "*" \
    --include "*.log"
```

输出：

```
copy: s3://amzn-s3-demo-bucket/logs/test/test.log to s3://amzn-s3-demo-bucket2/logs/
test/test.log
```

```
copy: s3://amzn-s3-demo-bucket/logs/test3.log to s3://amzn-s3-demo-bucket2/logs/
test3.log
```

示例 9：复制 S3 对象时设置访问控制列表（ACL）

以下 cp 命令会将单个对象复制到指定的存储桶和密钥，同时将 ACL 设置为 public-read-write：

```
aws s3 cp s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket/test2.txt \
    --acl public-read-write
```

输出：

```
copy: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

请注意，如果使用 --acl 选项，请确保所有相关的 IAM 策略都包含 "s3:PutObjectAcl" 操作：

```
aws iam get-user-policy \
    --user-name myuser \
    --policy-name mypolicy
```

输出：

```
{
    "UserName": "myuser",
    "PolicyName": "mypolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": [
                    "s3:PutObject",
                    "s3:PutObjectAcl"
                ],
                "Resource": [
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                ],
                "Effect": "Allow",
                "Sid": "Stmt1234567891234"
            }
```

```
        ]
    }
}
```

示例 10：为 S3 对象授予权限

以下 cp 命令说明如何使用 --grants 选项向所有通过 URI 标识的用户授予读取权限，并向通过 Canonical ID 标识的特定用户授予完全控制权限：

```
aws s3 cp file.txt s3://amzn-s3-demo-bucket/ --grants read=uri=http://
acs.amazonaws.com/groups/global/
AllUsers full=id=79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

输出：

```
upload: file.txt to s3://amzn-s3-demo-bucket/file.txt
```

示例 11：将本地文件流上传到 S3

PowerShell 可能会更改管道输入的编码或向其添加 CRLF。

以下 cp 命令会将本地文件流从标准输入上传到指定的存储桶和密钥：

```
aws s3 cp - s3://amzn-s3-demo-bucket/stream.txt
```

示例 12：将大于 50GB 的本地文件流上传到 S3

以下 cp 命令会将 51GB 的本地文件流从标准输入上传到指定的存储桶和密钥。必须提供 --expected-size 选项，否则当达到 10,000 的默认分段限制时，上传可能会失败：

```
aws s3 cp - s3://amzn-s3-demo-bucket/stream.txt --expected-size 54760833024
```

示例 13：将 S3 对象下载为本地文件流

PowerShell 可能会更改管道或重定向输出的编码或向其添加 CRLF。

以下 cp 命令会将 S3 对象作为流下载到本地标准输出。以流形式下载目前与 --recursive 参数不兼容：

```
aws s3 cp s3://amzn-s3-demo-bucket/stream.txt -
```

示例 14：上传到 S3 接入点

以下 cp 命令会将单个文件（mydoc.txt）上传到密钥（mykey）处的接入点（myaccesspoint）：

```
aws s3 cp mydoc.txt s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/mykey
```

输出：

```
upload: mydoc.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/mykey
```

示例 15：从 S3 接入点下载

以下 cp 命令会将单个对象（mykey）从接入点（myaccesspoint）下载到本地文件（mydoc.txt）：

```
aws s3 cp s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
mykey mydoc.txt
```

输出：

```
download: s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey to
 mydoc.txt
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Cp。

create-bucket

以下代码示例演示了如何使用 create-bucket。

Amazon CLI

示例 1：创建存储桶

以下 create-bucket 示例创建一个名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api create-bucket \
    --bucket amzn-s3-demo-bucket \
```

```
    --region us-east-1
```

输出：

```
{
    "Location": "/amzn-s3-demo-bucket"
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的创建存储桶。

示例 2：创建带有强制拥有者的存储桶

以下 create-bucket 示例创建一个名为 amzn-s3-demo-bucket 的存储桶，该存储桶对于 S3 对象所有权使用强制存储桶拥有者设置。

```
aws s3api create-bucket \
    --bucket amzn-s3-demo-bucket \
    --region us-east-1 \
    --object-ownership BucketOwnerEnforced
```

输出：

```
{
    "Location": "/amzn-s3-demo-bucket"
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的控制对象所有权和禁用 ACL。

示例 3：在“us-east-1”区域之外创建存储桶

以下 create-bucket 示例在 eu-west-1 区域中创建名为 amzn-s3-demo-bucket 的存储桶。us-east-1 之外的区域需要指定相应的 LocationConstraint 才能在所需区域创建存储桶。

```
aws s3api create-bucket \
    --bucket amzn-s3-demo-bucket \
    --region eu-west-1 \
    --create-bucket-configuration LocationConstraint=eu-west-1
```

输出：

```
{
    "Location": "http://amzn-s3-demo-bucket.s3.amazonaws.com/"
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的创建存储桶。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBucket。

create-multipart-upload

以下代码示例演示了如何使用 create-multipart-upload。

Amazon CLI

以下命令在存储桶 amzn-s3-demo-bucket 中创建带有密钥 multipart/01 的分段上传：

```
aws s3api create-multipart-upload --bucket amzn-s3-demo-bucket --key 'multipart/01'
```

输出：

```
{
    "Bucket": "amzn-s3-demo-bucket",
    "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3URC
    "Key": "multipart/01"
}
```

完成的文件在存储桶 amzn-s3-demo-bucket 中名为 multipart 文件夹中将命名为 01。保存上传 ID、密钥和存储桶名称以供 upload-part 命令使用。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMultipartUpload。

delete-bucket-analytics-configuration

以下代码示例演示了如何使用 delete-bucket-analytics-configuration。

Amazon CLI

删除存储桶的分析配置

以下 delete-bucket-analytics-configuration 示例移除指定存储桶和 ID 的分析配置。

```
aws s3api delete-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketAnalyticsConfiguration。

delete-bucket-cors

以下代码示例演示了如何使用 delete-bucket-cors。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除跨源资源共享配置：

```
aws s3api delete-bucket-cors --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketCors。

delete-bucket-encryption

以下代码示例演示了如何使用 delete-bucket-encryption。

Amazon CLI

删除存储桶的服务器端加密配置

以下 delete-bucket-encryption 示例删除指定存储桶的服务器端加密配置。

```
aws s3api delete-bucket-encryption \
    --bucket amzn-s3-demo-bucket
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketEncryption。

delete-bucket-intelligent-tiering-configuration

以下代码示例演示了如何使用 delete-bucket-intelligent-tiering-configuration。

Amazon CLI

移除存储桶上的 S3 智能分层配置

以下 delete-bucket-intelligent-tiering-configuration 示例移除存储桶上名为 ExampleConfig 的 S3 智能分层配置。

```
aws s3api delete-bucket-intelligent-tiering-configuration \
    --bucket amzn-s3-demo-bucket \
    --id ExampleConfig
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon S3 用户指南》中的使用 S3 智能分层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketIntelligentTieringConfiguration。

delete-bucket-inventory-configuration

以下代码示例演示了如何使用 delete-bucket-inventory-configuration。

Amazon CLI

删除存储桶的清单配置

以下 delete-bucket-inventory-configuration 示例删除指定存储桶的 ID 为 1 的清单配置。

```
aws s3api delete-bucket-inventory-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketInventoryConfiguration。

delete-bucket-lifecycle

以下代码示例演示了如何使用 delete-bucket-lifecycle。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除生命周期配置：

```
aws s3api delete-bucket-lifecycle --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketLifecycle。

delete-bucket-metrics-configuration

以下代码示例演示了如何使用 delete-bucket-metrics-configuration。

Amazon CLI

删除存储桶的指标配置

以下 delete-bucket-metrics-configuration 示例移除指定存储桶和 ID 的指标配置。

```
aws s3api delete-bucket-metrics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 123
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketMetricsConfiguration。

delete-bucket-ownership-controls

以下代码示例演示了如何使用 delete-bucket-ownership-controls。

Amazon CLI

移除存储桶的存储桶所有权设置

以下 delete-bucket-ownership-controls 示例移除存储桶的存储桶所有权设置。

```
aws s3api delete-bucket-ownership-controls \
    --bucket amzn-s3-demo-bucket
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon S3 用户指南》中的设置现有存储桶的对象所有权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketOwnershipControls。

delete-bucket-policy

以下代码示例演示了如何使用 delete-bucket-policy。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除存储桶策略：

```
aws s3api delete-bucket-policy --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketPolicy。

delete-bucket-replication

以下代码示例演示了如何使用 delete-bucket-replication。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除复制配置：

```
aws s3api delete-bucket-replication --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketReplication。

delete-bucket-tagging

以下代码示例演示了如何使用 delete-bucket-tagging。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除标记配置：

```
aws s3api delete-bucket-tagging --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketTagging。

delete-bucket-website

以下代码示例演示了如何使用 delete-bucket-website。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除网站配置：

```
aws s3api delete-bucket-website --bucket amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucketWebsite。

delete-bucket

以下代码示例演示了如何使用 delete-bucket。

Amazon CLI

以下命令删除名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api delete-bucket --bucket amzn-s3-demo-bucket --region us-east-1
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucket。

delete-object-tagging

以下代码示例演示了如何使用 delete-object-tagging。

Amazon CLI

删除对象的标签集

以下 delete-object-tagging 示例从对象 doc1.rtf 中删除带有指定键的标签。

```
aws s3api delete-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObjectTagging。

delete-object

以下代码示例演示了如何使用 delete-object。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除名为 test.txt 的对象：

```
aws s3api delete-object --bucket amzn-s3-demo-bucket --key test.txt
```

如果启用了存储桶版本控制，则输出将包含删除标记的版本 ID：

```
{
  "VersionId": "9_gKg5vG56F.TTEUdwkxGpJ3tNDlWlGq",
  "DeleteMarker": true
}
```

有关删除对象的更多信息，请参阅《Amazon S3 开发人员指南》中的“删除对象”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObject。

delete-objects

以下代码示例演示了如何使用 delete-objects。

Amazon CLI

以下命令从名为 amzn-s3-demo-bucket 的存储桶中删除对象：

```
aws s3api delete-objects --bucket amzn-s3-demo-bucket --delete file://delete.json
```

delete.json 是当前目录中指定要删除的对象的 JSON 文档：

```
{
  "Objects": [
    {
      "Key": "test1.txt"
    }
  ],
  "Quiet": false
}
```

输出：

```
{
    "Deleted": [
        {
            "DeleteMarkerVersionId": "mYAT5Mc6F7aeUL8SS7FAAqUPO1koHwzU",
            "Key": "test1.txt",
            "DeleteMarker": true
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObjects。

delete-public-access-block

以下代码示例演示了如何使用 delete-public-access-block。

Amazon CLI

删除存储桶的屏蔽公共访问权限配置

以下 delete-public-access-block 示例移除指定存储桶上的屏蔽公共访问权限配置。

```
aws s3api delete-public-access-block \
    --bucket amzn-s3-demo-bucket
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePublicAccessBlock。

get-bucket-accelerate-configuration

以下代码示例演示了如何使用 get-bucket-accelerate-configuration。

Amazon CLI

检索存储桶的加速配置

以下 get-bucket-accelerate-configuration 示例检索指定存储桶的加速配置。

```
aws s3api get-bucket-accelerate-configuration \
```

```
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Status": "Enabled"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketAccelerateConfiguration。

get-bucket-acl

以下代码示例演示了如何使用 get-bucket-acl。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的访问控制列表：

```
aws s3api get-bucket-acl --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketAcl。

get-bucket-analytics-configuration

以下代码示例演示了如何使用 get-bucket-analytics-configuration。

Amazon CLI

检索具有特定 ID 的存储桶的分析配置

以下 get-bucket-analytics-configuration 示例显示了指定存储桶和 ID 的分析配置。

```
aws s3api get-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

输出：

```
{
    "AnalyticsConfiguration": {
        "StorageClassAnalysis": {},
        "Id": "1"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketAnalyticsConfiguration。

get-bucket-cors

以下代码示例演示了如何使用 get-bucket-cors。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的跨源资源共享配置：

```
aws s3api get-bucket-cors --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "CORSRules": [
        {
            "AllowedHeaders": [
                "*"
```

```
            ],
            "ExposeHeaders": [
                "x-amz-server-side-encryption"
            ],
            "AllowedMethods": [
                "PUT",
                "POST",
                "DELETE"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedOrigins": [
                "http://www.example.com"
            ]
        },
        {
            "AllowedHeaders": [
                "Authorization"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedMethods": [
                "GET"
            ],
            "AllowedOrigins": [
                "*"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketCors。

get-bucket-encryption

以下代码示例演示了如何使用 get-bucket-encryption。

Amazon CLI

检索存储桶的服务器端加密配置

以下 get-bucket-encryption 示例检索存储桶 amzn-s3-demo-bucket 的服务器端加密配置。

```
aws s3api get-bucket-encryption \
```

```
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "ServerSideEncryptionConfiguration": {
        "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "AES256"
                }
            }
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketEncryption。

get-bucket-intelligent-tiering-configuration

以下代码示例演示了如何使用 get-bucket-intelligent-tiering-configuration。

Amazon CLI

检索存储桶上的 S3 智能分层配置

以下 get-bucket-intelligent-tiering-configuration 示例检索存储桶上名为 ExampleConfig 的 S3 智能分层配置。

```
aws s3api get-bucket-intelligent-tiering-configuration \
    --bucket amzn-s3-demo-bucket \
    --id ExampleConfig
```

输出：

```
{
    "IntelligentTieringConfiguration": {
        "Id": "ExampleConfig2",
        "Filter": {
            "Prefix": "images"
        },
```

```
        "Status": "Enabled",
        "Tierings": [
            {
                "Days": 90,
                "AccessTier": "ARCHIVE_ACCESS"
            },
            {
                "Days": 180,
                "AccessTier": "DEEP_ARCHIVE_ACCESS"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的使用 S3 智能分层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketIntelligentTieringConfiguration。

get-bucket-inventory-configuration

以下代码示例演示了如何使用 get-bucket-inventory-configuration。

Amazon CLI

检索存储桶的清单配置

以下 get-bucket-inventory-configuration 示例检索 ID 为 1 的指定存储桶的清单配置。

```
aws s3api get-bucket-inventory-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1
```

输出：

```
{
    "InventoryConfiguration": {
        "IsEnabled": true,
        "Destination": {
            "S3BucketDestination": {
                "Format": "ORC",
                "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
                "AccountId": "123456789012"
            }
        },
        "IncludedObjectVersions": "Current",
        "Id": "1",
        "Schedule": {
            "Frequency": "Weekly"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketInventoryConfiguration。

get-bucket-lifecycle-configuration

以下代码示例演示了如何使用 get-bucket-lifecycle-configuration。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的生命周期配置：

```
aws s3api get-bucket-lifecycle-configuration --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
```

```
                {
                    "NoncurrentDays": 0,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketLifecycleConfiguration。

get-bucket-lifecycle

以下代码示例演示了如何使用 get-bucket-lifecycle。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的生命周期配置：

```
aws s3api get-bucket-lifecycle --bucket amzn-s3-demo-bucket
```

输出：

```
{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (objects in logs/2015/)",
      "Prefix": "logs/2015/",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    },
    {
      "Expiration": {
        "Date": "2016-01-01T00:00:00.000Z"
      },
      "ID": "Delete 2014 logs in 2016.",
      "Prefix": "logs/2014/",
      "Status": "Enabled"
```

```
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketLifecycle。

get-bucket-location

以下代码示例演示了如何使用 get-bucket-location。

Amazon CLI

如果存在约束条件，则以下命令会检索名为 amzn-s3-demo-bucket 的存储桶的位置约束：

```
aws s3api get-bucket-location --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "LocationConstraint": "us-west-2"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketLocation。

get-bucket-logging

以下代码示例演示了如何使用 get-bucket-logging。

Amazon CLI

检索存储桶的日志记录状态

以下 get-bucket-logging 示例检索指定存储桶的日志记录状态。

```
aws s3api get-bucket-logging \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "LoggingEnabled": {
```

```
        "TargetPrefix": "",
        "TargetBucket": "amzn-s3-demo-bucket-logs"
          }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketLogging。

get-bucket-metrics-configuration

以下代码示例演示了如何使用 get-bucket-metrics-configuration。

Amazon CLI

检索具有特定 ID 的存储桶的指标配置

以下 get-bucket-metrics-configuration 示例显示了指定存储桶和 ID 的指标配置。

```
aws s3api get-bucket-metrics-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 123
```

输出：

```
{
    "MetricsConfiguration": {
        "Filter": {
            "Prefix": "logs"
        },
        "Id": "123"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketMetricsConfiguration。

get-bucket-notification-configuration

以下代码示例演示了如何使用 get-bucket-notification-configuration。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的通知配置：

```
aws s3api get-bucket-notification-configuration --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "TopicConfigurations": [
        {
            "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWNl",
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",
            "Events": [
                "s3:ObjectCreated:*"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketNotificationConfiguration。

get-bucket-notification

以下代码示例演示了如何使用 get-bucket-notification。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的通知配置：

```
aws s3api get-bucket-notification --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "TopicConfiguration": {
        "Topic": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",
        "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWNl",
        "Event": "s3:ObjectCreated:*",
        "Events": [
            "s3:ObjectCreated:*"
        ]
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketNotification。

get-bucket-ownership-controls

以下代码示例演示了如何使用 get-bucket-ownership-controls。

Amazon CLI

检索存储桶的存储桶所有权设置

以下 get-bucket-ownership-controls 示例检索存储桶的存储桶所有权设置。

```
aws s3api get-bucket-ownership-controls \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "OwnershipControls": {
        "Rules": [
            {
                "ObjectOwnership": "BucketOwnerEnforced"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的查看 S3 存储桶的对象所有权设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketOwnershipControls。

get-bucket-policy-status

以下代码示例演示了如何使用 get-bucket-policy-status。

Amazon CLI

检索存储桶的策略状态，此状态指示存储桶是否为公有存储桶

以下 get-bucket-policy-status 示例检索存储桶 amzn-s3-demo-bucket 的策略状态。

```
aws s3api get-bucket-policy-status \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "PolicyStatus": {
        "IsPublic": false
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketPolicyStatus。

get-bucket-policy

以下代码示例演示了如何使用 get-bucket-policy。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的存储桶策略：

```
aws s3api get-bucket-policy --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Policy": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"\",\"Effect
\":\"Allow\",\"Principal\":\"*\",\"Action\":\"s3:GetObject\",\"Resource\":
\"arn:aws:s3:::amzn-s3-demo-bucket/*\"},{\"Sid\":\"\",\"Effect\":\"Deny\",
\"Principal\":\"*\",\"Action\":\"s3:GetObject\",\"Resource\":\"arn:aws:s3:::amzn-s3-
demo-bucket/secret/*\"}]}"
}
```

获取并放置存储桶策略 以下示例演示了如何下载 Amazon S3 存储桶策略，修改文件，然后使用 put-bucket-policy 来应用修改后的存储桶策略。要将存储桶策略下载到文件中，您可以运行：

```
aws s3api get-bucket-policy --bucket amzn-s3-demo-bucket --query Policy --output
 text > policy.json
```

然后，您可以根据需要修改 policy.json 文件。最后，您可以通过运行以下对象，将此修改后的策略应用回 S3 存储桶：

policy.json 文件（根据需要）。最后，您可以通过运行以下对象，将此修改后的策略应用回 S3 存储桶：

文件（根据需要）。最后，您可以通过运行以下对象，将此修改后的策略应用回 S3 存储桶：

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketPolicy。

get-bucket-replication

以下代码示例演示了如何使用 get-bucket-replication。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的复制配置：

```
aws s3api get-bucket-replication --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "ReplicationConfiguration": {
        "Rules": [
            {
                "Status": "Enabled",
                "Prefix": "",
                "Destination": {
                    "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket-backup",
                    "StorageClass": "STANDARD"
                },
                "ID": "ZmUwNzE4ZmQ4tMjVhOS00MTlkLOGI4NDkzZTIWJjNTUtYTA1"
            }
        ],
        "Role": "arn:aws:iam::123456789012:role/s3-replication-role"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketReplication。

get-bucket-request-payment

以下代码示例演示了如何使用 get-bucket-request-payment。

Amazon CLI

检索存储桶的请求付款配置

以下 get-bucket-request-payment 示例检索指定存储桶的申请方付款配置。

```
aws s3api get-bucket-request-payment \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Payer": "BucketOwner"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketRequestPayment。

get-bucket-tagging

以下代码示例演示了如何使用 get-bucket-tagging。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的标记配置：

```
aws s3api get-bucket-tagging --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "TagSet": [
        {
            "Value": "marketing",
            "Key": "organization"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketTagging。

get-bucket-versioning

以下代码示例演示了如何使用 get-bucket-versioning。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的版本控制配置：

```
aws s3api get-bucket-versioning --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Status": "Enabled"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketVersioning。

get-bucket-website

以下代码示例演示了如何使用 get-bucket-website。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶的静态网站配置：

```
aws s3api get-bucket-website --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetBucketWebsite。

get-object-acl

以下代码示例演示了如何使用 get-object-acl。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶中对象的访问控制列表：

```
aws s3api get-object-acl --bucket amzn-s3-demo-bucket --key index.html
```

输出：

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectAcl。

get-object-attributes

以下代码示例演示了如何使用 get-object-attributes。

Amazon CLI

从对象检索元数据而不返回对象本身

以下 get-object-attributes 示例从对象 doc1.rtf 检索元数据。

```
aws s3api get-object-attributes \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf \
    --object-attributes "StorageClass" "ETag" "ObjectSize"
```

输出：

```
{
    "LastModified": "2022-03-15T19:37:31+00:00",
    "VersionId": "IuCPjXTDzHNfldAuitVBIKJpF2p1fg4P",
    "ETag": "b662d79adeb7c8d787ea7eafb9ef6207",
    "StorageClass": "STANDARD",
    "ObjectSize": 405
}
```

有关更多信息，请参阅《Amazon S3 API Reference》中的 GetObjectAttributes。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectAttributes。

get-object-legal-hold

以下代码示例演示了如何使用 get-object-legal-hold。

Amazon CLI

检索对象的法定保留状态

以下 get-object-legal-hold 示例检索指定对象的法定保留状态。

```
aws s3api get-object-legal-hold \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf
```

输出：

```
{
    "LegalHold": {
        "Status": "ON"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectLegalHold。

get-object-lock-configuration

以下代码示例演示了如何使用 get-object-lock-configuration。

Amazon CLI

检索存储桶的对象锁定配置

以下 get-object-lock-configuration 示例检索指定存储桶的对象锁定配置。

```
aws s3api get-object-lock-configuration \
    --bucket amzn-s3-demo-bucket-with-object-lock
```

输出：

```
{
    "ObjectLockConfiguration": {
        "ObjectLockEnabled": "Enabled",
        "Rule": {
            "DefaultRetention": {
                "Mode": "COMPLIANCE",
                "Days": 50
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectLockConfiguration。

get-object-retention

以下代码示例演示了如何使用 get-object-retention。

Amazon CLI

检索对象的对象保留配置

以下 get-object-retention 示例检索指定对象的对象保留配置。

```
aws s3api get-object-retention \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf
```

输出：

```
{
    "Retention": {
        "Mode": "GOVERNANCE",
        "RetainUntilDate": "2025-01-01T00:00:00.000Z"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectRetention。

get-object-tagging

以下代码示例演示了如何使用 get-object-tagging。

Amazon CLI

检索附加到对象的标签

以下 get-object-tagging 示例从指定的对象中检索指定键的值。

```
aws s3api get-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf
```

输出：

```
{
    "TagSet": [
        {
            "Value": "confidential",
```

```
            "Key": "designation"
        }
    ]
}
```

以下 get-object-tagging 示例尝试检索没有标签的对象 doc2.rtf 的标签集。

```
aws s3api get-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc2.rtf
```

输出：

```
{
    "TagSet": []
}
```

以下 get-object-tagging 示例检索具有多个标签的对象 doc3.rtf 的标签集。

```
aws s3api get-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc3.rtf
```

输出：

```
{
    "TagSet": [
        {
            "Value": "confidential",
            "Key": "designation"
        },
        {
            "Value": "finance",
            "Key": "department"
        },
        {
            "Value": "payroll",
            "Key": "team"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectTagging。

get-object-torrent

以下代码示例演示了如何使用 get-object-torrent。

Amazon CLI

以下命令为名为 amzn-s3-demo-bucket 的存储桶中的对象创建种子文件：

```
aws s3api get-object-torrent --bucket amzn-s3-demo-bucket --key large-video-file.mp4 large-video-file.torrent
```

种子文件保存在本地的当前文件夹中。请注意，指定输出 filename (large-video-file.torrent) 时没有选项名称，并且必须是命令中的最后一个参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObjectTorrent。

get-object

以下代码示例演示了如何使用 get-object。

Amazon CLI

以下示例使用 get-object 命令从 Amazon S3 下载对象：

```
aws s3api get-object --bucket text-content --key dir/my_images.tar.bz2 my_images.tar.bz2
```

请注意，指定 outfile 参数时没有诸如“--outfile”之类的选项名称。输出文件的名称必须是命令中的最后一个参数。

以下示例演示了如何使用 --range 从对象下载特定字节范围。请注意，字节范围需要以“bytes=”为前缀：

```
aws s3api get-object --bucket text-content --key dir/my_data --range bytes=8888-9999 my_data_range
```

有关检索对象的更多信息，请参阅《Amazon S3 开发人员指南》中的“获取对象”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObject。

get-public-access-block

以下代码示例演示了如何使用 get-public-access-block。

Amazon CLI

设置或修改存储桶的屏蔽公共访问权限配置

以下 get-public-access-block 示例显示了指定存储桶的屏蔽公共访问权限配置。

```
aws s3api get-public-access-block \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "PublicAccessBlockConfiguration": {
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "BlockPublicAcls": true,
        "RestrictPublicBuckets": true
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicAccessBlock。

head-bucket

以下代码示例演示了如何使用 head-bucket。

Amazon CLI

以下命令验证对名为 amzn-s3-demo-bucket 的存储桶的访问权限：

```
aws s3api head-bucket --bucket amzn-s3-demo-bucket
```

如果存储桶存在并且您可以访问它，则不返回任何输出。否则，会显示错误消息。例如：

```
A client error (404) occurred when calling the HeadBucket operation: Not Found
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 HeadBucket。

`head-object`

以下代码示例演示了如何使用 `head-object`。

Amazon CLI

以下命令检索名为 `amzn-s3-demo-bucket` 的存储桶中对象的元数据：

```
aws s3api head-object --bucket amzn-s3-demo-bucket --key index.html
```

输出：

```
{
    "AcceptRanges": "bytes",
    "ContentType": "text/html",
    "LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
    "ContentLength": 77,
    "VersionId": "null",
    "ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
    "Metadata": {}
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 HeadObject。

`list-bucket-analytics-configurations`

以下代码示例演示了如何使用 `list-bucket-analytics-configurations`。

Amazon CLI

检索存储桶的分析配置列表

下面的 `list-bucket-analytics-configurations` 检索指定存储桶的分析配置列表。

```
aws s3api list-bucket-analytics-configurations \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "AnalyticsConfigurationList": [
        {
            "StorageClassAnalysis": {},
            "Id": "1"
        }
    ],
    "IsTruncated": false
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketAnalyticsConfigurations。

list-bucket-intelligent-tiering-configurations

以下代码示例演示了如何使用 list-bucket-intelligent-tiering-configurations。

Amazon CLI

检索存储桶上的所有 S3 智能分层配置

以下 list-bucket-intelligent-tiering-configurations 示例检索存储桶上的所有 S3 智能分层配置。

```
aws s3api list-bucket-intelligent-tiering-configurations \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "IsTruncated": false,
    "IntelligentTieringConfigurationList": [
        {
            "Id": "ExampleConfig",
            "Filter": {
                "Prefix": "images"
            },
            "Status": "Enabled",
            "Tierings": [
                {
                    "Days": 90,
                    "AccessTier": "ARCHIVE_ACCESS"
```

```
                },
                {
                    "Days": 180,
                    "AccessTier": "DEEP_ARCHIVE_ACCESS"
                }
            ]
        },
        {
            "Id": "ExampleConfig2",
            "Status": "Disabled",
            "Tierings": [
                {
                    "Days": 730,
                    "AccessTier": "ARCHIVE_ACCESS"
                }
            ]
        },
        {
            "Id": "ExampleConfig3",
            "Filter": {
                "Tag": {
                    "Key": "documents",
                    "Value": "taxes"
                }
            },
            "Status": "Enabled",
            "Tierings": [
                {
                    "Days": 90,
                    "AccessTier": "ARCHIVE_ACCESS"
                },
                {
                    "Days": 365,
                    "AccessTier": "DEEP_ARCHIVE_ACCESS"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon S3 用户指南》中的使用 S3 智能分层。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketIntelligentTieringConfigurations。

list-bucket-inventory-configurations

以下代码示例演示了如何使用 list-bucket-inventory-configurations。

Amazon CLI

检索存储桶的清单配置列表

以下 list-bucket-inventory-configurations 示例列出了指定存储桶的清单配置。

```
aws s3api list-bucket-inventory-configurations \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "InventoryConfigurationList": [
        {
            "IsEnabled": true,
            "Destination": {
                "S3BucketDestination": {
                    "Format": "ORC",
                    "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
                    "AccountId": "123456789012"
                }
            },
            "IncludedObjectVersions": "Current",
            "Id": "1",
            "Schedule": {
                "Frequency": "Weekly"
            }
        },
        {
            "IsEnabled": true,
            "Destination": {
                "S3BucketDestination": {
                    "Format": "CSV",
                    "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
                    "AccountId": "123456789012"
                }
            },
            "IncludedObjectVersions": "Current",
            "Id": "2",
```

```
            "Schedule": {
                "Frequency": "Daily"
            }
        }
    ],
    "IsTruncated": false
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketInventoryConfigurations。

list-bucket-metrics-configurations

以下代码示例演示了如何使用 list-bucket-metrics-configurations。

Amazon CLI

检索存储桶的指标配置列表

以下 list-bucket-metrics-configurations 示例检索指定存储桶的指标配置列表。

```
aws s3api list-bucket-metrics-configurations \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "IsTruncated": false,
    "MetricsConfigurationList": [
        {
            "Filter": {
                "Prefix": "logs"
            },
            "Id": "123"
        },
        {
            "Filter": {
                "Prefix": "tmp"
            },
            "Id": "234"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBucketMetricsConfigurations。

list-buckets

以下代码示例演示了如何使用 list-buckets。

Amazon CLI

以下命令使用 list-buckets 命令显示所有 Amazon S3 存储桶的名称（跨所有区域）：

```
aws s3api list-buckets --query "Buckets[].Name"
```

查询选项会筛选 list-buckets 的输出，使其范围缩小到仅限存储桶名称。

有关存储桶的更多信息，请参阅《Amazon S3 开发人员指南》中的“使用 Amazon S3 存储桶”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListBuckets。

list-multipart-uploads

以下代码示例演示了如何使用 list-multipart-uploads。

Amazon CLI

以下命令列出名为 amzn-s3-demo-bucket 的存储桶的所有活动分段上传：

```
aws s3api list-multipart-uploads --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Uploads": [
        {
            "Initiator": {
                "DisplayName": "username",
                "ID": "arn:aws:iam::0123456789012:user/username"
            },
            "Initiated": "2015-06-02T18:01:30.000Z",
            "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3URC
            "StorageClass": "STANDARD",
            "Key": "multipart/01",
```

```
            "Owner": {
                "DisplayName": "aws-account-name",
                "ID":
 "100719349fc3b6dcd7c820a124bf7aecd408092c3d7b51b38494939801fc248b"
            }
        }
    ],
    "CommonPrefixes": []
}
```

正在进行的分段上传会产生 Amazon S3 存储费用。完成或中止活动分段上传，可将其从您的账户中移除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMultipartUploads。

`list-object-versions`

以下代码示例演示了如何使用 `list-object-versions`。

Amazon CLI

以下命令检索名为 amzn-s3-demo-bucket 的存储桶中对象的版本信息：

```
aws s3api list-object-versions --bucket amzn-s3-demo-bucket --prefix index.html
```

输出：

```
{
    "DeleteMarkers": [
        {
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": true,
            "VersionId": "B2VsEK5saUNNHKcOAJj7hIE86RozToyq",
            "Key": "index.html",
            "LastModified": "2015-11-10T00:57:03.000Z"
        },
        {
            "Owner": {
                "DisplayName": "my-username",
```

```
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "VersionId": ".FLQEZscLIcfxSq.jsFJ.szUkmng2Yw6",
            "Key": "index.html",
            "LastModified": "2015-11-09T23:32:20.000Z"
        }
    ],
    "Versions": [
        {
            "LastModified": "2015-11-10T00:20:11.000Z",
            "VersionId": "Rb_l2T8UHDkFEwCgJjhlgPOZC0qJ.vpD",
            "ETag": "\"0622528de826c0df5db1258a23b80be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
        {
            "LastModified": "2015-11-09T23:26:41.000Z",
            "VersionId": "rasWWGpgk9E4s0LyTJgusGeRQKLVIAFf",
            "ETag": "\"06225825b8028de826c0df5db1a23be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
        {
            "LastModified": "2015-11-09T22:50:50.000Z",
            "VersionId": "null",
            "ETag": "\"d1f45267a863c8392e07d24dd592f1b9\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
```

```
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 533823
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListObjectVersions。

list-objects-v2

以下代码示例演示了如何使用 list-objects-v2。

Amazon CLI

获取存储桶中对象的列表

以下 list-objects-v2 示例列出了指定存储桶中的对象。

```
aws s3api list-objects-v2 \
    --bucket amzn-s3-demo-bucket
```

输出：

```
{
    "Contents": [
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"621503c373607d548b37cff8778d992c\"",
            "StorageClass": "STANDARD",
            "Key": "doc1.rtf",
            "Size": 391
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"a2cecc36ab7c7fe3a71a273b9d45b1b5\"",
            "StorageClass": "STANDARD",
            "Key": "doc2.rtf",
```

```
            "Size": 373
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"08210852f65a2e9cb999972539a64d68\"",
            "StorageClass": "STANDARD",
            "Key": "doc3.rtf",
            "Size": 399
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"d1852dd683f404306569471af106988e\"",
            "StorageClass": "STANDARD",
            "Key": "doc4.rtf",
            "Size": 6225
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListObjectsV2。

list-objects

以下代码示例演示了如何使用 list-objects。

Amazon CLI

以下示例使用 list-objects 命令显示指定存储桶中所有对象的名称：

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size:
 Size}'
```

该示例使用 --query 参数筛选 list-objects 的输出，使其范围缩小到每个对象的键值和大小

有关对象的更多信息，请参阅《Amazon S3 开发人员指南》中的“使用 Amazon S3 对象”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListObjects。

list-parts

以下代码示例演示了如何使用 list-parts。

Amazon CLI

以下命令列出存储桶 amzn-s3-demo-bucket 中，使用密钥 multipart/01 的分段上传中已上传的所有部分：

```
aws s3api list-parts --bucket amzn-s3-demo-bucket --key 'multipart/01' --upload-
id dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3UR
```

输出：

```
{
    "Owner": {
        "DisplayName": "aws-account-name",
        "ID": "100719349fc3b6dcd7c820a124bf7aecd408092c3d7b51b38494939801fc248b"
    },
    "Initiator": {
        "DisplayName": "username",
        "ID": "arn:aws:iam::0123456789012:user/username"
    },
    "Parts": [
        {
            "LastModified": "2015-06-02T18:07:35.000Z",
            "PartNumber": 1,
            "ETag": "\"e868e0f4719e394144ef36531ee6824c\"",
            "Size": 5242880
        },
        {
            "LastModified": "2015-06-02T18:07:42.000Z",
            "PartNumber": 2,
            "ETag": "\"6bb2b12753d66fe86da4998aa33fffb0\"",
            "Size": 5242880
        },
        {
            "LastModified": "2015-06-02T18:07:47.000Z",
            "PartNumber": 3,
            "ETag": "\"d0a0112e841abec9c9ec83406f0159c8\"",
            "Size": 5242880
        }
    ],
    "StorageClass": "STANDARD"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListParts。

ls

以下代码示例演示了如何使用 ls。

Amazon CLI

示例 1：列出用户拥有的所有存储桶

以下 ls 命令列出用户拥有的所有存储桶。在此示例中，用户拥有存储桶 amzn-s3-demo-bucket 和 amzn-s3-demo-bucket2。时间戳是存储桶的创建日期，以计算机的时区显示。对存储桶进行更改（例如编辑存储桶策略）时，此日期可能会更改。请注意，如果路径参数 <S3Uri> 使用了 s3://，则也会列出所有存储桶。

```
aws s3 ls
```

输出：

```
2013-07-11 17:08:50 amzn-s3-demo-bucket
2013-07-24 14:55:44 amzn-s3-demo-bucket2
```

示例 2：列出存储桶中的所有前缀和对象

以下 ls 命令列出指定存储桶和前缀下的对象和常用前缀。在此示例中，用户拥有带有对象 test.txt 和 somePrefix/test.txt 的存储桶 amzn-s3-demo-bucket。LastWriteTime 和 Length 是任意值。请注意，由于 ls 命令与本地文件系统没有交互，因此不需要使用 s3:// URI 方案来解决歧义，可以省略。

```
aws s3 ls s3://amzn-s3-demo-bucket
```

输出：

```
                           PRE somePrefix/
2013-07-25 17:06:27         88 test.txt
```

示例 3：列出特定存储桶和前缀中的所有前缀和对象

以下 ls 命令列出指定存储桶和前缀下的对象和常用前缀。但是，指定的存储桶和前缀下没有对象，也没有常用前缀。

```
aws s3 ls s3://amzn-s3-demo-bucket/noExistPrefix
```

输出：

```
None
```

示例 4：递归列出存储桶中的所有前缀和对象

以下 ls 命令将递归列出存储桶中的对象。输出中将不再显示 PRE dirname/，而是按顺序列出存储桶中的所有内容。

```
aws s3 ls s3://amzn-s3-demo-bucket \
    --recursive
```

输出：

```
2013-09-02 21:37:53         10 a.txt
2013-09-02 21:37:53    2863288 foo.zip
2013-09-02 21:32:57         23 foo/bar/.baz/a
2013-09-02 21:32:58         41 foo/bar/.baz/b
2013-09-02 21:32:57        281 foo/bar/.baz/c
2013-09-02 21:32:57         73 foo/bar/.baz/d
2013-09-02 21:32:57        452 foo/bar/.baz/e
2013-09-02 21:32:57        896 foo/bar/.baz/hooks/bar
2013-09-02 21:32:57        189 foo/bar/.baz/hooks/foo
2013-09-02 21:32:57        398 z.txt
```

示例 5：汇总存储桶中的所有前缀和对象

以下 ls 命令使用 --human-readable 和 --summarize 选项演示了相同的命令。--human readable 会以 Bytes/MiB/KiB/GiB/TiB/PiB/EiB 为单位显示文件大小。--summarize 会在结果列表的末尾显示对象的总数和总大小：

```
aws s3 ls s3://amzn-s3-demo-bucket \
    --recursive \
    --human-readable \
    --summarize
```

输出：

```
2013-09-02 21:37:53   10 Bytes a.txt
2013-09-02 21:37:53  2.9 MiB foo.zip
2013-09-02 21:32:57   23 Bytes foo/bar/.baz/a
```

```
2013-09-02 21:32:58   41 Bytes foo/bar/.baz/b
2013-09-02 21:32:57  281 Bytes foo/bar/.baz/c
2013-09-02 21:32:57   73 Bytes foo/bar/.baz/d
2013-09-02 21:32:57  452 Bytes foo/bar/.baz/e
2013-09-02 21:32:57  896 Bytes foo/bar/.baz/hooks/bar
2013-09-02 21:32:57  189 Bytes foo/bar/.baz/hooks/foo
2013-09-02 21:32:57  398 Bytes z.txt

Total Objects: 10
   Total Size: 2.9 MiB
```

示例 6：从 S3 接入点列出

以下 `ls` 命令列出来自接入点（`myaccesspoint`）的对象：

```
aws s3 ls s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
```

输出：

```
                           PRE somePrefix/
2013-07-25 17:06:27         88 test.txt
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Ls。

mb

以下代码示例演示了如何使用 mb。

Amazon CLI

示例 1：创建存储桶

以下 mb 命令创建存储桶。在此示例中，用户创建了存储桶 amzn-s3-demo-bucket。存储桶可在用户配置文件中指定的区域内创建：

```
aws s3 mb s3://amzn-s3-demo-bucket
```

输出：

```
make_bucket: s3://amzn-s3-demo-bucket
```

示例 2：在指定区域内创建存储桶

以下 mb 命令在 --region 参数指定的区域内创建一个存储桶。在此示例中，用户在区域 us-west-1 内创建了存储桶 amzn-s3-demo-bucket：

```
aws s3 mb s3://amzn-s3-demo-bucket \
    --region us-west-1
```

输出：

```
make_bucket: s3://amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Mb。

mv

以下代码示例演示了如何使用 mv。

Amazon CLI

示例 1：将本地文件移动到指定的存储桶

以下 mv 命令将单个文件移动到指定的存储桶和密钥。

```
aws s3 mv test.txt s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
move: test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 2：将对象移动到指定的存储桶和密钥

以下 mv 命令将单个 S3 对象移动到指定的存储桶和密钥。

```
aws s3 mv s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
move: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 3：将 S3 对象移动到本地目录

以下 mv 命令将单个对象移动到指定的本地文件。

```
aws s3 mv s3://amzn-s3-demo-bucket/test.txt test2.txt
```

输出：

```
move: s3://amzn-s3-demo-bucket/test.txt to test2.txt
```

示例 4：将具有原始名称的对象移动到指定的存储桶

以下 mv 命令将单个对象移动到指定的存储桶，同时保留其原始名称：

```
aws s3 mv s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket2/
```

输出：

```
move: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket2/test.txt
```

示例 5：将存储桶中的所有对象和前缀移动到本地目录

当与参数 --recursive 一起传递时，以下 mv 命令将指定前缀和存储桶下的所有对象递归移动到指定目录。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 test2.txt。

```
aws s3 mv s3://amzn-s3-demo-bucket . \
    --recursive
```

输出：

```
move: s3://amzn-s3-demo-bucket/test1.txt to test1.txt
move: s3://amzn-s3-demo-bucket/test2.txt to test2.txt
```

示例 6：将存储桶中的所有对象和前缀移动到本地目录（“.jpg”文件除外）

当与参数 --recursive 一起传递时，以下 mv 命令将指定目录下的所有文件递归移动到指定的存储桶和前缀，同时使用 --exclude 参数将某些文件排除在外。在此示例中，目录 myDir 中包含文件 test1.txt 和 test2.jpg。

```
aws s3 mv myDir s3://amzn-s3-demo-bucket/ \
    --recursive \
    --exclude "*.jpg"
```

输出：

```
move: myDir/test1.txt to s3://amzn-s3-demo-bucket2/test1.txt
```

示例 7：将存储桶中的所有对象和前缀移动到本地目录（指定前缀除外）

当与参数 --recursive 一起传递时，以下 mv 命令将指定存储桶下的所有对象递归移动到另一个存储桶，同时使用 --exclude 参数将某些对象排除在外。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 another/test1.txt。

```
aws s3 mv s3://amzn-s3-demo-bucket/ s3://amzn-s3-demo-bucket2/ \
    --recursive \
    --exclude "amzn-s3-demo-bucket/another/*"
```

输出：

```
move: s3://amzn-s3-demo-bucket/test1.txt to s3://amzn-s3-demo-bucket2/test1.txt
```

示例 8：将对象移动到指定的存储桶并设置 ACL

以下 mv 命令将单个对象移动到指定的存储桶和密钥，同时将 ACL 设置为 public-read-write。

```
aws s3 mv s3://amzn-s3-demo-bucket/test.txt s3://amzn-s3-demo-bucket/test2.txt \
    --acl public-read-write
```

输出：

```
move: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 9：将本地文件移动到指定的存储桶并授予权限

以下 mv 命令说明如何使用 --grants 选项向所有用户授予读取权限，并向通过电子邮件地址识别的特定用户授予完全控制权限。

```
aws s3 mv file.txt s3://amzn-s3-demo-bucket/ \
```

```
    --grants read=uri=http://acs.amazonaws.com/groups/global/
AllUsers full=emailaddress=user@example.com
```

输出：

```
move: file.txt to s3://amzn-s3-demo-bucket/file.txt
```

示例 10：将文件移动到 S3 接入点

以下 mv 命令将名为 mydoc.txt 的单个文件移动到名为 mykey 的密钥下的名为 myaccesspoint 的接入点。

```
aws s3 mv mydoc.txt s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/mykey
```

输出：

```
move: mydoc.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
mykey
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Mv。

presign

以下代码示例演示了如何使用 presign。

Amazon CLI

示例 1：创建默认生命周期为一小时的预签名 URL，链接到 S3 存储桶中的对象

以下 presign 命令为指定的存储桶和密钥生成一个预签名 URL，有效期为一小时。

```
aws s3 presign s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/key?X-Amz-Algorithm=AWS4-
HMAC-SHA256&X-Amz-Credential=AKIAEXAMPLE123456789%2F20210621%2Fus-
west-2%2Fs3%2Faws4_request&X-Amz-Date=20210621T041609Z&X-Amz-Expires=3600&X-Amz-
SignedHeaders=host&X-Amz-
Signature=EXAMBLE1234494d5fba3fed607f98018e1dfc62e2529ae96d844123456
```

示例 2：创建自定义生命周期的预签名 URL，链接到 S3 存储桶中的对象

以下 presign 命令为指定的存储桶和密钥生成一个预签名 URL，有效期为一周。

```
aws s3 presign s3://amzn-s3-demo-bucket/test2.txt \
    --expires-in 604800
```

输出：

```
https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/key?X-Amz-Algorithm=AWS4-
HMAC-SHA256&X-Amz-Credential=AKIAEXAMPLE123456789%2F20210621%2Fus-
west-2%2Fs3%2Faws4_request&X-Amz-Date=20210621T041609Z&X-Amz-Expires=604800&X-Amz-
SignedHeaders=host&X-Amz-
Signature=EXAMBLE1234494d5fba3fed607f98018e1dfc62e2529ae96d844123456
```

有关更多信息，请参阅《S3 开发人员指南》中的与其他用户共享对象。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Presign。

put-bucket-accelerate-configuration

以下代码示例演示了如何使用 put-bucket-accelerate-configuration。

Amazon CLI

设置存储桶的加速配置

以下 put-bucket-accelerate-configuration 示例启用指定存储桶的加速配置。

```
aws s3api put-bucket-accelerate-configuration \
    --bucket amzn-s3-demo-bucket \
    --accelerate-configuration Status=Enabled
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketAccelerateConfiguration。

put-bucket-acl

以下代码示例演示了如何使用 put-bucket-acl。

Amazon CLI

此示例向两个 Amazon 用户（user1@example.com 和 user2@example.com）授予 `full control` 权限，并向所有人授予 `read` 权限：

```
aws s3api put-bucket-acl --bucket amzn-s3-demo-bucket --grant-full-
control emailaddress=user1@example.com,emailaddress=user2@example.com --grant-
read uri=http://acs.amazonaws.com/groups/global/AllUsers
```

有关自定义 ACL（s3api ACL 命令（如 `put-bucket-acl`）使用相同的速记参数表示法）的详细信息，请参阅 http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketAcl。

put-bucket-analytics-configuration

以下代码示例演示了如何使用 `put-bucket-analytics-configuration`。

Amazon CLI

设置存储桶的分析配置

以下 `put-bucket-analytics-configuration` 示例启用指定存储桶的分析配置。

```
aws s3api put-bucket-analytics-configuration \
    --bucket amzn-s3-demo-bucket --id 1 \
    --analytics-configuration '{"Id": "1","StorageClassAnalysis": {}}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketAnalyticsConfiguration。

put-bucket-cors

以下代码示例演示了如何使用 `put-bucket-cors`。

Amazon CLI

以下示例启用来自 www.example.com 的 PUT、POST 和 DELETE 请求，并启用来自任何域的 GET 请求：

```
aws s3api put-bucket-cors --bucket amzn-s3-demo-bucket --cors-configuration file://
cors.json

cors.json:
{
  "CORSRules": [
    {
      "AllowedOrigins": ["http://www.example.com"],
      "AllowedHeaders": ["*"],
      "AllowedMethods": ["PUT", "POST", "DELETE"],
      "MaxAgeSeconds": 3000,
      "ExposeHeaders": ["x-amz-server-side-encryption"]
    },
    {
      "AllowedOrigins": ["*"],
      "AllowedHeaders": ["Authorization"],
      "AllowedMethods": ["GET"],
      "MaxAgeSeconds": 3000
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketCors。

put-bucket-encryption

以下代码示例演示了如何使用 put-bucket-encryption。

Amazon CLI

配置存储桶的服务器端加密

以下 put-bucket-encryption 示例将 AES256 加密设置为指定存储桶的默认值。

```
aws s3api put-bucket-encryption \
    --bucket amzn-s3-demo-bucket \
    --server-side-encryption-configuration '{"Rules":
 [{"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}]}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketEncryption。

put-bucket-intelligent-tiering-configuration

以下代码示例演示了如何使用 put-bucket-intelligent-tiering-configuration。

Amazon CLI

更新存储桶上的 S3 智能分层配置

以下 put-bucket-intelligent-tiering-configuration 示例更新存储桶上名为 ExampleConfig 的 S3 智能分层配置。该配置会在 90 天后将前缀映像下未曾访问过的对象过渡到归档访问层，并在 180 天后过渡到深度归档访问层。

```
aws s3api put-bucket-intelligent-tiering-configuration \
    --bucket amzn-s3-demo-bucket \
    --id "ExampleConfig" \
    --intelligent-tiering-configuration file://intelligent-tiering-
configuration.json
```

的内容intelligent-tiering-configuration.json：

```
{
    "Id": "ExampleConfig",
    "Status": "Enabled",
    "Filter": {
        "Prefix": "images"
        },
    "Tierings": [
        {
            "Days": 90,
            "AccessTier": "ARCHIVE_ACCESS"
        },
        {
            "Days": 180,
            "AccessTier": "DEEP_ARCHIVE_ACCESS"
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon S3 用户指南》中的设置现有存储桶的对象所有权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketIntelligentTieringConfiguration。

put-bucket-inventory-configuration

以下代码示例演示了如何使用 put-bucket-inventory-configuration。

Amazon CLI

示例 1：为存储桶设置清单配置

以下 put-bucket-inventory-configuration 示例为存储桶 amzn-s3-demo-bucket 设置 ORC 格式的每周清单报告。

```
aws s3api put-bucket-inventory-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 1 \
    --inventory-configuration '{"Destination": { "S3BucketDestination":
 { "AccountId": "123456789012", "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
 "Format": "ORC" }}, "IsEnabled": true, "Id": "1", "IncludedObjectVersions":
 "Current", "Schedule": { "Frequency": "Weekly" }}'
```

此命令不生成任何输出。

示例 2：为存储桶设置清单配置

以下 put-bucket-inventory-configuration 示例为存储桶 amzn-s3-demo-bucket 设置 CSV 格式的每日清单报告。

```
aws s3api put-bucket-inventory-configuration \
    --bucket amzn-s3-demo-bucket \
    --id 2 \
    --inventory-configuration '{"Destination": { "S3BucketDestination":
 { "AccountId": "123456789012", "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket",
 "Format": "CSV" }}, "IsEnabled": true, "Id": "2", "IncludedObjectVersions":
 "Current", "Schedule": { "Frequency": "Daily" }}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketInventoryConfiguration。

put-bucket-lifecycle-configuration

以下代码示例演示了如何使用 put-bucket-lifecycle-configuration。

Amazon CLI

以下命令将生命周期配置应用于名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api put-bucket-lifecycle-configuration --bucket amzn-s3-demo-bucket --
lifecycle-configuration  file://lifecycle.json
```

文件 lifecycle.json 是当前文件夹中指定两个规则的 JSON 文档：

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
                {
                    "NoncurrentDays": 2,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

第一条规则在指定日期将带有前缀 rotated 的文件移动到 Glacier。第二条规则在旧对象版本不再是最新版本时将其移至 Glacier。有关可接受时间戳格式的信息，请参阅《Amazon CLI 用户指南》中的“指定参数值”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketLifecycleConfiguration。

put-bucket-lifecycle

以下代码示例演示了如何使用 put-bucket-lifecycle。

Amazon CLI

以下命令将生命周期配置应用于存储桶 amzn-s3-demo-bucket：

```
aws s3api put-bucket-lifecycle --bucket amzn-s3-demo-bucket --lifecycle-
configuration file://lifecycle.json
```

文件 lifecycle.json 是当前文件夹中指定两个规则的 JSON 文档：

```
{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (objects in logs/2015/)",
      "Prefix": "logs/2015/",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    },
    {
      "Expiration": {
        "Date": "2016-01-01T00:00:00.000Z"
      },
      "ID": "Delete 2014 logs in 2016.",
      "Prefix": "logs/2014/",
      "Status": "Enabled"
    }
  ]
}
```

第一条规则是在六十天后将文件移动到 Amazon Glacier。第二条规则是在指定日期从 Amazon S3 中删除文件。有关可接受时间戳格式的信息，请参阅《Amazon CLI 用户指南》中的“指定参数值”。

上面示例中的每条规则都指定了其适用的策略（Transition 或 Expiration）和文件前缀（文件夹名称）。您还可以通过指定空白前缀来创建适用于整个存储桶的规则：

```
{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (all objects in bucket)",
      "Prefix": "",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketLifecycle。

put-bucket-logging

以下代码示例演示了如何使用 put-bucket-logging。

Amazon CLI

示例 1：设置存储桶策略日志记录

以下 put-bucket-logging 示例为 amzn-s3-demo-bucket 设置日志记录策略。首先，使用 put-bucket-policy 命令在存储桶策略中向日志记录服务主体授予权限。

```
aws s3api put-bucket-policy \
    --bucket amzn-s3-demo-bucket \
    --policy file://policy.json
```

的内容policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {"Service": "logging.s3.amazonaws.com"},
            "Action": "s3:PutObject",
```

```
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/Logs/*",
            "Condition": {
                "ArnLike": {"aws:SourceARN": "arn:aws:s3:::SOURCE-BUCKET-NAME"},
                "StringEquals": {"aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"}
            }
        }
    ]
}
```

要应用日志记录策略，请使用 put-bucket-logging。

```
aws s3api put-bucket-logging \
    --bucket amzn-s3-demo-bucket \
    --bucket-logging-status file://logging.json
```

的内容logging.json：

```
{
     "LoggingEnabled": {
         "TargetBucket": "amzn-s3-demo-bucket",
         "TargetPrefix": "Logs/"
     }
 }
```

向日志记录服务主体授予 s3:PutObject 权限需要使用 put-bucket-policy 命令。

有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的 Amazon S3 服务器访问日志记录。

示例 2：设置仅向单个用户授予日志访问权限的存储桶策略

以下 put-bucket-logging 示例为 amzn-s3-demo-bucket 设置日志记录策略。Amazon 用户 bob@example.com 将对日志文件具有完全控制权限，其他人没有任何访问权限。首先，使用 put-bucket-acl 授予 S3 权限。

```
aws s3api put-bucket-acl \
    --bucket amzn-s3-demo-bucket \
    --grant-write URI=http://acs.amazonaws.com/groups/s3/LogDelivery \
    --grant-read-acp URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

然后，使用 put-bucket-logging 应用日志记录策略。

```
aws s3api put-bucket-logging \
    --bucket amzn-s3-demo-bucket \
    --bucket-logging-status file://logging.json
```

的内容logging.json：

```
{
    "LoggingEnabled": {
        "TargetBucket": "amzn-s3-demo-bucket",
        "TargetPrefix": "amzn-s3-demo-bucket-logs/",
        "TargetGrants": [
            {
                "Grantee": {
                    "Type": "AmazonCustomerByEmail",
                    "EmailAddress": "bob@example.com"
                },
                "Permission": "FULL_CONTROL"
            }
        ]
    }
}
```

必须使用 put-bucket-acl 命令向 S3 的日志传输系统授予必要的权限（write-acp 和 read-acp 权限）。

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的 Amazon S3 服务器访问日志记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketLogging。

put-bucket-metrics-configuration

以下代码示例演示了如何使用 put-bucket-metrics-configuration。

Amazon CLI

设置存储桶的指标配置

以下 put-bucket-metrics-configuration 示例为指定存储桶设置 ID 为 123 的指标配置。

```
aws s3api put-bucket-metrics-configuration \
```

```
    --bucket amzn-s3-demo-bucket \
    --id 123 \
    --metrics-configuration '{"Id": "123", "Filter": {"Prefix": "logs"}}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketMetricsConfiguration。

put-bucket-notification-configuration

以下代码示例演示了如何使用 put-bucket-notification-configuration。

Amazon CLI

启用存储桶的指定通知

以下 put-bucket-notification-configuration 示例将通知配置应用于名为 amzn-s3-demo-bucket 的存储桶。文件 notification.json 是当前文件夹中的一个 JSON 文件，用于指定 SNS 主题和要监控的事件类型。

```
aws s3api put-bucket-notification-configuration \
    --bucket amzn-s3-demo-bucket \
    --notification-configuration file://notification.json
```

的内容notification.json：

```
{
    "TopicConfigurations": [
        {
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic",
            "Events": [
                "s3:ObjectCreated:*"
            ]
        }
    ]
}
```

SNS 主题必须附加一个 IAM 策略，以允许 Amazon S3 向其发布。

```
{
    "Version":"2012-10-17",
```

```
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "arn:aws:sns:us-west-2:123456789012::s3-notification-topic",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:amzn-s3-demo-bucket"
                }
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketNotificationConfiguration。

put-bucket-notification

以下代码示例演示了如何使用 put-bucket-notification。

Amazon CLI

将通知配置应用于名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api put-bucket-notification --bucket amzn-s3-demo-bucket --notification-
configuration file://notification.json
```

文件 notification.json 是当前文件夹中的一个 JSON 文件，用于指定 SNS 主题和要监控的事件类型：

```
{
  "TopicConfiguration": {
    "Event": "s3:ObjectCreated:*",
    "Topic": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic"
```

```
  }
}
```

SNS 主题必须附加一个 IAM 策略，以允许 Amazon S3 向其发布：

```
{
 "Version":"2012-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
     "Service": "s3.amazonaws.com"
   },
   "Action": [
    "SNS:Publish"
   ],
   "Resource": "arn:aws:sns:us-west-2:123456789012:amzn-s3-demo-bucket",
   "Condition": {
      "ArnLike": {
      "aws:SourceArn": "arn:aws:s3:*:*:amzn-s3-demo-bucket"
    }
   }
  }
 ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketNotification。

put-bucket-ownership-controls

以下代码示例演示了如何使用 put-bucket-ownership-controls。

Amazon CLI

更新存储桶的存储桶所有权设置

以下 put-bucket-ownership-controls 示例更新存储桶的存储桶所有权设置。

```
aws s3api put-bucket-ownership-controls \
    --bucket amzn-s3-demo-bucket \
```

```
    --ownership-controls="Rules=[{ObjectOwnership=BucketOwnerEnforced}]"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon S3 用户指南》中的设置现有存储桶的对象所有权。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketOwnershipControls。

put-bucket-policy

以下代码示例演示了如何使用 put-bucket-policy。

Amazon CLI

此示例允许所有用户检索 amzn-s3-demo-bucket 中的任何对象，但 MySecretFolder 中的对象除外。它还向 Amazon 账户 1234-5678-9012 的根用户授予 put 和 delete 权限：

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json

policy.json:
{
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      },
      {
         "Effect": "Deny",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/MySecretFolder/*"
      },
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
         },
         "Action": [
            "s3:DeleteObject",
            "s3:PutObject"
         ],
```

```
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }

    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketPolicy。

put-bucket-replication

以下代码示例演示了如何使用 put-bucket-replication。

Amazon CLI

为 S3 存储桶配置复制

以下 put-bucket-replication 示例将复制配置应用于指定的 S3 存储桶。

```
aws s3api put-bucket-replication \
    --bucket amzn-s3-demo-bucket1 \
    --replication-configuration file://replication.json
```

的内容replication.json：

```
{
    "Role": "arn:aws:iam::123456789012:role/s3-replication-role",
    "Rules": [
        {
            "Status": "Enabled",
            "Priority": 1,
            "DeleteMarkerReplication": { "Status": "Disabled" },
            "Filter" : { "Prefix": ""},
            "Destination": {
                "Bucket": "arn:aws:s3:::amzn-s3-demo-bucket2"
            }
        }
    ]
}
```

目标存储桶必须已启用版本控制。指定的角色必须具有写入目标存储桶的权限，并且必须建立允许 Amazon S3 代入角色的信任关系。

示例角色权限策略：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetReplicationConfiguration",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObjectVersion",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ReplicateObject",
                "s3:ReplicateDelete",
                "s3:ReplicateTags"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/*"
        }
    ]
}
```

示例信任关系策略：

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Storage Service 控制台用户指南》中的这是主题标题：

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketReplication。

put-bucket-request-payment

以下代码示例演示了如何使用 put-bucket-request-payment。

Amazon CLI

示例 1：为存储桶启用“申请方付款”配置

以下 put-bucket-request-payment 示例为指定的存储桶启用 requester pays。

```
aws s3api put-bucket-request-payment \
    --bucket amzn-s3-demo-bucket \
    --request-payment-configuration '{"Payer":"Requester"}'
```

此命令不生成任何输出。

示例 2：为存储桶禁用“申请方付款”配置

以下 put-bucket-request-payment 示例为指定的存储桶禁用 requester pays。

```
aws s3api put-bucket-request-payment \
    --bucket amzn-s3-demo-bucket \
    --request-payment-configuration '{"Payer":"BucketOwner"}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketRequestPayment。

put-bucket-tagging

以下代码示例演示了如何使用 put-bucket-tagging。

Amazon CLI

以下命令将标记配置应用于名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api put-bucket-tagging --bucket amzn-s3-demo-bucket --tagging file://
tagging.json
```

文件 tagging.json 是当前文件夹中指定标签的 JSON 文档：

```
{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

或者，直接从命令行将标记配置应用于 amzn-s3-demo-bucket：

```
aws s3api put-bucket-tagging --bucket amzn-s3-demo-bucket --tagging
 'TagSet=[{Key=organization,Value=marketing}]'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketTagging。

put-bucket-versioning

以下代码示例演示了如何使用 put-bucket-versioning。

Amazon CLI

以下命令对于名为 amzn-s3-demo-bucket 的存储桶启用版本控制。

```
aws s3api put-bucket-versioning --bucket amzn-s3-demo-bucket --versioning-
configuration Status=Enabled
```

以下命令启用版本控制，并使用 mfa 代码

```
aws s3api put-bucket-versioning --bucket amzn-s3-demo-bucket --versioning-
configuration Status=Enabled --mfa "SERIAL 123456"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketVersioning。

put-bucket-website

以下代码示例演示了如何使用 put-bucket-website。

Amazon CLI

将静态网站配置应用于名为 amzn-s3-demo-bucket 的存储桶：

```
aws s3api put-bucket-website --bucket amzn-s3-demo-bucket --website-
configuration file://website.json
```

文件 website.json 是当前文件夹中的一个 JSON 文档，用于指定网站的索引和错误页面：

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutBucketWebsite。

put-object-acl

以下代码示例演示了如何使用 put-object-acl。

Amazon CLI

以下命令向两个 Amazon 用户（user1@example.com 和 user2@example.com）授予 full control 权限，并向所有人授予 read 权限：

```
aws s3api put-object-acl --bucket amzn-s3-demo-bucket --key file.txt --grant-full-
control emailaddress=user1@example.com,emailaddress=user2@example.com --grant-
read uri=http://acs.amazonaws.com/groups/global/AllUsers
```

有关自定义 ACL（s3api ACL 命令（如 put-object-acl）使用相同的速记参数表示法）的详细信息，请参阅 http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectAcl。

put-object-legal-hold

以下代码示例演示了如何使用 put-object-legal-hold。

Amazon CLI

对对象应用法定保留

以下 put-object-legal-hold 示例在 doc1.rtf 对象上设置了法定保留。

```
aws s3api put-object-legal-hold \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf \
    --legal-hold Status=ON
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectLegalHold。

put-object-lock-configuration

以下代码示例演示了如何使用 put-object-lock-configuration。

Amazon CLI

在存储桶上设置对象锁定配置

以下 put-object-lock-configuration 示例在指定存储桶上设置了 50 天的对象锁定。

```
aws s3api put-object-lock-configuration \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule":
 { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectLockConfiguration。

put-object-retention

以下代码示例演示了如何使用 put-object-retention。

Amazon CLI

为对象设置对象保留配置

以下 put-object-retention 示例为指定对象设置直到 2025 年 1 月 1 日的对象保留配置。

```
aws s3api put-object-retention \
    --bucket amzn-s3-demo-bucket-with-object-lock \
    --key doc1.rtf \
    --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectRetention。

put-object-tagging

以下代码示例演示了如何使用 put-object-tagging。

Amazon CLI

为对象设置标签

以下 put-object-tagging 示例会在指定对象上设置带有键 designation 和值 confidential 的标签。

```
aws s3api put-object-tagging \
    --bucket amzn-s3-demo-bucket \
    --key doc1.rtf \
    --tagging '{"TagSet": [{ "Key": "designation", "Value": "confidential" }]}'
```

此命令不生成任何输出。

以下 put-object-tagging 示例在指定对象上设置多个标签集。

```
aws s3api put-object-tagging \
    --bucket amzn-s3-demo-bucket-example \
    --key doc3.rtf \
    --tagging '{"TagSet": [{ "Key": "designation", "Value": "confidential" },
 { "Key": "department", "Value": "finance" }, { "Key": "team", "Value":
 "payroll" } ]}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObjectTagging。

put-object

以下代码示例演示了如何使用 put-object。

Amazon CLI

示例 1：将对象上传到 Amazon S3

以下 put-object 命令示例将对象上传到 Amazon S3。

```
aws s3api put-object \
    --bucket amzn-s3-demo-bucket \
    --key my-dir/MySampleImage.png \
    --body MySampleImage.png
```

有关上传对象的更多信息，请参阅《Amazon S3 开发人员指南》中的“上传对象”<http://docs.aws.amazon.com/AmazonS3/latest/dev/UploadingObjects.html>。

示例 2：将视频文件上传到 Amazon S3

以下 put-object 命令示例上传视频文件。

```
aws s3api put-object \
    --bucket amzn-s3-demo-bucket \
    --key my-dir/big-video-file.mp4 \
    --body /media/videos/f-sharp-3-data-services.mp4
```

有关上传对象的更多信息，请参阅《Amazon S3 开发人员指南》中的“上传对象”<http://docs.aws.amazon.com/AmazonS3/latest/dev/UploadingObjects.html>。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObject。

put-public-access-block

以下代码示例演示了如何使用 put-public-access-block。

Amazon CLI

为存储桶设置阻止公共访问配置

以下 put-public-access-block 示例为指定存储桶设置限制性阻止公共访问配置。

```
aws s3api put-public-access-block \
    --bucket amzn-s3-demo-bucket \
    --public-access-block-
configuration "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPub
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPublicAccessBlock。

rb

以下代码示例演示了如何使用 rb。

Amazon CLI

示例 1：删除存储桶

以下 rb 命令将移除存储桶。在此示例中，用户的存储桶为 amzn-s3-demo-bucket。请注意，存储桶必须为空才能移除：

```
aws s3 rb s3://amzn-s3-demo-bucket
```

输出：

```
remove_bucket: amzn-s3-demo-bucket
```

示例 2：强制删除存储桶

以下 rb 命令使用 --force 参数，首先移除存储桶中的所有对象，然后移除存储桶本身。在此示例中，用户的存储桶为 amzn-s3-demo-bucket，amzn-s3-demo-bucket 中的对象为 test1.txt 和 test2.txt：

```
aws s3 rb s3://amzn-s3-demo-bucket \
    --force
```

输出：

```
delete: s3://amzn-s3-demo-bucket/test1.txt
delete: s3://amzn-s3-demo-bucket/test2.txt
remove_bucket: amzn-s3-demo-bucket
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Rb。

restore-object

以下代码示例演示了如何使用 restore-object。

Amazon CLI

为对象创建还原请求

以下 restore-object 示例将存储桶 my-glacier-bucket 的指定 Amazon S3 Glacier 对象还原为 10 天。

```
aws s3api restore-object \
    --bucket my-glacier-bucket \
    --key doc1.rtf \
    --restore-request Days=10
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreObject。

rm

以下代码示例演示了如何使用 rm。

Amazon CLI

示例 1：删除 S3 对象

以下 rm 命令将删除单个 S3 对象：

```
aws s3 rm s3://amzn-s3-demo-bucket/test2.txt
```

输出：

```
delete: s3://amzn-s3-demo-bucket/test2.txt
```

示例 2：删除存储桶中的所有内容

以下 rm 命令在传递参数 --recursive 时，会递归删除指定存储桶和前缀下的所有对象。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 test2.txt：

```
aws s3 rm s3://amzn-s3-demo-bucket \
    --recursive
```

输出：

```
delete: s3://amzn-s3-demo-bucket/test1.txt
delete: s3://amzn-s3-demo-bucket/test2.txt
```

示例 3：删除存储桶中的所有内容（“.jpg”文件除外

以下 rm 命令在传递参数 --recursive 时，会递归删除指定存储桶和前缀下的所有对象，同时使用 --exclude 参数将某些对象排除在外。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 test2.jpg：

```
aws s3 rm s3://amzn-s3-demo-bucket/ \
    --recursive \
    --exclude "*.jpg"
```

输出：

```
delete: s3://amzn-s3-demo-bucket/test1.txt
```

示例 4：删除存储桶中的所有内容（指定前缀下的对象除外

以下 rm 命令在传递参数 --recursive 时，会递归删除指定存储桶和前缀下的所有对象，同时使用 --exclude 参数将特定前缀下的所有对象排除在外。在此示例中，存储桶 amzn-s3-demo-bucket 中包含对象 test1.txt 和 another/test.txt：

```
aws s3 rm s3://amzn-s3-demo-bucket/ \
    --recursive \
    --exclude "another/*"
```

输出：

```
delete: s3://amzn-s3-demo-bucket/test1.txt
```

示例 5：从 S3 接入点中删除对象

以下 rm 命令从接入点（myaccesspoint）中删除单个对象（mykey）。：：以下 rm 命令从接入点（myaccesspoint）中删除单个对象（mykey）。

```
aws s3 rm s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey
```

输出：

```
delete: s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Rm。

select-object-content

以下代码示例演示了如何使用 select-object-content。

Amazon CLI

基于 SQL 语句筛选 Amazon S3 对象的内容

以下 select-object-content 示例使用指定的 SQL 语句筛选 my-data-file.csv 对象并将输出发送到文件。

```
aws s3api select-object-content \
    --bucket amzn-s3-demo-bucket \
    --key my-data-file.csv \
    --expression "select * from s3object limit 100" \
    --expression-type 'SQL' \
    --input-serialization '{"CSV": {}, "CompressionType": "NONE"}' \
    --output-serialization '{"CSV": {}}' "output.csv"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SelectObjectContent。

sync

以下代码示例演示了如何使用 sync。

Amazon CLI

示例 1：将所有本地对象同步到指定存储桶

以下 sync 命令通过将本地文件上传到 S3，将本地目录中的对象同步到指定的前缀和存储桶。如果本地文件的大小与 S3 对象的大小不同，本地文件的上次修改时间晚于 S3 对象的上次修改时间，或者本地文件在指定的存储桶和前缀下不存在，则需要上传本地文件。在此示例中，用户将存储桶 amzn-s3-demo-bucket 同步到了当前本地目录中。当前本地目录包含文件 test.txt 和 test2.txt。存储桶 amzn-s3-demo-bucket 不包含任何对象。

```
aws s3 sync . s3://amzn-s3-demo-bucket
```

输出：

```
upload: test.txt to s3://amzn-s3-demo-bucket/test.txt
upload: test2.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 2：将指定的 S3 存储桶中的所有 S3 对象同步到另一个存储桶

以下 sync 命令通过复制 S3 对象，将指定前缀和存储桶下的对象同步到另一个指定前缀和存储桶下的对象。如果两个 S3 对象的大小不同，源的上次修改时间晚于目标的上次修改时间，或者 S3 对象在指定的存储桶和前缀目标下不存在，则需要复制 S3 对象。

在此示例中，用户将存储桶 amzn-s3-demo-bucket 同步到了存储桶 amzn-s3-demo-bucket2。存储桶 amzn-s3-demo-bucket 包含对象 test.txt 和 test2.txt。存储桶 amzn-s3-demo-bucket2 不包含任何对象：

```
aws s3 sync s3://amzn-s3-demo-bucket s3://amzn-s3-demo-bucket2
```

输出：

```
copy: s3://amzn-s3-demo-bucket/test.txt to s3://amzn-s3-demo-bucket2/test.txt
```

```
copy: s3://amzn-s3-demo-bucket/test2.txt to s3://amzn-s3-demo-bucket2/test2.txt
```

示例 3：将指定 S3 存储桶中的所有 S3 对象同步到本地目录

以下 sync 命令通过下载 S3 对象，将指定 S3 存储桶中的文件同步到本地目录。如果 S3 对象的大小与本地文件的大小不同，S3 对象的上次修改时间晚于本地文件的上次修改时间，或者 S3 对象在本地目录中不存在，则需要下载 S3 对象。请注意，从 S3 下载对象时，本地文件的上次修改时间将更改为 S3 对象的上次修改时间。在此示例中，用户将存储桶 amzn-s3-demo-bucket 同步到了当前本地目录中。存储桶 amzn-s3-demo-bucket 包含对象 test.txt 和 test2.txt。当前本地目录中没有文件：

```
aws s3 sync s3://amzn-s3-demo-bucket .
```

输出：

```
download: s3://amzn-s3-demo-bucket/test.txt to test.txt
download: s3://amzn-s3-demo-bucket/test2.txt to test2.txt
```

示例 4：将所有本地对象同步到指定存储桶并删除所有不匹配的文件

以下 sync 命令通过将本地文件上传到 S3，将指定前缀和存储桶下的对象同步到本地目录中的文件。由于 --delete 参数的原因，任何存在于指定前缀和存储桶下但不存在于本地目录中的文件都将被删除。在此示例中，用户将存储桶 amzn-s3-demo-bucket 同步到了当前本地目录中。当前本地目录包含文件 test.txt 和 test2.txt。存储桶 amzn-s3-demo-bucket 包含对象 test3.txt：

```
aws s3 sync . s3://amzn-s3-demo-bucket \
    --delete
```

输出：

```
upload: test.txt to s3://amzn-s3-demo-bucket/test.txt
upload: test2.txt to s3://amzn-s3-demo-bucket/test2.txt
delete: s3://amzn-s3-demo-bucket/test3.txt
```

示例 5：将除“.jpg”文件之外的所有本地对象同步到指定存储桶

以下 sync 命令通过将本地文件上传到 S3，将指定前缀和存储桶下的对象同步到本地目录中的文件。由于 --exclude 参数的原因，所有与 S3 和本地存在的模式相匹配的文件都不会同步。在此

示例中，用户将存储桶 amzn-s3-demo-bucket 同步到了当前本地目录中。当前本地目录包含文件 test.jpg 和 test2.txt。存储桶 amzn-s3-demo-bucket 中的对象 test.jpg 与本地 test.jpg 的大小不同：

```
aws s3 sync . s3://amzn-s3-demo-bucket \
    --exclude "*.jpg"
```

输出：

```
upload: test2.txt to s3://amzn-s3-demo-bucket/test2.txt
```

示例 6：将除指定的目录文件之外的所有本地对象同步到指定存储桶

以下 sync 命令通过下载 S3 对象，将本地目录下的文件同步到指定前缀和存储桶下的对象。此示例使用 --exclude 参数标志，将指定目录和 S3 前缀排除在 sync 命令之外。在此示例中，用户将当前本地目录同步到了存储桶 amzn-s3-demo-bucket 中。当前本地目录包含文件 test.txt 和 another/test2.txt。存储桶 amzn-s3-demo-bucket 包含对象 another/test5.txt 和 test1.txt：

```
aws s3 sync s3://amzn-s3-demo-bucket/ . \
    --exclude "*another/*"
```

输出：

```
download: s3://amzn-s3-demo-bucket/test1.txt to test1.txt
```

示例 7：在不同区域的存储桶之间同步所有对象

以下 sync 命令可在不同区域的两个存储桶之间同步文件：

```
aws s3 sync s3://my-us-west-2-bucket s3://my-us-east-1-bucket \
    --source-region us-west-2 \
    --region us-east-1
```

输出：

```
download: s3://my-us-west-2-bucket/test1.txt to s3://my-us-east-1-bucket/test1.txt
```

示例 8：同步到 S3 接入点

以下 sync 命令会将当前目录同步到接入点（myaccesspoint）：

```
aws s3 sync . s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
```

输出：

```
upload: test.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/test.txt
upload: test2.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/test2.txt
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Sync。

upload-part-copy

以下代码示例演示了如何使用 upload-part-copy。

Amazon CLI

从现有对象复制数据作为数据来源，上传部分对象

以下 upload-part-copy 示例会通过从现有对象复制数据作为数据来源，上传部分对象。

```
aws s3api upload-part-copy \
    --bucket amzn-s3-demo-bucket \
    --key "Map_Data_June.mp4" \
    --copy-source "amzn-s3-demo-bucket/copy_of_Map_Data_June.mp4" \
    --part-number 1 \
    --upload-
id "bq0tdE1CDpWQYRPLHuNG50xAT6pA5D.m_RiBy0ggOH6b13pVRY7QjvLlf75iFdJqp_2wztk5hvpUM2SesXgrzbeh
```

输出：

```
{
    "CopyPartResult": {
        "LastModified": "2019-12-13T23:16:03.000Z",
        "ETag": "\"711470fc377698c393d94aed6305e245\""
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadPartCopy。

upload-part

以下代码示例演示了如何使用 upload-part。

Amazon CLI

以下命令上传使用 create-multipart-upload 命令启动的分段上传中的第一个分段：

```
aws s3api upload-part --bucket amzn-s3-demo-bucket --key
 'multipart/01' --part-number 1 --body part01 --upload-id
  "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3UR
```

body 选项采用本地文件的名称或路径进行上传（不要使用 file:// 前缀）。最小分段大小为 5 MB。上传 ID 由 create-multipart-upload 返回，也可以使用 list-multipart-uploads 进行检索。存储桶和键是在您创建分段上传时指定的。

输出：

```
{
    "ETag": "\"e868e0f4719e394144ef36531ee6824c\""
}
```

保存每个分段的 ETag 值以备后用。需要这些值才能完成分段上传。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UploadPart。

website

以下代码示例演示了如何使用 website。

Amazon CLI

将 S3 存储桶配置为静态网站

以下命令将名为 amzn-s3-demo-bucket 的存储桶配置为静态网站。索引文档选项指定了访客在导航到网站 URL 时将定向到的 amzn-s3-demo-bucket 文件。在此示例中，存储桶位于 us-west-2 区域中，因此网站将显示为 http://amzn-s3-demo-bucket.s3-website-us-west-2.amazonaws.com。

存储桶中显示在静态站点中的所有文件都必须配置为允许访客打开。文件权限与存储桶网站配置分开配置。

```
aws s3 website s3://amzn-s3-demo-bucket/ \
    --index-document index.html \
    --error-document error.html
```

有关在 Amazon S3 中托管静态网站的信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的托管静态网站。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Website。

使用 Amazon CLI 的 Amazon S3 控件示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon S3 控件结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-access-point`

以下代码示例演示了如何使用 `create-access-point`。

Amazon CLI

创建接入点

以下 `create-access-point` 示例为账户 123456789012 中的存储桶 `business-records` 创建名为 `finance-ap` 的接入点。在运行此示例之前，请针对您的使用案例将接入点名称、存储桶名称和账号替换为适合的值。

```
aws s3control create-access-point \
    --account-id 123456789012 \
```

```
    --bucket business-records \
    --name finance-ap
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的创建接入点。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccessPoint。

create-job

以下代码示例演示了如何使用 create-job。

Amazon CLI

创建 Amazon S3 批量操作任务

以下 create-job 示例创建一个 Amazon S3 批量操作任务，将对象标记为 confidential` in the bucket ``employee-records。

```
aws s3control create-job \
    --account-id 123456789012 \
    --operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"confidential",
 "Value":"true"}] }}' \
    --report '{"Bucket":"arn:aws:s3:::employee-records-logs","Prefix":"batch-op-
create-job",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":{"ObjectArn":"arn:aws:s3:::employee-records-logs/inv-
report/7a6a9be4-072c-407e-85a2-
ec3e982f773e.csv","ETag":"69f52a4e9f797e987155d9c8f5880897"}}' \
    --priority 42 \
    --role-arn arn:aws:iam::123456789012:role/S3BatchJobRole
```

输出：

```
{
    "JobId": "93735294-df46-44d5-8638-6356f335324e"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateJob。

delete-access-point-policy

以下代码示例演示了如何使用 delete-access-point-policy。

Amazon CLI

删除接入点策略

以下 delete-access-point-policy 示例删除账户 123456789012 中名为 finance-ap 的接入点的接入点策略。在运行此示例之前，请针对您的使用案例将接入点名称和账号替换为适合的值。

```
aws s3control delete-access-point-policy \
    --account-id 123456789012 \
    --name finance-ap
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessPointPolicy。

delete-access-point

以下代码示例演示了如何使用 delete-access-point。

Amazon CLI

删除接入点

以下 delete-access-point 示例在账户 123456789012 中删除名为 finance-ap 的接入点。在运行此示例之前，请针对您的使用案例将接入点名称和账号替换为适合的值。

```
aws s3control delete-access-point \
    --account-id 123456789012 \
    --name finance-ap
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessPoint。

delete-public-access-block

以下代码示例演示了如何使用 delete-public-access-block。

Amazon CLI

删除账户的屏蔽公共访问权限设置

以下 delete-public-access-block 示例删除指定账户的屏蔽公共访问权限设置。

```
aws s3control delete-public-access-block \
    --account-id 123456789012
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePublicAccessBlock。

describe-job

以下代码示例演示了如何使用 describe-job。

Amazon CLI

描述 Amazon S3 批量操作任务

以下 describe-job 命令可为指定的批量操作任务提供配置参数和状态。

```
aws s3control describe-job \
    --account-id 123456789012 \
    --job-id 93735294-df46-44d5-8638-6356f335324e
```

输出：

```
{
    "Job": {
        "TerminationDate": "2019-10-03T21:49:53.944Z",
        "JobId": "93735294-df46-44d5-8638-6356f335324e",
        "FailureReasons": [],
        "Manifest": {
            "Spec": {
```

```
                "Fields": [
                    "Bucket",
                    "Key"
                ],
                "Format": "S3BatchOperations_CSV_20180820"
            },
            "Location": {
                "ETag": "69f52a4e9f797e987155d9c8f5880897",
                "ObjectArn": "arn:aws:s3:::employee-records-logs/inv-
report/7a6a9be4-072c-407e-85a2-ec3e982f773e.csv"
            }
        },
        "Operation": {
            "S3PutObjectTagging": {
                "TagSet": [
                    {
                        "Value": "true",
                        "Key": "confidential"
                    }
                ]
            }
        },
        "RoleArn": "arn:aws:iam::123456789012:role/S3BatchJobRole",
        "ProgressSummary": {
            "TotalNumberOfTasks": 8,
            "NumberOfTasksFailed": 0,
            "NumberOfTasksSucceeded": 8
        },
        "Priority": 42,
        "Report": {
            "ReportScope": "AllTasks",
            "Format": "Report_CSV_20180820",
            "Enabled": true,
            "Prefix": "batch-op-create-job",
            "Bucket": "arn:aws:s3:::employee-records-logs"
        },
        "JobArn": "arn:aws:s3:us-west-2:123456789012:job/93735294-
df46-44d5-8638-6356f335324e",
        "CreationTime": "2019-10-03T21:48:48.048Z",
        "Status": "Complete"
    }
}
```

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeJob。

get-access-point-policy-status

以下代码示例演示了如何使用 get-access-point-policy-status。

Amazon CLI

检索接入点策略状态

以下 get-access-point-policy-status 示例检索账户 123456789012 中名为 finance-ap 的接入点的接入点策略状态。接入点策略状态表示了该接入点的策略是否允许公共访问。在运行此示例之前，请针对您的使用案例将接入点名称和账号替换为适合的值。

```
aws s3control get-access-point-policy-status \
    --account-id 123456789012 \
    --name finance-ap
```

输出：

```
{
    "PolicyStatus": {
        "IsPublic": false
    }
}
```

有关何时将接入点策略视为公共策略的更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中“公共”的含义。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessPointPolicyStatus。

get-access-point-policy

以下代码示例演示了如何使用 get-access-point-policy。

Amazon CLI

检索接入点策略

以下 get-access-point-policy 示例检索账户 123456789012 中名为 finance-ap 的接入点的接入点策略。在运行此示例之前，请针对您的使用案例将接入点名称和账号替换为适合的值。

```
aws s3control get-access-point-policy \
    --account-id 123456789012 \
```

```
    --name finance-ap
```

输出：

```
{
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow
\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:role/Admin\"},\"Action\":
\"s3:GetObject\",\"Resource\":\"arn:aws:s3:us-west-2:123456789012:accesspoint/
finance-ap/object/records/*\"}]}"
}
```

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessPointPolicy。

get-access-point

以下代码示例演示了如何使用 get-access-point。

Amazon CLI

检索接入点配置详细信息

以下 get-access-point 示例检索账户 123456789012 中名为 finance-ap 的接入点的配置详细信息。在运行此示例之前，请针对您的使用案例将接入点名称和账号替换为适合的值。

```
aws s3control get-access-point \
    --account-id 123456789012 \
    --name finance-ap
```

输出：

```
{
    "Name": "finance-ap",
    "Bucket": "business-records",
    "NetworkOrigin": "Internet",
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": false,
        "IgnorePublicAcls": false,
        "BlockPublicPolicy": false,
        "RestrictPublicBuckets": false
```

```
    },
    "CreationDate": "2020-01-01T00:00:00Z"
}
```

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessPoint。

get-multi-region-access-point-routes

以下代码示例演示了如何使用 get-multi-region-access-point-routes。

Amazon CLI

查询当前多区域接入点路由配置

以下示例 get-multi-region-access-point-routes 将返回指定多区域接入点的当前路由配置。

```
aws s3control get-multi-region-access-point-routes \
    --region Region \
    --account-id 111122223333 \
    --mrap MultiRegionAccessPoint_ARN
```

输出：

```
{
    "Mrap": "arn:aws:s3::111122223333:accesspoint/0000000000000.mrap",
    "Routes": [
        {
            "Bucket": "amzn-s3-demo-bucket1",
            "Region": "ap-southeast-2",
            "TrafficDialPercentage": 100
        },
        {
            "Bucket": "amzn-s3-demo-bucket2",
            "Region": "us-west-1",
            "TrafficDialPercentage": 0
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMultiRegionAccessPointRoutes。

get-public-access-block

以下代码示例演示了如何使用 get-public-access-block。

Amazon CLI

列出账户的屏蔽公共访问权限设置

以下 get-public-access-block 示例显示指定账户的屏蔽公共访问权限设置。

```
aws s3control get-public-access-block \
    --account-id 123456789012
```

输出：

```
{
   "PublicAccessBlockConfiguration": {
      "BlockPublicPolicy": true,
      "RestrictPublicBuckets": true,
      "IgnorePublicAcls": true,
      "BlockPublicAcls": true
   }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPublicAccessBlock。

list-access-points

以下代码示例演示了如何使用 list-access-points。

Amazon CLI

示例 1：检索账户的所有接入点列表

以下 list-access-points 示例显示挂载到账户 123456789012 所拥有的存储桶的所有接入点列表。

```
aws s3control list-access-points \
```

```
    --account-id 123456789012
```

输出：

```
{
    "AccessPointList": [
        {
            "Name": "finance-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "business-records"
        },
        {
            "Name": "managers-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "business-records"
        },
        {
            "Name": "private-network-ap",
            "NetworkOrigin": "VPC",
            "VpcConfiguration": {
                "VpcId": "1a2b3c"
            },
            "Bucket": "business-records"
        },
        {
            "Name": "customer-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "external-docs"
        },
        {
            "Name": "public-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "external-docs"
        }
    ]
}
```

示例 2：检索存储桶的所有接入点列表

以下 `list-access-points` 示例检索挂载到账户 123456789012 所拥有的存储桶 `external-docs` 的所有接入点列表。

```
aws s3control list-access-points \
```

```
    --account-id 123456789012 \
    --bucket external-docs
```

输出：

```
{
    "AccessPointList": [
        {
            "Name": "customer-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "external-docs"
        },
        {
            "Name": "public-ap",
            "NetworkOrigin": "Internet",
            "Bucket": "external-docs"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessPoints。

list-jobs

以下代码示例演示了如何使用 list-jobs。

Amazon CLI

列出账户 Amazon S3 批量操作任务

以下 list-jobs 示例列出了指定账户最近的所有批量操作任务。

```
aws s3control list-jobs \
    --account-id 123456789012
```

输出：

```
{
```

```
    "Jobs": [
        {
            "Operation": "S3PutObjectTagging",
            "ProgressSummary": {
                "NumberOfTasksFailed": 0,
                "NumberOfTasksSucceeded": 8,
                "TotalNumberOfTasks": 8
            },
            "CreationTime": "2019-10-03T21:48:48.048Z",
            "Status": "Complete",
            "JobId": "93735294-df46-44d5-8638-6356f335324e",
            "Priority": 42
        },
        {
            "Operation": "S3PutObjectTagging",
            "ProgressSummary": {
                "NumberOfTasksFailed": 0,
                "NumberOfTasksSucceeded": 0,
                "TotalNumberOfTasks": 0
            },
            "CreationTime": "2019-10-03T21:46:07.084Z",
            "Status": "Failed",
            "JobId": "3f3c7619-02d3-4779-97f6-1d98dd313108",
            "Priority": 42
        },
    ]
}
```

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

put-access-point-policy

以下代码示例演示了如何使用 put-access-point-policy。

Amazon CLI

设置接入点策略

以下 put-access-point-policy 示例为账户 123456789012 中的接入点 finance-ap 指定相应的接入点策略。如果接入点 finance-ap 已有策略，则此命令会将现有策略替换为该命令中指定的策略。在运行此示例之前，请针对您的使用案例将账号、接入点名称和策略说明替换为适合的值。

```
aws s3control put-access-point-policy \
    --account-id 123456789012 \
    --name finance-ap \
    --policy file://ap-policy.json
```

的内容ap-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Alice"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/finance-ap/
object/Alice/*"
        }
    ]
}
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的使用 Amazon S3 接入点管理数据访问。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAccessPointPolicy。

put-public-access-block

以下代码示例演示了如何使用 put-public-access-block。

Amazon CLI

编辑账户的屏蔽公共访问权限设置

以下 put-public-access-block 示例会将指定账户的所有屏蔽公共访问权限设置切换为 true。

```
aws s3control put-public-access-block \
```

```
    --account-id 123456789012 \
    --public-access-block-configuration '{"BlockPublicAcls": true,
"IgnorePublicAcls": true, "BlockPublicPolicy": true, "RestrictPublicBuckets":
true}'
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutPublicAccessBlock。

submit-multi-region-access-point-routes

以下代码示例演示了如何使用 submit-multi-region-access-point-routes。

Amazon CLI

更新您的多区域接入点路由配置

以下 submit-multi-region-access-point-routes 示例更新您的多区域接入点中 ap-southeast-2 区域的 amzn-s3-demo-bucket1 和 amzn-s3-demo-bucket2 路由状态。

```
aws s3control submit-multi-region-access-point-routes \
    --region ap-southeast-2 \
    --account-id 111122223333 \
    --mrap MultiRegionAccessPoint_ARN \
    --route-updates Bucket=amzn-s3-demo-
bucket1,TrafficDialPercentage=100 Bucket=amzn-s3-demo-
bucket2,TrafficDialPercentage=0
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SubmitMultiRegionAccessPointRoutes。

update-job-priority

以下代码示例演示了如何使用 update-job-priority。

Amazon CLI

更新 Amazon S3 批量操作任务的任务优先级

以下 update-job-priority 示例会将指定的任务更新为新的优先级。

```
aws s3control update-job-priority \
    --account-id 123456789012 \
    --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \
    --priority 52
```

输出：

```
{
    "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386",
    "Priority": 52
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJobPriority。

update-job-status

以下代码示例演示了如何使用 update-job-status。

Amazon CLI

更新 Amazon S3 批量操作任务的状态

以下 update-job-status 示例取消正在等待批准的指定任务。

```
aws s3control update-job-status \
    --account-id 123456789012 \
    --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \
    --requested-job-status Cancelled
```

输出：

```
{
    "Status": "Cancelled",
    "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386"
}
```

以下 update-job-status 示例确认并运行正在等待批准的指定任务。

```
aws s3control update-job-status \
```

```
    --account-id 123456789012 \
    --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \
    --requested-job-status Ready

Output::

{
    "Status": "Ready",
    "JobId": "5782949f-3301-4fb3-
be34-8d5bab54dbca"
}
```

以下 update-job-status 示例取消正在运行的指定任务。

```
 aws s3control update-job-status \
    --account-id 123456789012 \
    --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \
    --requested-job-status Cancelled

Output::
{
         "Status": "Cancelling",
         "JobId": "5782949f-3301-4fb3-be34-8d5bab54dbca"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateJobStatus。

使用Amazon CLI的 Secrets Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Secrets Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

batch-get-secret-value

以下代码示例演示了如何使用 batch-get-secret-value。

Amazon CLI

示例 1：检索按名称列出的一组密钥的密钥值

以下 batch-get-secret-value 示例获取三个密钥的密钥值。

```
aws secretsmanager batch-get-secret-value \
    --secret-id-list MySecret1 MySecret2 MySecret3
```

输出：

```
{
    "SecretValues": [
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret1-
a1b2c3",
            "Name": "MySecret1",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
            "SecretString": "{\"username\":\"diego_ramirez\",\"password\":\"EXAMPLE-
PASSWORD\",\"engine\":\"mysql\",\"host\":\"secretsmanagertutorial.cluster.us-
west-2.rds.amazonaws.com\",\"port\":3306,\"dbClusterIdentifier\":
\"secretsmanagertutorial\"}",
            "VersionStages": [
                "AWSCURRENT"
            ],
            "CreatedDate": "1523477145.729"
        },
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret2-
a1b2c3",
            "Name": "MySecret2",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
            "SecretString": "{\"username\":\"akua_mansa\",\"password\":\"EXAMPLE-
PASSWORD\"",
            "VersionStages": [
                "AWSCURRENT"
            ],
```

```
            "CreatedDate": "1673477781.275"
        },
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret3-
a1b2c3",
            "Name": "MySecret3",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEccccc",
            "SecretString": "{\"username\":\"jie_liu\",\"password\":\"EXAMPLE-
PASSWORD\"",
            "VersionStages": [
                "AWSCURRENT"
            ],
            "CreatedDate": "1373477721.124"
        }
    ],
    "Errors": []
}
```

有关更多信息，请参阅《Amazon Secrets Manager 用户指南》中的批量检索一组密钥。

示例 2：检索筛选器选择的一组密钥的密钥值

以下 batch-get-secret-value 示例获取您的账户中名称包含 MySecret 的密钥值。按名称筛选区分大小写。

```
aws secretsmanager batch-get-secret-value \
    --filters Key="name",Values="MySecret"
```

输出：

```
{
    "SecretValues": [
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret1-
a1b2c3",
            "Name": "MySecret1",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
            "SecretString": "{\"username\":\"diego_ramirez\",\"password\":\"EXAMPLE-
PASSWORD\",\"engine\":\"mysql\",\"host\":\"secretsmanagertutorial.cluster.us-
west-2.rds.amazonaws.com\",\"port\":3306,\"dbClusterIdentifier\":
\"secretsmanagertutorial\"}",
            "VersionStages": [
```

```
                "AWSCURRENT"
            ],
            "CreatedDate": "1523477145.729"
        },
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret2-
a1b2c3",
            "Name": "MySecret2",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
            "SecretString": "{\"username\":\"akua_mansa\",\"password\":\"EXAMPLE-
PASSWORD\"",
            "VersionStages": [
                "AWSCURRENT"
            ],
            "CreatedDate": "1673477781.275"
        },
        {
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret3-
a1b2c3",
            "Name": "MySecret3",
            "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEccccc",
            "SecretString": "{\"username\":\"jie_liu\",\"password\":\"EXAMPLE-
PASSWORD\"",
            "VersionStages": [
                "AWSCURRENT"
            ],
            "CreatedDate": "1373477721.124"
        }
    ],
    "Errors": []
}
```

有关更多信息，请参阅《Amazon Secrets Manager 用户指南》中的批量检索一组密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetSecretValue。

`cancel-rotate-secret`

以下代码示例演示了如何使用 `cancel-rotate-secret`。

Amazon CLI

关闭密钥的自动轮换功能

以下 cancel-rotate-secret 示例关闭密钥的自动轮换功能。要恢复轮换，请调用 rotate-secret。

```
aws secretsmanager cancel-rotate-secret \
    --secret-id MyTestSecret
```

输出：

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
  "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelRotateSecret。

create-secret

以下代码示例演示了如何使用 create-secret。

Amazon CLI

示例 1：根据 JSON 文件中的凭证创建密钥

以下 create-secret 示例将根据文件中的凭证创建密钥。有关更多信息，请参阅《Amazon CLI 用户指南》中的从文件加载 Amazon CLI 参数。

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --secret-string file://mycreds.json
```

的内容mycreds.json：

```
{
  "engine": "mysql",
  "username": "saanvis",
  "password": "EXAMPLE-PASSWORD",
  "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",
  "dbname": "myDatabase",
```

```
  "port": "3306"
}
```

输出：

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
  "Name": "MyTestSecret",
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的创建密钥。

示例 2：创建密钥

以下 create-secret 示例将创建包含两个键值对的密钥。当您在命令 shell 中输入命令时，存在访问命令历史记录或实用程序可以访问您命令参数的风险。如果命令包含密钥的值，就会引起关注。有关更多信息，请参阅《Secrets Manager 用户指南》中的降低使用命令行工具存储密钥的风险。

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
```

输出：

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
  "Name": "MyTestSecret",
  "VersionId": "EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的创建密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSecret。

delete-resource-policy

以下代码示例演示了如何使用 delete-resource-policy。

Amazon CLI

删除附加到密钥的基于资源的策略

以下 delete-resource-policy 示例将删除附加到密钥的基于资源的策略。

```
aws secretsmanager delete-resource-policy \
    --secret-id MyTestSecret
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的身份验证和访问权限控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourcePolicy。

delete-secret

以下代码示例演示了如何使用 delete-secret。

Amazon CLI

示例 1：删除密钥

以下 delete-secret 示例将删除密钥。您可以使用 DeletionDate 恢复密钥，直到 restore-secret 响应字段中的日期和时间。要删除复制到其他区域的密钥，请先使用 remove-regions-from-replication 删除其副本，然后调用 delete-secret。

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --recovery-window-in-days 7
```

输出：

```
{
```

```
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "DeletionDate": 1524085349.095
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的删除密钥。

示例 2：立即删除密钥

以下 delete-secret 示例将立即删除密钥而没有恢复时段。您无法恢复此密钥。

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --force-delete-without-recovery
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "DeletionDate": 1508750180.309
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的删除密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSecret。

describe-secret

以下代码示例演示了如何使用 describe-secret。

Amazon CLI

检索密钥的详细信息

以下 describe-secret 示例显示密钥的详细信息。

```
aws secretsmanager describe-secret \
    --secret-id MyTestSecret
```

输出：


```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
Ca8JGt",
    "Name": "MyTestSecret",
    "Description": "My test secret",
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-
ba987EXAMPLE",
    "RotationEnabled": true,
    "RotationLambdaARN": "arn:aws:lambda:us-
west-2:123456789012:function:MyTestRotationLambda",
    "RotationRules": {
        "AutomaticallyAfterDays": 2,
        "Duration": "2h",
        "ScheduleExpression": "cron(0 16 1,15 * ? *)"
    },
    "LastRotatedDate": 1525747253.72,
    "LastChangedDate": 1523477145.729,
    "LastAccessedDate": 1524572133.25,
    "Tags": [
        {
            "Key": "SecondTag",
            "Value": "AnotherValue"
        },
        {
            "Key": "FirstTag",
            "Value": "SomeValue"
        }
    ],
    "VersionIdsToStages": {
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [
            "AWSPREVIOUS"
        ],
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222": [
            "AWSCURRENT"
        ],
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333": [
            "AWSPENDING"
        ]
    },
    "CreatedDate": 1521534252.66,
    "PrimaryRegion": "us-west-2",
    "ReplicationStatus": [
```

```
        {
            "Region": "eu-west-3",
            "KmsKeyId": "alias/aws/secretsmanager",
            "Status": "InSync",
            "StatusMessage": "Replication succeeded"
        }
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecret。

`get-random-password`

以下代码示例演示了如何使用 `get-random-password`。

Amazon CLI

生成随机密码

以下 `get-random-password` 示例生成一个长度为 20 个字符的随机密码，其中至少包含一个大写字母、小写字母、数字和标点符号。

```
aws secretsmanager get-random-password \
    --require-each-included-type \
    --password-length 20
```

输出：

```
{
    "RandomPassword": "EXAMPLE-PASSWORD"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的创建和管理密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRandomPassword。

`get-resource-policy`

以下代码示例演示了如何使用 `get-resource-policy`。

Amazon CLI

检索附加到密钥的基于资源的策略

以下 get-resource-policy 示例将检索附加到密钥的基于资源的策略。

```
aws secretsmanager get-resource-policy \
    --secret-id MyTestSecret
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "ResourcePolicy": "{\n\"Version\":\"2012-10-17\",\n\"Statement\":[{\n\"Effect\":
\"Allow\",\n
    \"Principal\":{\n\"AWS\":\"arn:aws:iam::123456789012:root\"\n},\n\"Action\":
    \"secretsmanager:GetSecretValue\",\n\"Resource\":\"*\"\n}]\n}"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的身份验证和访问权限控制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResourcePolicy。

get-secret-value

以下代码示例演示了如何使用 get-secret-value。

Amazon CLI

示例 1：检索密钥的加密密钥值

以下 get-secret-value 示例获取当前密钥值。

```
aws secretsmanager get-secret-value \
    --secret-id MyTestSecret
```

输出：

```
{
```

```
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "SecretString": "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1523477145.713
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的检索密钥。

示例 2：检索之前的密钥值

以下 `get-secret-value` 示例获取之前的密钥值。

```
aws secretsmanager get-secret-value \
    --secret-id MyTestSecret
    --version-stage AWSPREVIOUS
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "SecretString": "{\"user\":\"diegor\",\"password\":\"PREVIOUS-EXAMPLE-PASSWORD
\"}",
    "VersionStages": [
        "AWSPREVIOUS"
    ],
    "CreatedDate": 1523477145.713
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的检索密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSecretValue。

list-secret-version-ids

以下代码示例演示了如何使用 `list-secret-version-ids`。

Amazon CLI

列出与密钥相关的所有密钥版本

以下 `list-secret-version-ids` 示例获取密钥所有版本的列表。

```
aws secretsmanager list-secret-version-ids \
    --secret-id MyTestSecret
```

输出：

```
{
  "Versions": [
    {
        "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "VersionStages": [
            "AWSPREVIOUS"
        ],
        "LastAccessedDate": 1523477145.713,
        "CreatedDate": 1523477145.713
    },
    {
        "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "VersionStages": [
            "AWSCURRENT"
        ],
        "LastAccessedDate": 1523477145.713,
        "CreatedDate": 1523486221.391
    },
    {
        "CreatedDate": 1.51197446236E9,
        "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333;"
    }
    ],
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecretVersionIds。

list-secrets

以下代码示例演示了如何使用 list-secrets。

Amazon CLI

示例 1：列出您账户中的密钥

以下 list-secrets 示例获取了您账户中的密钥列表。

```
aws secretsmanager list-secrets
```

输出：

```
{
    "SecretList": [
        {
            "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-a1b2c3",
            "Name": "MyTestSecret",
            "LastChangedDate": 1523477145.729,
            "SecretVersionsToStages": {
                "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [
                    "AWSCURRENT"
                ]
            }
        },
        {
            "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:AnotherSecret-d4e5f6",
            "Name": "AnotherSecret",
            "LastChangedDate": 1523482025.685,
            "SecretVersionsToStages": {
                "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222": [
                    "AWSCURRENT"
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的查找密钥。

示例 2：筛选您账户中的密钥列表

以下 `list-secrets` 示例将获取您的账户中名称包含 `Test` 的密钥列表。按名称筛选区分大小写。

```
aws secretsmanager list-secrets \
    --filter Key="name",Values="Test"
```

输出：

```
{
    "SecretList": [
        {
            "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-a1b2c3",
            "Name": "MyTestSecret",
            "LastChangedDate": 1523477145.729,
            "SecretVersionsToStages": {
                "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [
                    "AWSCURRENT"
                ]
            }
        }
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的查找密钥。

示例 3：列出您账户中由其他服务管理的密钥

以下 `list-secrets` 示例返回您账户中由 Amazon RDS 管理的密钥。

```
aws secretsmanager list-secrets \
    --filter Key="owning-service",Values="rds"
```

输出：

```
{
    "SecretList": [
        {
            "Name": "rds!cluster-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Tags": [
```

```
                {
                    "Value": "arn:aws:rds:us-
west-2:123456789012:cluster:database-1",
                    "Key": "aws:rds:primaryDBClusterArn"
                },
                {
                    "Value": "rds",
                    "Key": "aws:secretsmanager:owningService"
                }
            ],
            "RotationRules": {
                "AutomaticallyAfterDays": 1
            },
            "LastChangedDate": 1673477781.275,
            "LastRotatedDate": 1673477781.26,
            "SecretVersionsToStages": {
                "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa": [
                    "AWSPREVIOUS"
                ],
                "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb": [
                    "AWSCURRENT",
                    "AWSPENDING"
                ]
            },
            "OwningService": "rds",
            "RotationEnabled": true,
            "CreatedDate": 1673467300.7,
            "LastAccessedDate": 1673395200.0,
            "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:rds!
cluster-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111-a1b2c3",
            "Description": "Secret associated with primary RDS DB cluster:
 arn:aws:rds:us-west-2:123456789012:cluster:database-1"
        }
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的由其他服务管理的密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecrets。

put-resource-policy

以下代码示例演示了如何使用 put-resource-policy。

Amazon CLI

将基于资源的策略添加到密钥

以下 put-resource-policy 示例将向密钥添加权限策略，首先检查该策略是否不提供对该密钥的广泛访问权限。该策略是从文件中读取的。有关更多信息，请参阅《Amazon CLI 用户指南》中的从文件加载 Amazon CLI 参数。

```
aws secretsmanager put-resource-policy \
    --secret-id MyTestSecret \
    --resource-policy file://mypolicy.json \
    --block-public-policy
```

的内容mypolicy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的将权限策略附加到密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutResourcePolicy。

put-secret-value

以下代码示例演示了如何使用 put-secret-value。

Amazon CLI

示例 1：在密钥中存储新的密钥值

以下 put-secret-value 示例将创建包含两个键值对的新版本密钥。

```
aws secretsmanager put-secret-value \
    --secret-id MyTestSecret \
    --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-1a2b3c",
    "Name": "MyTestSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "VersionStages": [
        "AWSCURRENT"
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的修改密钥。

示例 2：将凭证中的新密钥值存储在 JSON 文件中

以下 put-secret-value 示例将根据文件中的凭证创建新版本密钥。有关更多信息，请参阅《Amazon CLI 用户指南》中的从文件加载 Amazon CLI 参数。

```
aws secretsmanager put-secret-value \
    --secret-id MyTestSecret \
    --secret-string file://mycreds.json
```

的内容mycreds.json：

```
{
  "engine": "mysql",
```

```
  "username": "saanvis",
  "password": "EXAMPLE-PASSWORD",
  "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",
  "dbname": "myDatabase",
  "port": "3306"
}
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "VersionStages": [
        "AWSCURRENT"
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的修改密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutSecretValue。

`remove-regions-from-replication`

以下代码示例演示了如何使用 `remove-regions-from-replication`。

Amazon CLI

删除副本密钥

以下 `remove-regions-from-replication` 示例将删除 eu-west-3 中的副本密钥。要删除复制到其他区域的主密钥，请先删除副本，然后调用 `delete-secret`。

```
aws secretsmanager remove-regions-from-replication \
    --secret-id MyTestSecret \
    --remove-replica-regions eu-west-3
```

输出：

```
{
```

```
    "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-1a2b3c",
    "ReplicationStatus": []
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的删除副本密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveRegionsFromReplication。

replicate-secret-to-regions

以下代码示例演示了如何使用 replicate-secret-to-regions。

Amazon CLI

将密钥复制到其他区域

以下 replicate-secret-to-regions 示例将密钥复制到 eu-west-3。副本使用 Amazon 托管密钥 aws/secretsmanager 加密。

```
aws secretsmanager replicate-secret-to-regions \
    --secret-id MyTestSecret \
    --add-replica-regions Region=eu-west-3
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-1a2b3c",
    "ReplicationStatus": [
        {
            "Region": "eu-west-3",
            "KmsKeyId": "alias/aws/secretsmanager",
            "Status": "InProgress"
        }
    ]
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的将密钥复制到其他区域。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReplicateSecretToRegions。

restore-secret

以下代码示例演示了如何使用 restore-secret。

Amazon CLI

恢复先前删除的密钥

以下 restore-secret 示例恢复了先前计划删除的密钥。

```
aws secretsmanager restore-secret \
    --secret-id MyTestSecret
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的删除密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreSecret。

rotate-secret

以下代码示例演示了如何使用 rotate-secret。

Amazon CLI

示例 1：配置并启动密钥的自动轮换功能

以下 rotate-secret 示例配置并启动密钥的自动轮换功能。Secrets Manager 会立即轮换一次密钥，然后每八小时轮换一次，轮换时间为两小时。输出显示了通过轮换创建的新密钥版本的 VersionId。

```
aws secretsmanager rotate-secret \
    --secret-id MyTestDatabaseSecret \
    --rotation-lambda-arn arn:aws:lambda:us-
west-2:1234566789012:function:SecretsManagerTestRotationLambda \
```

```
    --rotation-rules "{\"ScheduleExpression\": \"cron(0 8/8 * * ? *)\", \"Duration
\": \"2h\"}"
```

输出：

```
{
    "ARN": "aws:arn:secretsmanager:us-
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",
    "Name": "MyTestDatabaseSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的轮换密钥。

示例 2：配置并启动按轮换间隔执行的自动轮换功能

以下 `rotate-secret` 示例配置并启动密钥的自动轮换功能。Secrets Manager 会立即轮换一次密钥，然后每 10 天轮换一次。输出显示了通过轮换创建的新密钥版本的 `VersionId`。

```
aws secretsmanager rotate-secret \
    --secret-id MyTestDatabaseSecret \
    --rotation-lambda-arn arn:aws:lambda:us-
west-2:1234566789012:function:SecretsManagerTestRotationLambda \
    --rotation-rules "{\"ScheduleExpression\": \"rate(10 days)\"}"
```

输出：

```
{
    "ARN": "aws:arn:secretsmanager:us-
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",
    "Name": "MyTestDatabaseSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的轮换密钥。

示例 3：立即轮换密钥

以下 `rotate-secret` 示例将立即开始轮换。输出显示了通过轮换创建的新密钥版本的 `VersionId`。密钥必须已配置轮换。

```
aws secretsmanager rotate-secret \
    --secret-id MyTestDatabaseSecret
```

输出：

```
{
    "ARN": "aws:arn:secretsmanager:us-
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",
    "Name": "MyTestDatabaseSecret",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的轮换密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RotateSecret。

stop-replication-to-replica

以下代码示例演示了如何使用 stop-replication-to-replica。

Amazon CLI

将副本密钥提升为主密钥

以下 stop-replication-to-replica 示例将删除副本密钥与主密钥之间的链接。副本密钥在副本区域中被提升为主密钥。您必须从副本区域内调用 stop-replication-to-replica。

```
aws secretsmanager stop-replication-to-replica \
    --secret-id MyTestSecret
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的提升副本密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopReplicationToReplica。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

示例 1：将标签添加到密钥

以下 示例说明了如何使用速记语法附加标签。

```
aws secretsmanager tag-resource \
    --secret-id MyTestSecret \
    --tags Key=FirstTag,Value=FirstValue
```

此命令不生成任何输出。

有关更多信息，请参阅《Secrets Manager 用户指南》中的标记密钥。

示例 2：将多个标签添加到密钥

以下 tag-resource 示例将向密钥附加两个键值标签。

```
aws secretsmanager tag-resource \
    --secret-id MyTestSecret \
    --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag",
 "Value": "SecondValue"}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Secrets Manager 用户指南》中的标记密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从密钥中移除标签

以下 untag-resource 示例将从密钥中删除两个标签。对于每个标签，键和值都会被删除。

```
aws secretsmanager untag-resource \
    --secret-id MyTestSecret \
    --tag-keys '[ "FirstTag", "SecondTag"]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Secrets Manager 用户指南》中的标记密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-secret-version-stage

以下代码示例演示了如何使用 update-secret-version-stage。

Amazon CLI

示例 1：将密钥还原为先前版本

以下 update-secret-version-stage 示例会将 AmazonCURRENT 暂存标签移动到密钥的先前版本，这样做会将密钥还原为先前的版本。要查找先前版本的 ID，请使用 list-secret-version-ids。对于此示例，带有 AmazonCURRENT 标签的版本为 a1b2c3d4-5678-90ab-cdef-EXAMPLE11111，带有 AmazonPREVIOUS 标签的版本为 a1b2c3d4-5678-90ab-cdef-EXAMPLE22222。在此示例中，您将 AmazonCURRENT 标签从版本 11111 移动到 22222。由于 AmazonCURRENT 标签已从版本中移除，因此 update-secret-version-stage 会自动将 AmazonPREVIOUS 标签移动到该版本（11111）。结果是交换了 AmazonCURRENT 和 AmazonPREVIOUS 版本。

```
aws secretsmanager update-secret-version-stage \
    --secret-id MyTestSecret \
    --version-stage AWSCURRENT \
    --move-to-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
    --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的版本。

示例 2：添加附加到密钥版本的暂存标签

以下 update-secret-version-stage 示例为密钥版本添加暂存标签。您可以通过运行 list-secret-version-ids 并查看受影响版本的响应字段 VersionStages 来查看结果。

```
aws secretsmanager update-secret-version-stage \
    --secret-id MyTestSecret \
    --version-stage STAGINGLABEL1 \
    --move-to-version-id EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的版本。

示例 3：删除附加到密钥版本的暂存标签

以下 update-secret-version-stage 示例删除附加在密钥版本上的暂存标签。您可以通过运行 list-secret-version-ids 并查看受影响版本的响应字段 VersionStages 来查看结果。

```
aws secretsmanager update-secret-version-stage \
    --secret-id MyTestSecret \
    --version-stage STAGINGLABEL1 \
    --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的版本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecretVersionStage。

update-secret

以下代码示例演示了如何使用 update-secret。

Amazon CLI

示例 1：更新密钥的描述

以下 update-secret 示例将更新密钥的描述。

```
aws secretsmanager update-secret \
    --secret-id MyTestSecret \
    --description "This is a new description for the secret."
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的修改密钥。

示例 2：更新与密钥关联的加密密钥

以下 update-secret 示例将更新用于加密密钥值的 KMS 密钥。该 KMS 密钥必须与加密密钥位于同一区域中。

```
aws secretsmanager update-secret \
    --secret-id MyTestSecret \
    --kms-key-id arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-
ba987EXAMPLE
```

输出：

```
{
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
```

```
    "Name": "MyTestSecret"
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的修改密钥。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecret。

validate-resource-policy

以下代码示例演示了如何使用 validate-resource-policy。

Amazon CLI

验证资源策略

以下 validate-resource-policy 示例检查资源策略是否未授予对密钥的广泛访问权限。该策略是从磁盘上的文件中读取的。有关更多信息，请参阅《Amazon CLI 用户指南》中的从文件加载 Amazon CLI 参数。

```
aws secretsmanager validate-resource-policy \
    --resource-policy file://mypolicy.json
```

的内容mypolicy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

输出：

```
{
```

```
    "PolicyValidationPassed": true,
    "ValidationErrors": []
}
```

有关更多信息，请参阅《Secrets Manager 用户指南》中的 Secrets Manager 权限参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ValidateResourcePolicy。

使用 Amazon CLI 的 Security Hub CSPM 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Security Hub CSPM 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-administrator-invitation`

以下代码示例演示了如何使用 `accept-administrator-invitation`。

Amazon CLI

接受管理员账户的邀请

以下 `accept-administrator-invitation` 示例接受来自指定管理员账户的指定邀请。

```
aws securityhub accept-invitation \
    --administrator-id 123456789012 \
    --invitation-id 7ab938c5d52d7904ad09f9e7c20cc4eb
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptAdministratorInvitation。

accept-invitation

以下代码示例演示了如何使用 accept-invitation。

Amazon CLI

接受管理员账户的邀请

以下 accept-invitation 示例接受来自指定管理员账户的指定邀请。

```
aws securityhub accept-invitation \
    --master-id 123456789012 \
    --invitation-id 7ab938c5d52d7904ad09f9e7c20cc4eb
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptInvitation。

batch-delete-automation-rules

以下代码示例演示了如何使用 batch-delete-automation-rules。

Amazon CLI

删除自动化规则

以下 batch-delete-automation-rules 示例删除指定的自动化规则。您只需一个命令即可删除一个或多个规则。只有 Security Hub 管理员账户才能运行此命令。

```
aws securityhub batch-delete-automation-rules \
    --automation-rules-arns '["arn:aws:securityhub:us-
east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"]'
```

输出：

```
{
    "ProcessedAutomationRules": [
```

```
        "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    ],
    "UnprocessedAutomationRules": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的删除自动化规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDeleteAutomationRules。

batch-disable-standards

以下代码示例演示了如何使用 batch-disable-standards。

Amazon CLI

禁用标准

以下 batch-disable-standards 示例禁用与指定订阅 ARN 相关的标准。

```
aws securityhub batch-disable-standards \
    --standards-subscription-arns "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
```

输出：

```
{
    "StandardsSubscriptions": [
        {
            "StandardsArn": "arn:aws:securityhub:eu-central-1::standards/pci-dss/
v/3.2.1",
            "StandardsInput": { },
            "StandardsStatus": "DELETING",
            "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的禁用或启用安全标准。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchDisableStandards。

batch-enable-standards

以下代码示例演示了如何使用 batch-enable-standards。

Amazon CLI

启用标准

以下 batch-enable-standards 示例为请求的账户启用 PCI DSS 标准。

```
aws securityhub batch-enable-standards \
    --standards-subscription-requests '{"StandardsArn":"arn:aws:securityhub:us-
west-1::standards/pci-dss/v/3.2.1"}'
```

输出：

```
{
    "StandardsSubscriptions": [
        {
            "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/
v/3.2.1",
            "StandardsInput": { },
            "StandardsStatus": "PENDING",
            "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的禁用或启用安全标准。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchEnableStandards。

batch-get-automation-rules

以下代码示例演示了如何使用 batch-get-automation-rules。

Amazon CLI

获取自动化规则的详细信息

以下 batch-get-automation-rules 示例获取指定自动化规则的详细信息。您只需一个命令即可获得一个或多个自动化规则的详细信息。

```
aws securityhub batch-get-automation-rules \
    --automation-rules-arns '["arn:aws:securityhub:us-
east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"]'
```

输出：

```
{
    "Rules": [
        {
            "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "RuleStatus": "ENABLED",
            "RuleOrder": 1,
            "RuleName": "Suppress informational findings",
            "Description": "Suppress GuardDuty findings with Informational
 severity",
            "IsTerminal": false,
            "Criteria": {
                "ProductName": [
                    {
                        "Value": "GuardDuty",
                        "Comparison": "EQUALS"
                    }
                ],
                "SeverityLabel": [
                    {
                        "Value": "INFORMATIONAL",
                        "Comparison": "EQUALS"
                    }
                ],
                "WorkflowStatus": [
                    {
                        "Value": "NEW",
                        "Comparison": "EQUALS"
                    }
                ],
                "RecordState": [
                    {
                        "Value": "ACTIVE",
                        "Comparison": "EQUALS"
                    }
                ]
            },
```

```
            "Actions": [
                {
                    "Type": "FINDING_FIELDS_UPDATE",
                    "FindingFieldsUpdate": {
                        "Note": {
                            "Text": "Automatically suppress GuardDuty findings with
 Informational severity",
                            "UpdatedBy": "sechub-automation"
                        },
                        "Workflow": {
                            "Status": "SUPPRESSED"
                        }
                    }
                }
            ],
            "CreatedAt": "2023-05-31T17:56:14.837000+00:00",
            "UpdatedAt": "2023-05-31T17:59:38.466000+00:00",
            "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
        }
    ],
    "UnprocessedAutomationRules": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看自动化规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetAutomationRules。

batch-get-configuration-policy-associations

以下代码示例演示了如何使用 batch-get-configuration-policy-associations。

Amazon CLI

获取一批目标的配置关联详细信息

以下 batch-get-configuration-policy-associations 示例检索指定目标的关联详细信息。您可以提供目标的账户 ID、组织单位 ID 或根 ID。

```
aws securityhub batch-get-configuration-policy-associations \
    --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

输出：

```
{
    "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "TargetId": "ou-6hi7-8j91kl2m",
    "TargetType": "ORGANIZATIONAL_UNIT",
    "AssociationType": "APPLIED",
    "UpdatedAt": "2023-09-26T21:13:01.816000+00:00",
    "AssociationStatus": "SUCCESS",
    "AssociationStatusMessage": "Association applied successfully on this target."
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看 Security Hub 配置策略。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetConfigurationPolicyAssociations。

batch-get-security-controls

以下代码示例演示了如何使用 batch-get-security-controls。

Amazon CLI

获取安全控件详细信息

以下 batch-get-security-controls 示例获取当前 Amazon 账户和 Amazon 区域中安全控件 ACM.1 和 IAM.1 的详细信息。

```
aws securityhub batch-get-security-controls \
    --security-control-ids '["ACM.1", "IAM.1"]'
```

输出：

```
{
    "SecurityControls": [
        {
            "SecurityControlId": "ACM.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/ACM.1",
            "Title": "Imported and ACM-issued certificates should be renewed after a
 specified time period",
            "Description": "This control checks whether an AWS Certificate Manager
 (ACM) certificate is renewed within the specified time period. It checks both
```

```
 imported certificates and certificates provided by ACM. The control fails if the
 certificate isn't renewed within the specified time period. Unless you provide a
 custom parameter value for the renewal period, Security Hub uses a default value of
 30 days.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
ACM.1/remediation",
            "SeverityRating": "MEDIUM",
            "SecurityControlStatus": "ENABLED"
            "UpdateStatus": "READY",
            "Parameters": {
                "daysToExpiration": {
                    "ValueType": CUSTOM,
                    "Value": {
                        "Integer": 15
                    }
                }
            },
            "LastUpdateReason": "Updated control parameter"
        },
        {
            "SecurityControlId": "IAM.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/IAM.1",
            "Title": "IAM policies should not allow full \"*\" administrative
 privileges",
            "Description": "This AWS control checks whether the default version of
 AWS Identity and Access Management (IAM) policies (also known as customer managed
 policies) do not have administrator access with a statement that has \"Effect\":
 \"Allow\" with \"Action\": \"*\" over \"Resource\": \"*\". It only checks for
 the Customer Managed Policies that you created, but not inline and AWS Managed
 Policies.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
IAM.1/remediation",
            "SeverityRating": "HIGH",
            "SecurityControlStatus": "ENABLED"
            "UpdateStatus": "READY",
            "Parameters": {}
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看控件的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetSecurityControls。

batch-get-standards-control-associations

以下代码示例演示了如何使用 batch-get-standards-control-associations。

Amazon CLI

获取控件的启用状态

以下 batch-get-standards-control-associations 示例确定指定标准中是否启用了指定控件。

```
aws securityhub batch-get-standards-control-associations \
    --standards-control-association-ids '[{"SecurityControlId":
 "Config.1","StandardsArn": "arn:aws:securityhub:us-east-1:123456789012:ruleset/cis-
aws-foundations-benchmark/v/1.2.0"}, {"SecurityControlId": "IAM.6","StandardsArn":
 "arn:aws:securityhub:us-east-1:123456789012:standards/aws-foundational-security-
best-practices/v/1.0.0"}]'
```

输出：

```
{
    "StandardsControlAssociationDetails": [
        {
            "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0",
            "SecurityControlId": "Config.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-1:068873283051:security-control/Config.1",
            "AssociationStatus": "ENABLED",
            "RelatedRequirements": [
                "CIS AWS Foundations 2.5"
            ],
            "UpdatedAt": "2022-10-27T16:07:12.960000+00:00",
            "StandardsControlTitle": "Ensure AWS Config is enabled",
            "StandardsControlDescription": "AWS Config is a web service that
 performs configuration management of supported AWS resources within your account
 and delivers log files to you. The recorded information includes the configuration
 item (AWS resource), relationships between configuration items (AWS resources), and
 any configuration changes between resources. It is recommended to enable AWS Config
 in all regions.",
            "StandardsControlArns": [
                "arn:aws:securityhub:us-east-1:068873283051:control/cis-aws-
foundations-benchmark/v/1.2.0/2.5"
```

```
            ]
        },
        {
            "StandardsArn": "arn:aws:securityhub:us-east-1::standards/aws-
foundational-security-best-practices/v/1.0.0",
            "SecurityControlId": "IAM.6",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-1:068873283051:security-control/IAM.6",
            "AssociationStatus": "DISABLED",
            "RelatedRequirements": [],
            "UpdatedAt": "2022-11-22T21:30:35.080000+00:00",
            "UpdatedReason": "test",
            "StandardsControlTitle": "Hardware MFA should be enabled for the root
 user",
            "StandardsControlDescription": "This AWS control checks whether your AWS
 account is enabled to use a hardware multi-factor authentication (MFA) device to
 sign in with root user credentials.",
            "StandardsControlArns": [
                "arn:aws:securityhub:us-east-1:068873283051:control/aws-
foundational-security-best-practices/v/1.0.0/IAM.6"
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的在指定标准中启用或禁用控件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetStandardsControlAssociations。

batch-import-findings

以下代码示例演示了如何使用 batch-import-findings。

Amazon CLI

更新调查发现

以下 batch-import-findings 示例更新调查发现。

```
aws securityhub batch-import-findings \
     --findings '
        [{
```

```
            "AwsAccountId": "123456789012",
            "CreatedAt": "2020-05-27T17:05:54.832Z",
            "Description": "Vulnerability in a CloudTrail trail",
            "FindingProviderFields": {
                "Severity": {
                    "Label": "LOW",
                    "Original": "10"
                },
                "Types": [
                    "Software and Configuration Checks/Vulnerabilities/CVE"
                ]
            },
            "GeneratorId": "TestGeneratorId",
            "Id": "Id1",
            "ProductArn": "arn:aws:securityhub:us-
west-1:123456789012:product/123456789012/default",
            "Resources": [
                {
                    "Id": "arn:aws:cloudtrail:us-west-1:123456789012:trail/
TrailName",
                    "Partition": "aws",
                    "Region": "us-west-1",
                    "Type": "AwsCloudTrailTrail"
                }
            ],
            "SchemaVersion": "2018-10-08",
            "Title": "CloudTrail trail vulnerability",
            "UpdatedAt": "2020-06-02T16:05:54.832Z"
        }]'
```

输出：

```
{
    "FailedCount": 0,
    "SuccessCount": 1,
    "FailedFindings": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的使用 BatchImportFindings 创建和更新调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchImportFindings。

batch-update-automation-rules

以下代码示例演示了如何使用 batch-update-automation-rules。

Amazon CLI

更新自动化规则

以下 batch-update-automation-rules 示例更新指定的自动化规则。您只需一个命令即可更新一个或多个规则。只有 Security Hub 管理员账户才能运行此命令。

```
aws securityhub batch-update-automation-rules \
    --update-automation-rules-request-items '[ \
        { \
            "Actions": [{ \
                "Type": "FINDING_FIELDS_UPDATE", \
                "FindingFieldsUpdate": { \
                    "Note": { \
                        "Text": "Known issue that is a risk", \
                        "UpdatedBy": "sechub-automation" \
                    }, \
                    "Workflow": { \
                        "Status": "NEW" \
                    } \
                } \
            }], \
            "Criteria": { \
                "SeverityLabel": [{ \
                    "Value": "LOW", \
                    "Comparison": "EQUALS" \
                }] \
            }, \
            "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111", \
            "RuleOrder": 1, \
            "RuleStatus": "DISABLED" \
        } \
    ]'
```

输出：

```
{
    "ProcessedAutomationRules": [
```

```
        "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    ],
    "UnprocessedAutomationRules": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的编辑自动化规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateAutomationRules。

batch-update-findings

以下代码示例演示了如何使用 batch-update-findings。

Amazon CLI

示例 1：更新调查发现

以下 batch-update-findings 示例更新两个调查发现，以添加注释、更改严重性标签并解决这些问题。

```
aws securityhub batch-update-findings \
    --finding-identifiers '[{"Id": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111", "ProductArn": "arn:aws:securityhub:us-
west-1::product/aws/securityhub"}, {"Id": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222", "ProductArn": "arn:aws:securityhub:us-
west-1::product/aws/securityhub"}]' \
    --note '{"Text": "Known issue that is not a risk.", "UpdatedBy": "user1"}' \
    --severity '{"Label": "LOW"}' \
    --workflow '{"Status": "RESOLVED"}'
```

输出：

```
{
    "ProcessedFindings": [
        {
            "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/
v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
        },
```

```
        {
            "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/
v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
        }
    ],
    "UnprocessedFindings": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的使用 BatchUpdateFindings 更新调查发现。

示例 2：使用速记语法更新调查发现

以下 batch-update-findings 示例更新两个调查发现，以使用速记语法添加注释、更改严重性标签并解决这些问题。

```
aws securityhub batch-update-findings \
    --finding-identifiers Id="arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",ProductArn="arn:aws:securityhub:us-
west-1::product/aws/securityhub" Id="arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",ProductArn="arn:aws:securityhub:us-
west-1::product/aws/securityhub" \
    --note Text="Known issue that is not a risk.",UpdatedBy="user1" \
    --severity Label="LOW" \
    --workflow Status="RESOLVED"
```

输出：

```
{
    "ProcessedFindings": [
        {
            "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/
v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
        },
        {
            "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/
v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
```

```
        }
    ],
    "UnprocessedFindings": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的使用 BatchUpdateFindings 更新调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateFindings。

batch-update-standards-control-associations

以下代码示例演示了如何使用 batch-update-standards-control-associations。

Amazon CLI

更新已启用标准中控件的启用状态

以下 batch-update-standards-control-associations 示例在指定标准中禁用 CloudTrail.1。

```
aws securityhub batch-update-standards-control-associations \
    --standards-control-association-updates '[{"SecurityControlId": "CloudTrail.1",
 "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0", "AssociationStatus": "DISABLED", "UpdatedReason": "Not applicable
 to environment"}, {"SecurityControlId": "CloudTrail.1", "StandardsArn":
 "arn:aws:securityhub:::standards/cis-aws-foundations-benchmark/v/1.4.0",
 "AssociationStatus": "DISABLED", "UpdatedReason": "Not applicable to
 environment"}]'
```

如果成功，此命令不会产生任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的在指定标准中启用或禁用控件和在所有标准中启用和禁用控件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchUpdateStandardsControlAssociations。

create-action-target

以下代码示例演示了如何使用 create-action-target。

Amazon CLI

创建自定义操作

以下 create-action-target 示例创建一个自定义操作。它将提供操作的名称、描述和标识符。

```
aws securityhub create-action-target \
    --name "Send to remediation" \
    --description "Action to send the finding for remediation tracking" \
    --id "Remediation"
```

输出：

```
{
    "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/custom/
Remediation"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建自定义操作并将其与 CloudWatch 事件规则关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateActionTarget。

create-automation-rule

以下代码示例演示了如何使用 create-automation-rule。

Amazon CLI

创建自动化规则

以下 create-automation-rule 示例在当前 Amazon 账户和 Amazon 区域中创建一个自动化规则。Security Hub 会根据指定的条件筛选您的调查发现，并将操作应用于匹配的调查发现。只有 Security Hub 管理员账户才能运行此命令。

```
aws securityhub create-automation-rule \
    --actions '[{ \
        "Type": "FINDING_FIELDS_UPDATE", \
        "FindingFieldsUpdate": { \
            "Severity": { \
```

```
                "Label": "HIGH" \
            }, \
            "Note": { \
                "Text": "Known issue that is a risk. Updated by automation rules", \
                "UpdatedBy": "sechub-automation" \
            } \
        } \
    }]' \
    --criteria '{ \
        "SeverityLabel": [{ \
            "Value": "INFORMATIONAL", \
            "Comparison": "EQUALS" \
        }] \
    }' \
    --description "A sample rule" \
    --no-is-terminal \
    --rule-name "sample rule" \
    --rule-order 1 \
    --rule-status "ENABLED"
```

输出：

```
{
    "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建自动化规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAutomationRule。

create-configuration-policy

以下代码示例演示了如何使用 create-configuration-policy。

Amazon CLI

创建配置策略

以下 create-configuration-policy 示例使用指定设置创建一个配置策略。

```
aws securityhub create-configuration-policy \
    --name "SampleConfigurationPolicy" \
```

```
    --description "SampleDescription" \
    --configuration-policy '{"SecurityHub": {"ServiceEnabled":
 true, "EnabledStandardIdentifiers": ["arn:aws:securityhub:eu-
central-1::standards/aws-foundational-security-best-practices/
v/1.0.0","arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"],"SecurityControlsConfiguration":{"DisabledSecurityControlIdentifiers":
 ["CloudTrail.2"], "SecurityControlCustomParameters": [{"SecurityControlId":
 "ACM.1", "Parameters": {"daysToExpiration": {"ValueType": "CUSTOM", "Value":
 {"Integer": 15}}}}]}}}' \
    --tags '{"Environment": "Prod"}'
```

输出：

```
{
    "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Name": "SampleConfigurationPolicy",
    "Description": "SampleDescription",
    "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
    "CreatedAt": "2023-11-28T20:28:04.494000+00:00",
    "ConfigurationPolicy": {
        "SecurityHub": {
            "ServiceEnabled": true,
            "EnabledStandardIdentifiers": [
                "arn:aws:securityhub:eu-central-1::standards/aws-foundational-
security-best-practices/v/1.0.0",
                "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"
            ],
            "SecurityControlsConfiguration": {
                "DisabledSecurityControlIdentifiers": [
                    "CloudTrail.2"
                ],
                "SecurityControlCustomParameters": [
                    {
                        "SecurityControlId": "ACM.1",
                        "Parameters": {
                            "daysToExpiration": {
                                "ValueType": "CUSTOM",
                                "Value": {
                                    "Integer": 15
                                }
```

```
                            }
                        }
                    }
                ]
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建并关联 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateConfigurationPolicy。

`create-finding-aggregator`

以下代码示例演示了如何使用 `create-finding-aggregator`。

Amazon CLI

启用调查发现聚合

以下 `create-finding-aggregator` 示例配置调查发现聚合。它会从美国东部（弗吉尼亚）运行，指定美国东部（弗吉尼亚）为聚合区域。它表示仅链接指定的区域，不自动链接新区域。它选择美国西部（北加利福尼亚）和美国西部（俄勒冈州）作为链接区域。

```
aws securityhub create-finding-aggregator \
    --region us-east-1 \
    --region-linking-mode SPECIFIED_REGIONS \
    --regions us-west-1,us-west-2
```

输出：

```
{
    "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000",
    "FindingAggregationRegion": "us-east-1",
    "RegionLinkingMode": "SPECIFIED_REGIONS",
    "Regions": "us-west-1,us-west-2"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的启用调查发现聚合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFindingAggregator。

create-insight

以下代码示例演示了如何使用 create-insight。

Amazon CLI

创建自定义见解

以下 create-insight 示例创建一个名为“关键角色调查发现”的自定义见解，将返回与 Amazon 角色相关的关键调查发现。

```
aws securityhub create-insight \
    --filters '{"ResourceType": [{ "Comparison": "EQUALS", "Value": "AwsIamRole"}],
 "SeverityLabel": [{"Comparison": "EQUALS", "Value": "CRITICAL"}]}' \
    --group-by-attribute "ResourceId" \
    --name "Critical role findings"
```

输出：

```
{
    "InsightArn": "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理自定义见解。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateInsight。

create-members

以下代码示例演示了如何使用 create-members。

Amazon CLI

将账户添加为成员账户

以下 create-members 示例会将两个账户作为成员账户添加到请求的管理员账户。

```
aws securityhub create-members \
```

```
    --account-details '[{"AccountId": "123456789111"}, {"AccountId":
 "123456789222"}]'
```

输出：

```
{
    "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMembers。

decline-invitations

以下代码示例演示了如何使用 decline-invitations。

Amazon CLI

拒绝成为成员账户的邀请

以下 decline-invitations 示例拒绝成为指定管理员账户成员账户的邀请。成员账户是请求的账户。

```
aws securityhub decline-invitations \
    --account-ids "123456789012"
```

输出：

```
{
    "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeclineInvitations。

delete-action-target

以下代码示例演示了如何使用 delete-action-target。

Amazon CLI

删除自定义操作

以下 delete-action-target 示例删除由指定 ARN 标识的自定义操作。

```
aws securityhub delete-action-target \
    --action-target-arn "arn:aws:securityhub:us-west-1:123456789012:action/custom/Remediation"
```

输出：

```
{
    "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/custom/Remediation"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建自定义操作并将其与 CloudWatch 事件规则关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteActionTarget。

delete-configuration-policy

以下代码示例演示了如何使用 delete-configuration-policy。

Amazon CLI

要删除配置策略

以下 delete-configuration-policy 示例删除指定的配置策略。

```
aws securityhub delete-configuration-policy \
    --identifier "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的删除并解除与 Security Hub 配置策略的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteConfigurationPolicy。

delete-finding-aggregator

以下代码示例演示了如何使用 delete-finding-aggregator。

Amazon CLI

停止调查发现聚合

以下 delete-finding-aggregator 示例停止调查发现聚合。它从聚合区域（即美国东部（弗吉尼亚））运行。

```
aws securityhub delete-finding-aggregator \
    --region us-east-1 \
    --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-aggregator/123e4567-e89b-12d3-a456-426652340000
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的停止调查发现聚合。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFindingAggregator。

delete-insight

以下代码示例演示了如何使用 delete-insight。

Amazon CLI

删除自定义见解

以下 delete-insight 示例删除指定 ARN 的自定义见解。

```
aws securityhub delete-insight \
    --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

输出：

```
{
   "InsightArn": "arn:aws:securityhub:eu-central-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

```
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理自定义见解。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInsight。

delete-invitations

以下代码示例演示了如何使用 delete-invitations。

Amazon CLI

删除成为成员账户的邀请

以下 delete-invitations 示例删除指定管理员账户的成员账户邀请。成员账户是请求的账户。

```
aws securityhub delete-invitations \
    --account-ids "123456789012"
```

输出：

```
{
    "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInvitations。

delete-members

以下代码示例演示了如何使用 delete-members。

Amazon CLI

删除成员账户

以下 delete-members 示例从请求的管理员帐户中删除指定的成员帐户。

```
aws securityhub delete-members \
```

```
    --account-ids "123456789111" "123456789222"
```

输出：

```
{
    "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMembers。

describe-action-targets

以下代码示例演示了如何使用 describe-action-targets。

Amazon CLI

检索有关自定义操作的详细信息

以下 describe-action-targets 示例检索有关由指定 ARN 标识的自定义操作的信息。

```
aws securityhub describe-action-targets \
    --action-target-arns "arn:aws:securityhub:us-west-1:123456789012:action/custom/
Remediation"
```

输出：

```
{
    "ActionTargets": [
        {
            "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/
custom/Remediation",
            "Description": "Action to send the finding for remediation tracking",
            "Name": "Send to remediation"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建自定义操作并将其与 CloudWatch 事件规则关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeActionTargets。

describe-hub

以下代码示例演示了如何使用 describe-hub。

Amazon CLI

获取有关 Hub 资源的信息

以下 describe-hub 示例返回指定 Hub 资源的订阅日期和其他配置设置。Hub 资源由其 ARN 标识。

```
aws securityhub describe-hub \
    --hub-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default"
```

输出：

```
{
    "HubArn": "arn:aws:securityhub:us-west-1:123456789012:hub/default",
    "SubscribedAt": "2019-11-19T23:15:10.046Z",
    "AutoEnableControls": true,
    "ControlFindingGenerator": "SECURITY_CONTROL"
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 Amazon::SecurityHub::Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeHub。

describe-organization-configuration

以下代码示例演示了如何使用 describe-organization-configuration。

Amazon CLI

查看如何为组织配置 Security Hub

以下 describe-organization-configuration 示例返回有关组织在 Security Hub 中配置方式的信息。在此示例中，组织使用中央配置。只有 Security Hub 管理员账户才能运行此命令。

```
aws securityhub describe-organization-configuration
```

输出：

```
{
    "AutoEnable": false,
    "MemberAccountLimitReached": false,
    "AutoEnableStandards": "NONE",
    "OrganizationConfiguration": {
        "ConfigurationType": "LOCAL",
        "Status": "ENABLED",
        "StatusMessage": "Central configuration has been enabled successfully"
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的使用 Amazon Organizations 管理账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrganizationConfiguration。

describe-products

以下代码示例演示了如何使用 describe-products。

Amazon CLI

返回有关可用产品集成的信息

以下 describe-products 示例逐一返回可用的产品集成。

```
aws securityhub describe-products \
    --max-results 1
```

输出：

```
{
    "NextToken": "U2FsdGVkX18vvPlOqb7RDrWRWVFBJI46MOIAb+nZmRJmR15NoRi2gm13sdQEn3O/
pq/78dGs+bKpgA+7HMPHO0qX33/zoRI+uIG/F9yLNhcOrOWzFUdy36JcXLQji3Rpnn/
cD1SVkGA98qI3zPOSDg==",
    "Products": [
        {
            "ProductArn": "arn:aws:securityhub:us-west-1:123456789333:product/
crowdstrike/crowdstrike-falcon",
```

```
            "ProductName": "CrowdStrike Falcon",
            "CompanyName": "CrowdStrike",
            "Description": "CrowdStrike Falcon's single lightweight sensor unifies
 next-gen antivirus, endpoint detection and response, and 24/7 managed hunting, via
 the cloud.",
            "Categories": [
                "Endpoint Detection and Response (EDR)",
                "AV Scanning and Sandboxing",
                "Threat Intelligence Feeds and Reports",
                "Endpoint Forensics",
                "Network Forensics"
            ],
            "IntegrationTypes": [
                "SEND_FINDINGS_TO_SECURITY_HUB"
            ],
            "MarketplaceUrl": "https://aws.amazon.com/marketplace/seller-profile?
id=a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ActivationUrl": "https://falcon.crowdstrike.com/support/documentation",
            "ProductSubscriptionResourcePolicy": "{\"Version\":
\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":
\"123456789333\"},\"Action\":[\"securityhub:BatchImportFindings\"],\"Resource\":
\"arn:aws:securityhub:us-west-1:123456789012:product-subscription/crowdstrike/
crowdstrike-falcon\",\"Condition\":{\"StringEquals\":{\"securityhub:TargetAccount
\":\"123456789012\"}}},{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":
\"123456789012\"},\"Action\":[\"securityhub:BatchImportFindings\"],\"Resource
\":\"arn:aws:securityhub:us-west-1:123456789333:product/crowdstrike/crowdstrike-
falcon\",\"Condition\":{\"StringEquals\":{\"securityhub:TargetAccount\":
\"123456789012\"}}}]}"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理产品集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProducts。

`describe-standards-controls`

以下代码示例演示了如何使用 `describe-standards-controls`。

Amazon CLI

请求已启用标准中的控件列表

以下 describe-standards-controls 示例请求请求者账户订阅的 PCI DSS 标准中的控件列表。该请求一次返回两个控件。

```
aws securityhub describe-standards-controls \
    --standards-subscription-arn "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1" \
    --max-results 2
```

输出：

```
{
    "Controls": [
        {
            "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.AutoScaling.1",
            "ControlStatus": "ENABLED",
            "ControlStatusUpdatedAt": "2020-05-15T18:49:04.473000+00:00",
            "ControlId": "PCI.AutoScaling.1",
            "Title": "Auto scaling groups associated with a load balancer should use
 health checks",
            "Description": "This AWS control checks whether your Auto Scaling groups
 that are associated with a load balancer are using Elastic Load Balancing health
 checks.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
PCI.AutoScaling.1/remediation",
            "SeverityRating": "LOW",
            "RelatedRequirements": [
                "PCI DSS 2.2"
            ]
        },
        {
            "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.CW.1",
            "ControlStatus": "ENABLED",
            "ControlStatusUpdatedAt": "2020-05-15T18:49:04.498000+00:00",
            "ControlId": "PCI.CW.1",
            "Title": "A log metric filter and alarm should exist for usage of the
 \"root\" user",
            "Description": "This control checks for the CloudWatch metric
 filters using the following pattern { $.userIdentity.type = \"Root\" &&
 $.userIdentity.invokedBy NOT EXISTS && $.eventType != \"AwsServiceEvent\" }
 It checks that the log group name is configured for use with active multi-
region CloudTrail, that there is at least one Event Selector for a Trail with
```

```
 IncludeManagementEvents set to true and ReadWriteType set to All, and that there is
 at least one active subscriber to an SNS topic associated with the alarm.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
PCI.CW.1/remediation",
            "SeverityRating": "MEDIUM",
            "RelatedRequirements": [
                "PCI DSS 7.2.1"
            ]
        }
    ],
    "NextToken": "U2FsdGVkX1+eNkPoZHVl11ip5HUYQPWSWZGmftcmJiHL8JoKEsCDuaKayiPDyLK
+LiTkShveoOdvfxXCkOBaGhohIXhsIedN+LSjQV/
l7kfCfJcq4PziNC1N9xe9aq2pjlLVZnznTfSImrodT5bRNHe4fELCQq/z+5ka
+5Lzmc11axcwTd5lKgQyQqmUVoeriHZhyIiBgWKf7oNYdBVG8OEortVWvSkoUTt
+B2ThcnC7l43kI0UNxlkZ6sc64AsW"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看控件的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStandardsControls。

describe-standards

以下代码示例演示了如何使用 describe-standards。

Amazon CLI

返回可用标准列表

以下 describe-standards 示例返回可用标准的列表。

```
aws securityhub describe-standards
```

输出：

```
{
    "Standards": [
        {
            "StandardsArn": "arn:aws:securityhub:us-west-1::standards/aws-
foundational-security-best-practices/v/1.0.0",
            "Name": "AWS Foundational Security Best Practices v1.0.0",
            "Description": "The AWS Foundational Security Best Practices standard
 is a set of automated security checks that detect when AWS accounts and deployed
```

```
 resources do not align to security best practices. The standard is defined by AWS
 security experts. This curated set of controls helps improve your security posture
 in AWS, and cover AWS's most popular and foundational services.",
            "EnabledByDefault": true
        },
        {
            "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0",
            "Name": "CIS AWS Foundations Benchmark v1.2.0",
            "Description": "The Center for Internet Security (CIS) AWS Foundations
 Benchmark v1.2.0 is a set of security configuration best practices for AWS. This
 Security Hub standard automatically checks for your compliance readiness against a
 subset of CIS requirements.",
            "EnabledByDefault": true
        },
        {
            "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/
v/3.2.1",
            "Name": "PCI DSS v3.2.1",
            "Description": "The Payment Card Industry Data Security Standard (PCI
 DSS) v3.2.1 is an information security standard for entities that store, process,
 and/or transmit cardholder data. This Security Hub standard automatically checks
 for your compliance readiness against a subset of PCI DSS requirements.",
            "EnabledByDefault": false
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的 Amazon Security Hub 中的安全标准。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeStandards。

disable-import-findings-for-product

以下代码示例演示了如何使用 disable-import-findings-for-product。

Amazon CLI

停止接收来自产品集成的调查发现

以下 disable-import-findings-for-product 示例禁用指定订阅的产品集成的调查发现流。

```
aws securityhub disable-import-findings-for-product \
    --product-subscription-arn "arn:aws:securityhub:us-west-1:123456789012:product-subscription/crowdstrike/crowdstrike-falcon"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理产品集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableImportFindingsForProduct。

disable-organization-admin-account

以下代码示例演示了如何使用 disable-organization-admin-account。

Amazon CLI

移除 Security Hub 管理员账户

以下 disable-organization-admin-account 示例撤消指定账户作为 Amazon Organizations 的 Security Hub 管理员账户的分配。

```
aws securityhub disable-organization-admin-account \
    --admin-account-id 777788889999
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的指定 Security Hub 管理员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableOrganizationAdminAccount。

disable-security-hub

以下代码示例演示了如何使用 disable-security-hub。

Amazon CLI

禁用 Amazon Security Hub

以下 disable-security-hub 示例为请求账户禁用 Amazon Security Hub。

```
aws securityhub disable-security-hub
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的禁用 Amazon Security Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisableSecurityHub。

disassociate-from-administrator-account

以下代码示例演示了如何使用 disassociate-from-administrator-account。

Amazon CLI

从管理员账户解除关联

以下 disassociate-from-administrator-account 示例解除请求账户与其当前管理员账户的关联。

```
aws securityhub disassociate-from-administrator-account
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateFromAdministratorAccount。

disassociate-from-master-account

以下代码示例演示了如何使用 disassociate-from-master-account。

Amazon CLI

从管理员账户解除关联

以下 disassociate-from-master-account 示例解除请求账户与其当前管理员账户的关联。

```
aws securityhub disassociate-from-master-account
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateFromMasterAccount。

disassociate-members

以下代码示例演示了如何使用 disassociate-members。

Amazon CLI

取消成员账户的关联

以下 disassociate-members 示例从请求的管理员帐户中解除指定成员帐户的关联。

```
aws securityhub disassociate-members  \
    --account-ids "123456789111" "123456789222"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateMembers。

enable-import-findings-for-product

以下代码示例演示了如何使用 enable-import-findings-for-product。

Amazon CLI

开始接收来自产品集成的调查发现

以下 enable-import-findings-for-product 示例启用指定产品集成的调查发现流。

```
aws securityhub enable-import-findings-for-product \
    --product-arn "arn:aws:securityhub:us-east-1:123456789333:product/crowdstrike/
crowdstrike-falcon"
```

输出：

```
{
    "ProductSubscriptionArn": "arn:aws:securityhub:us-east-1:123456789012:product-
subscription/crowdstrike/crowdstrike-falcon"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理产品集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableImportFindingsForProduct。

enable-organization-admin-account

以下代码示例演示了如何使用 enable-organization-admin-account。

Amazon CLI

将组织账户指定为 Security Hub 管理员账户

以下 enable-organization-admin-account 示例会将指定账户指定为 Security Hub 管理员账户。

```
aws securityhub enable-organization-admin-account \
    --admin-account-id 777788889999
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的指定 Security Hub 管理员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableOrganizationAdminAccount。

enable-security-hub

以下代码示例演示了如何使用 enable-security-hub。

Amazon CLI

启用 Amazon Security Hub

以下 enable-security-hub 示例为请求账户启用 Amazon Security Hub。它会将 Security Hub 配置为启用默认标准。对于 Hub 资源，它会为标签 Department 分配值 Security。

```
aws securityhub enable-security-hub \
    --enable-default-standards \
    --tags '{"Department": "Security"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的启用 Security Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 EnableSecurityHub。

get-administrator-account

以下代码示例演示了如何使用 get-administrator-account。

Amazon CLI

检索管理员帐户的相关信息

以下 get-administrator-account 示例检索请求账户的管理员账户的相关信息。

```
aws securityhub get-administrator-account
```

输出：

```
{
   "Master": {
      "AccountId": "123456789012",
      "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
      "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
      "MemberStatus": "ASSOCIATED"
   }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAdministratorAccount。

get-configuration-policy-association

以下代码示例演示了如何使用 get-configuration-policy-association。

Amazon CLI

获取目标的配置关联详细信息

以下 get-configuration-policy-association 示例检索指定目标的关联详细信息。您可以提供目标的账户 ID、组织单位 ID 或根 ID。

```
aws securityhub get-configuration-policy-association \
    --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

输出：

```
{
    "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "TargetId": "ou-6hi7-8j91kl2m",
    "TargetType": "ORGANIZATIONAL_UNIT",
    "AssociationType": "APPLIED",
    "UpdatedAt": "2023-09-26T21:13:01.816000+00:00",
    "AssociationStatus": "SUCCESS",
    "AssociationStatusMessage": "Association applied successfully on this target."
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConfigurationPolicyAssociation。

get-configuration-policy

以下代码示例演示了如何使用 get-configuration-policy。

Amazon CLI

查看配置策略详细信息

以下 get-configuration-policy 示例检索有关指定配置策略的详细信息。

```
aws securityhub get-configuration-policy \
   --identifier "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

输出：

```
{
    "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Id": "ce5ed1e7-9639-4e2f-9313-fa87fcef944b",
    "Name": "SampleConfigurationPolicy",
    "Description": "SampleDescription",
    "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
    "CreatedAt": "2023-11-28T20:28:04.494000+00:00",
    "ConfigurationPolicy": {
        "SecurityHub": {
            "ServiceEnabled": true,
```

```
            "EnabledStandardIdentifiers": [
                "arn:aws:securityhub:eu-central-1::standards/aws-foundational-
security-best-practices/v/1.0.0",
                "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"
            ],
            "SecurityControlsConfiguration": {
                "DisabledSecurityControlIdentifiers": [
                    "CloudTrail.2"
                ],
                "SecurityControlCustomParameters": [
                    {
                        "SecurityControlId": "ACM.1",
                        "Parameters": {
                            "daysToExpiration": {
                                "ValueType": "CUSTOM",
                                "Value": {
                                    "Integer": 15
                                }
                            }
                        }
                    }
                ]
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetConfigurationPolicy。

get-enabled-standards

以下代码示例演示了如何使用 get-enabled-standards。

Amazon CLI

检索有关已启用标准的信息

以下 get-enabled-standards 示例检索有关 PCI DSS 标准的信息。

```
aws securityhub get-enabled-standards \
```

```
    --standards-subscription-arn "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
```

输出：

```
{
    "StandardsSubscriptions": [
        {
            "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/
v/3.2.1",
            "StandardsInput": { },
            "StandardsStatus": "READY",
            "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的 Amazon Security Hub 中的安全标准。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEnabledStandards。

get-finding-aggregator

以下代码示例演示了如何使用 get-finding-aggregator。

Amazon CLI

检索当前调查发现聚合的配置

以下 get-finding-aggregator 示例检索当前调查发现聚合的配置。

```
aws securityhub get-finding-aggregator \
    --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000
```

输出：

```
{
    "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000",
```

```
    "FindingAggregationRegion": "us-east-1",
    "RegionLinkingMode": "SPECIFIED_REGIONS",
    "Regions": "us-west-1,us-west-2"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看当前调查发现聚合的配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFindingAggregator。

get-finding-history

以下代码示例演示了如何使用 get-finding-history。

Amazon CLI

获取调查发现的历史记录

以下 get-finding-history 示例获取指定调查发现最近 90 天的历史记录。在此示例中，结果仅限于两条调查发现的历史记录。

```
aws securityhub get-finding-history \
    --finding-identifier Id="arn:aws:securityhub:us-
east-1:123456789012:security-control/S3.17/finding/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",ProductArn="arn:aws:securityhub:us-east-1::product/aws/securityhub"
```

输出：

```
{
    "Records": [
        {
            "FindingIdentifier": {
                "Id": "arn:aws:securityhub:us-east-1:123456789012:security-control/
S3.17/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/
securityhub"
            },
            "UpdateTime": "2023-06-02T03:15:25.685000+00:00",
            "FindingCreated": false,
            "UpdateSource": {
                "Type": "BATCH_IMPORT_FINDINGS",
                "Identity": "arn:aws:securityhub:us-east-1::product/aws/securityhub"
            },
            "Updates": [
```

```
                {
                    "UpdatedField": "Compliance.RelatedRequirements",
                    "OldValue": "[\"NIST.800-53.r5 SC-12(2)\",\"NIST.800-53.r5
 SC-12(3)\",\"NIST.800-53.r5 SC-12(6)\",\"NIST.800-53.r5 CM-3(6)\",\"NIST.800-53.r5
 SC-13\",\"NIST.800-53.r5 SC-28\",\"NIST.800-53.r5 SC-28(1)\",\"NIST.800-53.r5
 SC-7(10)\"]",
                    "NewValue": "[\"NIST.800-53.r5 SC-12(2)\",\"NIST.800-53.r5
 CM-3(6)\",\"NIST.800-53.r5 SC-13\",\"NIST.800-53.r5 SC-28\",\"NIST.800-53.r5
 SC-28(1)\",\"NIST.800-53.r5 SC-7(10)\",\"NIST.800-53.r5 CA-9(1)\",\"NIST.800-53.r5
 SI-7(6)\",\"NIST.800-53.r5 AU-9\"]"
                },
                {
                    "UpdatedField": "LastObservedAt",
                    "OldValue": "2023-06-01T09:15:38.587Z",
                    "NewValue": "2023-06-02T03:15:22.946Z"
                },
                {
                    "UpdatedField": "UpdatedAt",
                    "OldValue": "2023-06-01T09:15:31.049Z",
                    "NewValue": "2023-06-02T03:15:14.861Z"
                },
                {
                    "UpdatedField": "ProcessedAt",
                    "OldValue": "2023-06-01T09:15:41.058Z",
                    "NewValue": "2023-06-02T03:15:25.685Z"
                }
            ]
        },
        {
            "FindingIdentifier": {
                "Id": "arn:aws:securityhub:us-east-1:123456789012:security-control/
S3.17/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/
securityhub"
            },
            "UpdateTime": "2023-05-23T02:06:51.518000+00:00",
            "FindingCreated": "true",
            "UpdateSource": {
                "Type": "BATCH_IMPORT_FINDINGS",
                "Identity": "arn:aws:securityhub:us-east-1::product/aws/securityhub"
            },
            "Updates": []
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的调查发现的历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFindingHistory。

get-findings

以下代码示例演示了如何使用 get-findings。

Amazon CLI

示例 1：返回针对特定标准生成的调查发现

以下 get-findings 示例返回针对 PCI DSS 标准的调查发现。

```
aws securityhub get-findings \
    --filters '{"GeneratorId":[{"Value": "pci-dss","Comparison":"PREFIX"}]}' \
    --max-items 1
```

输出：

```
{
    "Findings": [
        {
            "SchemaVersion": "2018-10-08",
            "Id": "arn:aws:securityhub:eu-central-1:123456789012:subscription/pci-
dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub",
            "GeneratorId": "pci-dss/v/3.2.1/PCI.Lambda.2",
            "AwsAccountId": "123456789012",
            "Types": [
                "Software and Configuration Checks/Industry and Regulatory
 Standards/PCI-DSS"
            ],
            "FindingProviderFields": {
                "Severity": {
                    "Original": 0,
                    "Label": "INFORMATIONAL"
                },
                "Types": [
                    "Software and Configuration Checks/Industry and Regulatory
 Standards/PCI-DSS"
```

```
                ]
            },
            "FirstObservedAt": "2020-06-02T14:02:49.159Z",
            "LastObservedAt": "2020-06-02T14:02:52.397Z",
            "CreatedAt": "2020-06-02T14:02:49.159Z",
            "UpdatedAt": "2020-06-02T14:02:52.397Z",
            "Severity": {
                "Original": 0,
                "Label": "INFORMATIONAL",
                "Normalized": 0
            },
            "Title": "PCI.Lambda.2 Lambda functions should be in a VPC",
            "Description": "This AWS control checks whether a Lambda function is in
 a VPC.",
            "Remediation": {
                "Recommendation": {
                    "Text": "For directions on how to fix this issue, please consult
 the AWS Security Hub PCI DSS documentation.",
                    "Url": "https://docs.aws.amazon.com/console/securityhub/
PCI.Lambda.2/remediation"
                }
            },
            "ProductFields": {
                "StandardsArn": "arn:aws:securityhub:::standards/pci-dss/v/3.2.1",
                "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1",
                "ControlId": "PCI.Lambda.2",
                "RecommendationUrl": "https://docs.aws.amazon.com/console/
securityhub/PCI.Lambda.2/remediation",
                "RelatedAWSResources:0/name": "securityhub-lambda-inside-
vpc-0e904a3b",
                "RelatedAWSResources:0/type": "AWS::Config::ConfigRule",
                "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.Lambda.2",
                "aws/securityhub/SeverityLabel": "INFORMATIONAL",
                "aws/securityhub/ProductName": "Security Hub",
                "aws/securityhub/CompanyName": "AWS",
                "aws/securityhub/FindingId": "arn:aws:securityhub:eu-
central-1::product/aws/securityhub/arn:aws:securityhub:eu-
central-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        },
            "Resources": [
                {
```

```
                    "Type": "AwsAccount",
                    "Id": "AWS::::Account:123456789012",
                    "Partition": "aws",
                    "Region": "us-west-1"
                }
            ],
            "Compliance": {
                "Status": "PASSED",
                "RelatedRequirements": [
                    "PCI DSS 1.2.1",
                    "PCI DSS 1.3.1",
                    "PCI DSS 1.3.2",
                    "PCI DSS 1.3.4"
                ]
            },
            "WorkflowState": "NEW",
            "Workflow": {
                "Status": "NEW"
            },
            "RecordState": "ARCHIVED"
        }
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ=="
}
```

示例 2：返回工作流状态为 NOTIFIED 的关键调查发现

以下 `get-findings` 示例返回严重性标签值为 CRITICAL 且工作流状态为 NOTIFIED 的调查发现。结果按“置信度”值降序排序。

```
aws securityhub get-findings \
    --filters '{"SeverityLabel":[{"Value":
 "CRITICAL","Comparison":"EQUALS"}],"WorkflowStatus":
 [{"Value":"NOTIFIED","Comparison":"EQUALS"}]}' \
    --sort-criteria '{ "Field": "Confidence", "SortOrder": "desc"}' \
    --max-items 1
```

输出：

```
{
    "Findings": [
        {
            "SchemaVersion": "2018-10-08",
```

```
            "Id": "arn:aws:securityhub:us-west-1: 123456789012:subscription/cis-aws-
foundations-benchmark/v/1.2.0/1.13/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/securityhub",
            "GeneratorId": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0/rule/1.13",
            "AwsAccountId": "123456789012",
            "Types": [
                "Software and Configuration Checks/Industry and Regulatory
 Standards/CIS AWS Foundations Benchmark"
            ],
            "FindingProviderFields" {
                "Severity": {
                    "Original": 90,
                    "Label": "CRITICAL"
                },
                "Types": [
                    "Software and Configuration Checks/Industry and Regulatory
 Standards/CIS AWS Foundations Benchmark"
                ]
            },
            "FirstObservedAt": "2020-05-21T20:16:34.752Z",
            "LastObservedAt": "2020-06-09T08:16:37.171Z",
            "CreatedAt": "2020-05-21T20:16:34.752Z",
            "UpdatedAt": "2020-06-09T08:16:36.430Z",
            "Severity": {
                "Original": 90,
                "Label": "CRITICAL",
                "Normalized": 90
            },
            "Title": "1.13 Ensure MFA is enabled for the \"root\" account",
            "Description": "The root account is the most privileged user in an AWS
 account. MFA adds an extra layer of protection on top of a user name and password.
 With MFA enabled, when a user signs in to an AWS website, they will be prompted for
 their user name and password as well as for an authentication code from their AWS
 MFA device.",
            "Remediation": {
                "Recommendation": {
                    "Text": "For directions on how to fix this issue, please consult
 the AWS Security Hub CIS documentation.",
                    "Url": "https://docs.aws.amazon.com/console/securityhub/
standards-cis-1.13/remediation"
                }
            },
            "ProductFields": {
```

```
                "StandardsGuideArn": "arn:aws:securityhub:::ruleset/cis-aws-
foundations-benchmark/v/1.2.0",
                "StandardsGuideSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/cis-aws-foundations-benchmark/v/1.2.0",
                "RuleId": "1.13",
                "RecommendationUrl": "https://docs.aws.amazon.com/console/
securityhub/standards-cis-1.13/remediation",
                "RelatedAWSResources:0/name": "securityhub-root-account-mfa-
enabled-5pftha",
                "RelatedAWSResources:0/type": "AWS::Config::ConfigRule",
                "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/cis-aws-foundations-benchmark/v/1.2.0/1.13",
                "aws/securityhub/SeverityLabel": "CRITICAL",
                "aws/securityhub/ProductName": "Security Hub",
                "aws/securityhub/CompanyName": "AWS",
                "aws/securityhub/FindingId": "arn:aws:securityhub:us-
west-1::product/aws/securityhub/arn:aws:securityhub:us-
west-1:123456789012:subscription/cis-aws-foundations-benchmark/v/1.2.0/1.13/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            },
            "Resources": [
                {
                    "Type": "AwsAccount",
                    "Id": "AWS::::Account:123456789012",
                    "Partition": "aws",
                    "Region": "us-west-1"
                }
            ],
            "Compliance": {
                "Status": "FAILED"
            },
            "WorkflowState": "NEW",
            "Workflow": {
                "Status": "NOTIFIED"
            },
            "RecordState": "ACTIVE"
        }
    ]
}
```


有关更多信息，请参阅《Amazon Security Hub 用户指南》中的筛选和分组调查发现。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFindings。

get-insight-results

以下代码示例演示了如何使用 get-insight-results。

Amazon CLI

检索结果以获取见解

以下 get-insight-results 示例返回具有指定 ARN 见解的见解结果列表。

```
aws securityhub get-insight-results \
    --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

输出：

```
{
    "InsightResults": {
        "GroupByAttribute": "ResourceId",
        "InsightArn": "arn:aws:securityhub:us-
west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "ResultValues": [
            {
                "Count": 10,
                "GroupByAttributeValue": "AWS::::Account:123456789111"
            },
            {
                "Count": 3,
                "GroupByAttributeValue": "AWS::::Account:123456789222"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看见解结果和调查发现并对其采取行动。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInsightResults。

get-insights

以下代码示例演示了如何使用 get-insights。

Amazon CLI

检索有关见解的详细信息

以下 get-insights 示例检索具有指定 ARN 见解的配置详细信息。

```
aws securityhub get-insights \
    --insight-arns "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

输出：

```
{
    "Insights": [
        {
            "Filters": {
               "ResourceType": [
                    {
                        "Comparison": "EQUALS",
                        "Value": "AwsIamRole"
                    }
                ],
                "SeverityLabel": [
                    {
                        "Comparison": "EQUALS",
                        "Value": "CRITICAL"
                    }
                ],
            },
            "GroupByAttribute": "ResourceId",
            "InsightArn": "arn:aws:securityhub:us-
west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
            "Name": "Critical role findings"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的 Amazon Security Hub 见解。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInsights。

get-invitations-count

以下代码示例演示了如何使用 get-invitations-count。

Amazon CLI

检索未被接受的邀请数量

以下 get-invitations-count 示例检索请求账户拒绝或未回复的邀请数量。

```
aws securityhub get-invitations-count
```

输出：

```
{
  "InvitationsCount": 3
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetInvitationsCount。

get-master-account

以下代码示例演示了如何使用 get-master-account。

Amazon CLI

检索管理员帐户的相关信息

以下 get-master-account 示例检索请求账户的管理员账户的相关信息。

```
aws securityhub get-master-account
```

输出：

```
{
   "Master": {
      "AccountId": "123456789012",
      "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
      "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
```

```
      "MemberStatus": "ASSOCIATED"
   }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMasterAccount。

get-members

以下代码示例演示了如何使用 get-members。

Amazon CLI

检索所选成员账户的相关信息

以下 get-members 示例检索指定成员账户的相关信息。

```
aws securityhub get-members \
    --account-ids "444455556666" "777788889999"
```

输出：

```
{
    "Members": [
        {
            "AccountId": "123456789111",
            "AdministratorId": "123456789012",
            "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
            "MasterId": "123456789012",
            "MemberStatus": "ASSOCIATED",
            "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
        },
        {
            "AccountId": "123456789222",
            "AdministratorId": "123456789012",
            "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
            "MasterId": "123456789012",
            "MemberStatus": "ASSOCIATED",
            "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
        }
    ],
```

```
    "UnprocessedAccounts": [ ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMembers。

get-security-control-definition

以下代码示例演示了如何使用 get-security-control-definition。

Amazon CLI

获取安全控件的定义详细信息

以下 get-security-control-definition 示例检索 Security Hub 安全控件的定义详细信息。详细信息包括控件标题、描述、区域可用性、参数和其他信息。

```
aws securityhub get-security-control-definition \
    --security-control-id ACM.1
```

输出：

```
{
    "SecurityControlDefinition": {
        "SecurityControlId": "ACM.1",
        "Title": "Imported and ACM-issued certificates should be renewed after a
 specified time period",
        "Description": "This control checks whether an AWS Certificate Manager
 (ACM) certificate is renewed within the specified time period. It checks both
 imported certificates and certificates provided by ACM. The control fails if the
 certificate isn't renewed within the specified time period. Unless you provide a
 custom parameter value for the renewal period, Security Hub uses a default value of
 30 days.",
        "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/ACM.1/
remediation",
        "SeverityRating": "MEDIUM",
        "CurrentRegionAvailability": "AVAILABLE",
        "ParameterDefinitions": {
            "daysToExpiration": {
                "Description": "Number of days within which the ACM certificate must
 be renewed",
```

```
                "ConfigurationOptions": {
                    "Integer": {
                        "DefaultValue": 30,
                        "Min": 14,
                        "Max": 365
                    }
                }
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的自定义控件参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSecurityControlDefinition。

`invite-members`

以下代码示例演示了如何使用 `invite-members`。

Amazon CLI

向成员账户发送邀请

以下 `invite-members` 示例向指定的成员账户发送邀请。

```
aws securityhub invite-members \
    --account-ids "123456789111" "123456789222"
```

输出：

```
{
    "UnprocessedAccounts": []
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InviteMembers。

`list-automation-rules`

以下代码示例演示了如何使用 `list-automation-rules`。

Amazon CLI

查看自动化规则列表

以下 `list-automation-rules` 示例列出 Amazon 账户的自动化规则。只有 Security Hub 管理员账户才能运行此命令。

```
aws securityhub list-automation-rules \
    --max-results 3 \
    --next-token NULL
```

输出：

```
{
    "AutomationRulesMetadata": [
        {
            "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "RuleStatus": "ENABLED",
            "RuleOrder": 1,
            "RuleName": "Suppress informational findings",
            "Description": "Suppress GuardDuty findings with Informational
 severity",
            "IsTerminal": false,
            "CreatedAt": "2023-05-31T17:56:14.837000+00:00",
            "UpdatedAt": "2023-05-31T17:59:38.466000+00:00",
            "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
        },
        {
            "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "RuleStatus": "ENABLED",
            "RuleOrder": 1,
            "RuleName": "sample rule",
            "Description": "A sample rule",
            "IsTerminal": false,
            "CreatedAt": "2023-07-15T23:37:20.223000+00:00",
            "UpdatedAt": "2023-07-15T23:37:20.223000+00:00",
            "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
        },
        {
            "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
```

```
            "RuleStatus": "ENABLED",
            "RuleOrder": 1,
            "RuleName": "sample rule",
            "Description": "A sample rule",
            "IsTerminal": false,
            "CreatedAt": "2023-07-15T23:45:25.126000+00:00",
            "UpdatedAt": "2023-07-15T23:45:25.126000+00:00",
            "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看自动化规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAutomationRules。

list-configuration-policies

以下代码示例演示了如何使用 list-configuration-policies。

Amazon CLI

列出配置策略摘要

以下 list-configuration-policies 示例列出组织的配置策略摘要。

```
aws securityhub list-configuration-policies \
    --max-items 3
```

输出：

```
{
    "ConfigurationPolicySummaries": [
        {
            "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-
policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Name": "SampleConfigurationPolicy1",
            "Description": "SampleDescription1",
            "UpdatedAt": "2023-09-26T21:08:36.214000+00:00",
            "ServiceEnabled": true
        },
        {
```

```
            "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-
policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Name": "SampleConfigurationPolicy2",
            "Description": "SampleDescription2"
            "UpdatedAt": "2023-11-28T19:26:25.207000+00:00",
            "ServiceEnabled": true
        },
        {
            "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-
policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "Name": "SampleConfigurationPolicy3",
            "Description": "SampleDescription3",
            "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
            "ServiceEnabled": true
        }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConfigurationPolicies。

`list-configuration-policy-associations`

以下代码示例演示了如何使用 `list-configuration-policy-associations`。

Amazon CLI

列出配置关联

以下 `list-configuration-policy-associations` 示例列出组织的配置关联摘要。响应包括与配置策略和自行管理行为的关联。

```
aws securityhub list-configuration-policy-associations \
    --filters '{"AssociationType": "APPLIED"}' \
    --max-items 4
```

输出：

```
{
    "ConfigurationPolicyAssociationSummaries": [
        {
```

```
            "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "TargetId": "r-1ab2",
            "TargetType": "ROOT",
            "AssociationType": "APPLIED",
            "UpdatedAt": "2023-11-28T19:26:49.417000+00:00",
            "AssociationStatus": "FAILED",
            "AssociationStatusMessage": "Policy association failed because 2
 organizational units or accounts under this root failed."
        },
        {
            "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "TargetId": "ou-1ab2-c3de4f5g",
            "TargetType": "ORGANIZATIONAL_UNIT",
            "AssociationType": "APPLIED",
            "UpdatedAt": "2023-09-26T21:14:05.283000+00:00",
            "AssociationStatus": "FAILED",
            "AssociationStatusMessage": "One or more children under this target
 failed association."
        },
        {
            "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "TargetId": "ou-6hi7-8j91kl2m",
            "TargetType": "ORGANIZATIONAL_UNIT",
            "AssociationType": "APPLIED",
            "UpdatedAt": "2023-09-26T21:13:01.816000+00:00",
            "AssociationStatus": "SUCCESS",
            "AssociationStatusMessage": "Association applied successfully on this
 target."
        },
        {
            "ConfigurationPolicyId": "SELF_MANAGED_SECURITY_HUB",
            "TargetId": "111122223333",
            "TargetType": "ACCOUNT",
            "AssociationType": "APPLIED",
            "UpdatedAt": "2023-11-28T22:01:26.409000+00:00",
            "AssociationStatus": "SUCCESS"
    }
}
```

有关更多信息，请参阅《Amazon Security Hub User Guide》中的 Viewing configuration policy status and details。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListConfigurationPolicyAssociations。

list-enabled-products-for-import

以下代码示例演示了如何使用 list-enabled-products-for-import。

Amazon CLI

返回已启用产品集成的列表

以下 list-enabled-products-for-import 示例返回当前已启用产品集成的订阅 ARN 列表。

```
aws securityhub list-enabled-products-for-import
```

输出：

```
{
    "ProductSubscriptions": [ "arn:aws:securityhub:us-west-1:123456789012:product-subscription/crowdstrike/crowdstrike-falcon", "arn:aws:securityhub:us-west-1:123456789012:product-subscription/aws/securityhub" ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理产品集成。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEnabledProductsForImport。

list-finding-aggregators

以下代码示例演示了如何使用 list-finding-aggregators。

Amazon CLI

列出可用的小部件

以下 list-finding-aggregators 示例返回调查发现聚合配置的 ARN。

```
aws securityhub list-finding-aggregators
```

输出：

```
{
    "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-aggregator/123e4567-e89b-12d3-a456-426652340000"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看当前调查发现聚合的配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFindingAggregators。

list-invitations

以下代码示例演示了如何使用 list-invitations。

Amazon CLI

显示邀请列表

以下 list-invitations 示例检索发送到请求账户的邀请列表。

```
aws securityhub list-invitations
```

输出：

```
{
    "Invitations": [
        {
            "AccountId": "123456789012",
            "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
            "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
            "MemberStatus": "ASSOCIATED"
        }
    ],
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListInvitations。

list-members

以下代码示例演示了如何使用 list-members。

Amazon CLI

检索成员账户列表

以下 list-members 示例返回请求管理员账户的成员账户列表。

```
aws securityhub list-members
```

输出：

```
{
    "Members": [
        {
            "AccountId": "123456789111",
            "AdministratorId": "123456789012",
            "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
            "MasterId": "123456789012",
            "MemberStatus": "ASSOCIATED",
            "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
        },
        {
            "AccountId": "123456789222",
            "AdministratorId": "123456789012",
            "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
            "MasterId": "123456789012",
            "MemberStatus": "ASSOCIATED",
            "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
        }
    ],
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理管理员和成员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMembers。

`list-organization-admin-accounts`

以下代码示例演示了如何使用 `list-organization-admin-accounts`。

Amazon CLI

列出指定的 Security Hub 管理员账户

以下 `list-organization-admin-accounts` 示例列出组织的 Security Hub 管理员账户。

```
aws securityhub list-organization-admin-accounts
```

输出：

```
{
    AdminAccounts": [
        { "AccountId": "777788889999" },
        { "Status": "ENABLED" }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的指定 Security Hub 管理员账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizationAdminAccounts。

list-security-control-definitions

以下代码示例演示了如何使用 list-security-control-definitions。

Amazon CLI

示例 1：列出所有可用的安全控件

以下 list-security-control-definitions 示例列出所有 Security Hub 标准中可用的安全控件。此示例结果限制为三个控件。

```
aws securityhub list-security-control-definitions \
    --max-items 3
```

输出：

```
{
    "SecurityControlDefinitions": [
        {
            "SecurityControlId": "ACM.1",
            "Title": "Imported and ACM-issued certificates should be renewed after a
 specified time period",
            "Description": "This control checks whether an AWS Certificate Manager
 (ACM) certificate is renewed within the specified time period. It checks both
 imported certificates and certificates provided by ACM. The control fails if the
 certificate isn't renewed within the specified time period. Unless you provide a
 custom parameter value for the renewal period, Security Hub uses a default value of
 30 days.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
ACM.1/remediation",
            "SeverityRating": "MEDIUM",
```

```
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": [
                "Parameters"
            ]
        },
        {
            "SecurityControlId": "ACM.2",
            "Title": "RSA certificates managed by ACM should use a key length of at
 least 2,048 bits",
            "Description": "This control checks whether RSA certificates managed by
 AWS Certificate Manager use a key length of at least 2,048 bits. The control fails
 if the key length is smaller than 2,048 bits.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
ACM.2/remediation",
            "SeverityRating": "HIGH",
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": []
        },
        {
            "SecurityControlId": "APIGateway.1",
            "Title": "API Gateway REST and WebSocket API execution logging should be
 enabled",
            "Description": "This control checks whether all stages of an Amazon
 API Gateway REST or WebSocket API have logging enabled. The control fails if
 the 'loggingLevel' isn't 'ERROR' or 'INFO' for all stages of the API. Unless you
 provide custom parameter values to indicate that a specific log type should be
 enabled, Security Hub produces a passed finding if the logging level is either
 'ERROR' or 'INFO'.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
APIGateway.1/remediation",
            "SeverityRating": "MEDIUM",
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": [
                "Parameters"
            ]
        }
    ],
    "NextToken": "U2FsdGVkX1/UprCPzxVbkDeHikDXbDxfgJZ1w2RG1XWsFPTMTIQPVE0m/
FduIGxS7ObRtAbaUt/8/RCQcg2PU0YXI20hH/GrhoOTgv+TSm0qvQVFhkJepWmqh
+NYawjocVBeos6xzn/8qnbF9IuwGg=="
}
```


有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看标准的详细信息。

示例 2：列出特定标准的可用安全控件

以下 `list-security-control-definitions` 示例列出 CIS Amazon Foundations Benchmark v1.4.0 的可用安全控件。此示例结果限制为三个控件。

```
aws securityhub list-security-control-definitions \
    --standards-arn "arn:aws:securityhub:us-east-1::standards/cis-aws-foundations-benchmark/v/1.4.0" \
    --max-items 3
```

输出：

```
{
    "SecurityControlDefinitions": [
        {
            "SecurityControlId": "CloudTrail.1",
            "Title": "CloudTrail should be enabled and configured with at least one
 multi-Region trail that includes read and write management events",
            "Description": "This AWS control checks that there is at least one
 multi-region AWS CloudTrail trail includes read and write management events.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.1/remediation",
            "SeverityRating": "HIGH",
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": []
        },
        {
            "SecurityControlId": "CloudTrail.2",
            "Title": "CloudTrail should have encryption at-rest enabled",
            "Description": "This AWS control checks whether AWS CloudTrail is
 configured to use the server side encryption (SSE) AWS Key Management Service (AWS
 KMS) customer master key (CMK) encryption. The check will pass if the KmsKeyId is
 defined.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.2/remediation",
            "SeverityRating": "MEDIUM",
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": []
        },
        {
            "SecurityControlId": "CloudTrail.4",
            "Title": "CloudTrail log file validation should be enabled",
```

```
            "Description": "This AWS control checks whether CloudTrail log file
 validation is enabled.",
            "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.4/remediation",
            "SeverityRating": "MEDIUM",
            "CurrentRegionAvailability": "AVAILABLE",
            "CustomizableProperties": []
        }
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAzfQ=="
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的查看标准的详细信息。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSecurityControlDefinitions。

`list-standards-control-associations`

以下代码示例演示了如何使用 `list-standards-control-associations`。

Amazon CLI

获取每个已启用标准中控件的启用状态

以下 `list-standards-control-associations` 示例列出每个已启用标准中 CloudTrail.1 的启用状态。

```
aws securityhub list-standards-control-associations \
    --security-control-id CloudTrail.1
```

输出：

```
{
    "StandardsControlAssociationSummaries": [
        {
            "StandardsArn": "arn:aws:securityhub:us-east-2::standards/nist-800-53/
v/5.0.0",
            "SecurityControlId": "CloudTrail.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
            "AssociationStatus": "ENABLED",
            "RelatedRequirements": [
                "NIST.800-53.r5 AC-2(4)",
```

```
                "NIST.800-53.r5 AC-4(26)",
                "NIST.800-53.r5 AC-6(9)",
                "NIST.800-53.r5 AU-10",
                "NIST.800-53.r5 AU-12",
                "NIST.800-53.r5 AU-2",
                "NIST.800-53.r5 AU-3",
                "NIST.800-53.r5 AU-6(3)",
                "NIST.800-53.r5 AU-6(4)",
                "NIST.800-53.r5 AU-14(1)",
                "NIST.800-53.r5 CA-7",
                "NIST.800-53.r5 SC-7(9)",
                "NIST.800-53.r5 SI-3(8)",
                "NIST.800-53.r5 SI-4(20)",
                "NIST.800-53.r5 SI-7(8)",
                "NIST.800-53.r5 SA-8(22)"
            ],
            "UpdatedAt": "2023-05-15T17:52:21.304000+00:00",
            "StandardsControlTitle": "CloudTrail should be enabled and configured
 with at least one multi-Region trail that includes read and write management
 events",
            "StandardsControlDescription": "This AWS control checks that there is
 at least one multi-region AWS CloudTrail trail includes read and write management
 events."
        },
        {
            "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0",
            "SecurityControlId": "CloudTrail.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
            "AssociationStatus": "ENABLED",
            "RelatedRequirements": [
                "CIS AWS Foundations 2.1"
            ],
            "UpdatedAt": "2020-02-10T21:22:53.998000+00:00",
            "StandardsControlTitle": "Ensure CloudTrail is enabled in all regions",
            "StandardsControlDescription": "AWS CloudTrail is a web service that
 records AWS API calls for your account and delivers log files to you. The recorded
 information includes the identity of the API caller, the time of the API call,
 the source IP address of the API caller, the request parameters, and the response
 elements returned by the AWS service."
        },
        {
```

```
            "StandardsArn": "arn:aws:securityhub:us-east-2::standards/aws-
foundational-security-best-practices/v/1.0.0",
            "SecurityControlId": "CloudTrail.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
            "AssociationStatus": "DISABLED",
            "RelatedRequirements": [],
            "UpdatedAt": "2023-05-15T19:31:52.671000+00:00",
            "UpdatedReason": "Alternative compensating controls are in place",
            "StandardsControlTitle": "CloudTrail should be enabled and configured
 with at least one multi-Region trail that includes read and write management
 events",
            "StandardsControlDescription": "This AWS control checks that there is
 at least one multi-region AWS CloudTrail trail includes read and write management
 events."
        },
        {
            "StandardsArn": "arn:aws:securityhub:us-east-2::standards/cis-aws-
foundations-benchmark/v/1.4.0",
            "SecurityControlId": "CloudTrail.1",
            "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
            "AssociationStatus": "ENABLED",
            "RelatedRequirements": [
                "CIS AWS Foundations Benchmark v1.4.0/3.1"
            ],
            "UpdatedAt": "2022-11-10T15:40:36.021000+00:00",
            "StandardsControlTitle": "Ensure CloudTrail is enabled in all regions",
            "StandardsControlDescription": "AWS CloudTrail is a web service that
 records AWS API calls for your account and delivers log files to you. The recorded
 information includes the identity of the API caller, the time of the API call,
 the source IP address of the API caller, the request parameters, and the response
 elements returned by the AWS service. CloudTrail provides a history of AWS API
 calls for an account, including API calls made via the Management Console, SDKs,
 command line tools, and higher-level AWS services (such as CloudFormation)."
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的在指定标准中启用或禁用控件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListStandardsControlAssociations。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

检索分配给资源的标签

以下 list-tags-for-resource 示例返回分配给指定 Hub 资源的标签。

```
aws securityhub list-tags-for-resource \
    --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default"
```

输出：

```
{
    "Tags": {
        "Department" : "Operations",
        "Area" : "USMidwest"
    }
}
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 Amazon::SecurityHub::Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

start-configuration-policy-association

以下代码示例演示了如何使用 start-configuration-policy-association。

Amazon CLI

示例 1：关联配置策略

以下 start-configuration-policy-association 示例会将指定的配置策略与指定的组织单位相关联。配置可以与目标账户、组织单位或根用户相关联。

```
aws securityhub start-configuration-policy-association \
    --configuration-policy-identifier "arn:aws:securityhub:eu-
central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333" \
    --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

输出：

```
{
    "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "TargetId": "ou-6hi7-8j91kl2m",
    "TargetType": "ORGANIZATIONAL_UNIT",
    "AssociationType": "APPLIED",
    "UpdatedAt": "2023-11-29T17:40:52.468000+00:00",
    "AssociationStatus": "PENDING"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建并关联 Security Hub 配置策略。

示例 2：关联自行管理配置

以下 `start-configuration-policy-association` 示例会将自行管理配置与指定账户相关联。

```
aws securityhub start-configuration-policy-association \
    --configuration-policy-identifier "SELF_MANAGED_SECURITY_HUB" \
    --target '{"OrganizationalUnitId": "123456789012"}'
```

输出：

```
{
    "ConfigurationPolicyId": "SELF_MANAGED_SECURITY_HUB",
    "TargetId": "123456789012",
    "TargetType": "ACCOUNT",
    "AssociationType": "APPLIED",
    "UpdatedAt": "2023-11-29T17:40:52.468000+00:00",
    "AssociationStatus": "PENDING"
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建并关联 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartConfigurationPolicyAssociation。

`start-configuration-policy-disassociation`

以下代码示例演示了如何使用 `start-configuration-policy-disassociation`。

Amazon CLI

示例 1：解除配置策略关联

以下 start-configuration-policy-disassociation 示例解除指定组织单位的配置策略关联。配置可以解除与目标账户、组织单位或根用户的关联。

```
aws securityhub start-configuration-policy-disassociation \
    --configuration-policy-identifier "arn:aws:securityhub:eu-
central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333" \
    --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的解除配置与账户和 OU 的关联。

示例 2：解除自行管理配置关联

以下 start-configuration-policy-disassociation 示例解除指定账户的自行管理配置关联。

```
aws securityhub start-configuration-policy-disassociation \
    --configuration-policy-identifier "SELF_MANAGED_SECURITY_HUB" \
    --target '{"AccountId": "123456789012"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的解除配置与账户和 OU 的关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartConfigurationPolicyDisassociation。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签添加到资源

以下 tag-resource 示例为指定的 Hub 资源分配“Department”和“Area”标签的值。

```
aws securityhub tag-resource \
    --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default" \
    --tags '{"Department":"Operations", "Area":"USMidwest"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 Amazon::SecurityHub::Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从资源中删除标签值

以下 untag-resource 示例从指定的 Hub 资源中删除“Department”标签。

```
aws securityhub untag-resource \
    --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default" \
    --tag-keys "Department"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 Amazon::SecurityHub::Hub。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-action-target

以下代码示例演示了如何使用 update-action-target。

Amazon CLI

更新自定义操作

以下 update-action-target 示例更新由指定 ARN 标识的自定义操作名称。

```
aws securityhub update-action-target \
    --action-target-arn "arn:aws:securityhub:us-west-1:123456789012:action/custom/
Remediation" \
```

```
    --name "Send to remediation"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的创建自定义操作并将其与 CloudWatch 事件规则关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateActionTarget。

update-configuration-policy

以下代码示例演示了如何使用 update-configuration-policy。

Amazon CLI

更新配置策略

以下 update-configuration-policy 示例更新现有配置策略以使用指定的设置。

```
aws securityhub update-configuration-policy \
    --identifier "arn:aws:securityhub:eu-central-1:508236694226:configuration-
policy/09f37766-57d8-4ede-9d33-5d8b0fecf70e" \
    --name "SampleConfigurationPolicyUpdated" \
    --description "SampleDescriptionUpdated" \
    --configuration-policy '{"SecurityHub": {"ServiceEnabled":
 true, "EnabledStandardIdentifiers": ["arn:aws:securityhub:eu-
central-1::standards/aws-foundational-security-best-practices/
v/1.0.0","arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"],"SecurityControlsConfiguration":{"DisabledSecurityControlIdentifiers":
 ["CloudWatch.1"], "SecurityControlCustomParameters": [{"SecurityControlId":
 "ACM.1", "Parameters": {"daysToExpiration": {"ValueType": "CUSTOM", "Value":
 {"Integer": 21}}}}]}}}' \
    --updated-reason "Disabling CloudWatch.1 and changing parameter value"
```

输出：

```
{
    "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Name": "SampleConfigurationPolicyUpdated",
    "Description": "SampleDescriptionUpdated",
    "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
```

```
    "CreatedAt": "2023-11-28T20:28:04.494000+00:00",
    "ConfigurationPolicy": {
        "SecurityHub": {
            "ServiceEnabled": true,
            "EnabledStandardIdentifiers": [
                "arn:aws:securityhub:eu-central-1::standards/aws-foundational-
security-best-practices/v/1.0.0",
                "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"
            ],
            "SecurityControlsConfiguration": {
                "DisabledSecurityControlIdentifiers": [
                    "CloudWatch.1"
                ],
                "SecurityControlCustomParameters": [
                    {
                        "SecurityControlId": "ACM.1",
                        "Parameters": {
                            "daysToExpiration": {
                                "ValueType": "CUSTOM",
                                "Value": {
                                    "Integer": 21
                                }
                            }
                        }
                    }
                ]
            }
        }
    }
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的更新 Security Hub 配置策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateConfigurationPolicy。

update-finding-aggregator

以下代码示例演示了如何使用 update-finding-aggregator。

Amazon CLI

更新当前调查发现聚合配置

以下 update-finding-aggregator 示例会将调查发现聚合配置更改为从选定区域进行链接。它从聚合区域（即美国东部（弗吉尼亚））运行。它选择美国西部（北加利福尼亚）和美国西部（俄勒冈州）作为链接区域。

```
aws securityhub update-finding-aggregator \
    --region us-east-1 \
    --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-aggregator/123e4567-e89b-12d3-a456-426652340000 \
    --region-linking-mode SPECIFIED_REGIONS \
    --regions us-west-1,us-west-2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的更新调查发现聚合配置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFindingAggregator。

update-insight

以下代码示例演示了如何使用 update-insight。

Amazon CLI

示例 1：更改自定义见解的筛选条件

以下 update-insight 示例更改自定义见解的筛选条件。更新的见解会查找与 Amazon 角色相关的严重性较高的调查发现。

```
aws securityhub update-insight \
    --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
    --filters '{"ResourceType": [{ "Comparison": "EQUALS", "Value": "AwsIamRole"}], "SeverityLabel": [{"Comparison": "EQUALS", "Value": "HIGH"}]}' \
    --name "High severity role findings"
```

示例 2：更改自定义见解的分组属性

以下 update-insight 示例使用指定 ARN 更改自定义见解的分组属性。新的分组属性是资源 ID。

```
aws securityhub update-insight \
```

```
    --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
    --group-by-attribute "ResourceId" \
    --name "Critical role findings"
```

输出：

```
{
    "Insights": [
        {
            "InsightArn": "arn:aws:securityhub:us-
west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
            "Name": "Critical role findings",
            "Filters": {
                "SeverityLabel": [
                    {
                        "Value": "CRITICAL",
                        "Comparison": "EQUALS"
                    }
                ],
                "ResourceType": [
                    {
                        "Value": "AwsIamRole",
                        "Comparison": "EQUALS"
                    }
                ]
            },
            "GroupByAttribute": "ResourceId"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的管理自定义见解。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateInsight。

update-organization-configuration

以下代码示例演示了如何使用 update-organization-configuration。

Amazon CLI

更新为组织配置 Security Hub 的方式

以下 update-organization-configuration 示例指定 Security Hub 应使用集中配置来配置组织。运行此命令后，委托的 Security Hub 管理员可以通过创建和管理配置策略来配置组织。委托的管理员也可以使用此命令从集中配置切换到本地配置。如果配置类型为本地配置，委托管理员可以选择是否在新组织账户中自动启用 Security Hub 和默认安全标准。

```
aws securityhub update-organization-configuration \
    --no-auto-enable \
    --organization-configuration '{"ConfigurationType": "CENTRAL"}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的使用 Amazon Organizations 管理账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOrganizationConfiguration。

update-security-control

以下代码示例演示了如何使用 update-security-control。

Amazon CLI

更新安全控件属性

以下 update-security-control 示例为 Security Hub 安全控件参数指定自定义值。

```
aws securityhub update-security-control \
    --security-control-id ACM.1 \
    --parameters '{"daysToExpiration": {"ValueType": "CUSTOM", "Value": {"Integer": 15}}}' \
    --last-update-reason "Internal compliance requirement"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的自定义控件参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecurityControl。

update-security-hub-configuration

以下代码示例演示了如何使用 update-security-hub-configuration。

Amazon CLI

更新 Security Hub 配置

以下 update-security-hub-configuration 示例会将 Security Hub 配置为自动为已启用标准启用新控件。

```
aws securityhub update-security-hub-configuration \
    --auto-enable-controls
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的自动启用新控件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSecurityHubConfiguration。

update-standards-control

以下代码示例演示了如何使用 update-standards-control。

Amazon CLI

示例 1：禁用控件

以下 update-standards-control 示例禁用 PCI.AutoScaling.1 控件。

```
aws securityhub update-standards-control \
    --standards-control-arn "arn:aws:securityhub:us-west-1:123456789012:control/pci-dss/v/3.2.1/PCI.AutoScaling.1" \
    --control-status "DISABLED" \
    --disabled-reason "Not applicable for my service"
```

此命令不生成任何输出。

示例 2：启用控件

以下 update-standards-control 示例启用 PCI.AutoScaling.1 控件。

```
aws securityhub update-standards-control \
```

```
    --standards-control-arn "arn:aws:securityhub:us-west-1:123456789012:control/pci-dss/v/3.2.1/PCI.AutoScaling.1" \
    --control-status "ENABLED"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Hub 用户指南》中的禁用或启用单个控件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateStandardsControl。

使用 Amazon CLI 的 Security Lake 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Security Lake 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-aws-log-source`

以下代码示例演示了如何使用 `create-aws-log-source`。

Amazon CLI

将原生支持的 Amazon Web Service 添加为 Amazon Security Lake 源

以下 `create-aws-logsource` 示例在指定账户和区域中添加 VPC 流日志作为 Security Lake 源。

```
aws securitylake create-aws-log-source \
    --sources '[{"regions": ["us-east-1"], "accounts": ["123456789012"],
 "sourceName": "SH_FINDINGS", "sourceVersion": "2.0"}]'
```

输出：

```
{
    "failed": [
        "123456789012"
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的添加 Amazon 服务作为源。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateAwsLogSource。

create-custom-log-source

以下代码示例演示了如何使用 create-custom-log-source。

Amazon CLI

将自定义源添加为 Amazon Security Lake 源

以下 create-custom-logsource 示例在指定日志提供商账户和指定区域中添加自定义源作为 Security Lake 源。

```
aws securitylake create-custom-log-source \
    --source-name "VPC_FLOW" \
    --event-classes '["DNS_ACTIVITY", "NETWORK_ACTIVITY"]' \
    --configuration '{"crawlerConfiguration": {"roleArn": "arn:aws:glue:eu-
west-2:123456789012:crawler/E1WG1ZNPRXT0D4"},"providerIdentity": {"principal":
 "029189416600","externalId": "123456789012"}}' --region "us-east-1"
```

输出：

```
{
    "customLogSource": {
        "attributes": {
            "crawlerArn": "arn:aws:glue:eu-west-2:123456789012:crawler/
E1WG1ZNPRXT0D4",
            "databaseArn": "arn:aws:glue:eu-west-2:123456789012:database/
E1WG1ZNPRXT0D4",
            "tableArn": "arn:aws:glue:eu-west-2:123456789012:table/E1WG1ZNPRXT0D4"
        },
        "provider": {
```

```
            "location": "amzn-s3-demo-bucket--usw2-az1--x-s3",
            "roleArn": "arn:aws:iam::123456789012:role/AmazonSecurityLake-Provider-
testCustom2-eu-west-2"
        },
        "sourceName": "testCustom2"
        "sourceVersion": "2.0"
    }
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的添加自定义源。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateCustomLogSource。

create-data-lake-exception-subscription

以下代码示例演示了如何使用 create-data-lake-exception-subscription。

Amazon CLI

发送 Security Lake 异常通知

以下 create-data-lake-exception-subscription 示例通过短信向指定账户发送有关 Security Lake 异常的通知。异常消息将在指定时间段内保留。

```
aws securitylake create-data-lake-exception-subscription \
    --notification-endpoint "123456789012" \
    --exception-time-to-live 30 \
    --subscription-protocol "sms"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 故障排除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataLakeExceptionSubscription。

create-data-lake-organization-configuration

以下代码示例演示了如何使用 create-data-lake-organization-configuration。

Amazon CLI

在新组织账户中配置 Security Lake

以下 `create-data-lake-organization-configuration` 示例启用 Security Lake 以及新组织账户中指定源事件和日志的收集。

```
aws securitylake create-data-lake-organization-configuration \
    --auto-enable-new-account '[{"region":"us-east-1","sources":
[{"sourceName":"SH_FINDINGS","sourceVersion": "1.0"}]}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的使用 Amazon Organizations 管理多个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataLakeOrganizationConfiguration。

create-data-lake

以下代码示例演示了如何使用 `create-data-lake`。

Amazon CLI

示例 1：在多个区域配置数据湖

以下 `create-data-lake` 示例在多个 Amazon 区域启用 Amazon Security Lake 并配置您的数据湖。

```
aws securitylake create-data-lake \
    --configurations '[{"encryptionConfiguration":
 {"kmsKeyId":"S3_MANAGED_KEY"},"region":"us-east-1","lifecycleConfiguration":
 {"expiration":{"days":365},"transitions":
[{"days":60,"storageClass":"ONEZONE_IA"}]}}, {"encryptionConfiguration":
 {"kmsKeyId":"S3_MANAGED_KEY"},"region":"us-east-2","lifecycleConfiguration":
 {"expiration":{"days":365},"transitions":
[{"days":60,"storageClass":"ONEZONE_IA"}]}}]' \
    --meta-store-manager-role-arn "arn:aws:iam:us-east-1:123456789012:role/service-
role/AmazonSecurityLakeMetaStoreManager"
```

输出：

```
{
    "dataLakes": [
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-1:522481757177:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "S3_MANAGED_KEY"
            },
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 365
                },
                "transitions": [
                    {
                        "days": 60,
                        "storageClass": "ONEZONE_IA"
                    }
                ]
            },
            "region": "us-east-1",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-1-
gnevt6s8z7bzby8oi3uiaysbr8v2ml",
            "updateStatus": {
                "exception": {},
                "requestId": "f20a6450-d24a-4f87-a6be-1d4c075a59c2",
                "status": "INITIALIZED"
            }
        },
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-2:522481757177:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "S3_MANAGED_KEY"
            },
```

```
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 365
                },
                "transitions": [
                    {
                        "days": 60,
                        "storageClass": "ONEZONE_IA"
                    }
                ]
            },
            "region": "us-east-2",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-2-
cehuifzl5rwmhm6m62h7zhvtseogr9",
            "updateStatus": {
                "exception": {},
                "requestId": "f20a6450-d24a-4f87-a6be-1d4c075a59c2",
                "status": "INITIALIZED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 入门。

示例 2：在单个区域配置数据湖

以下 `create-data-lake` 示例在单个 Amazon 区域启用 Amazon Security Lake 并配置您的数据湖。

```
aws securitylake create-data-lake \
    --configurations '[{"encryptionConfiguration":
 {"kmsKeyId":"1234abcd-12ab-34cd-56ef-1234567890ab"},"region":"us-
east-2","lifecycleConfiguration": {"expiration":{"days":500},"transitions":
[{"days":30,"storageClass":"GLACIER"}]}}]' \
```

```
    --meta-store-manager-role-arn "arn:aws:iam:us-east-1:123456789012:role/service-role/AmazonSecurityLakeMetaStoreManager"
```

输出：

```
{
    "dataLakes": [
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-2:522481757177:data-lake/default",
            "encryptionConfiguration": {
                "kmsKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
            },
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 500
                },
                "transitions": [
                    {
                        "days": 30,
                        "storageClass": "GLACIER"
                    }
                ]
            },
            "region": "us-east-2",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-2-cehuifzl5rwmhm6m62h7zhvtseogr9",
            "updateStatus": {
                "exception": {},
                "requestId": "77702a53-dcbf-493e-b8ef-518e362f3003",
                "status": "INITIALIZED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDataLake。

create-subscriber-notification

以下代码示例演示了如何使用 create-subscriber-notification。

Amazon CLI

创建订阅用户通知

以下 create-subscriber-notification 示例说明如何指定订阅用户通知，以便在向数据湖写入新数据时创建通知。

```
aws securitylake create-subscriber-notification \
    --subscriber-id "12345ab8-1a34-1c34-1bd4-12345ab9012" \
    --configuration '{"httpsNotificationConfiguration":
 {"targetRoleArn":"arn:aws:iam::XXX:role/service-role/RoleName",
 "endpoint":"https://account-management.$3.$2.securitylake.aws.dev/v1/datalake"}}'
```

输出：

```
{
    "subscriberEndpoint": [
        "https://account-management.$3.$2.securitylake.aws.dev/v1/datalake"
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscriberNotification。

create-subscriber

以下代码示例演示了如何使用 create-subscriber。

Amazon CLI

示例 1：创建具有数据访问权限的订阅用户

以下 create-subscriber 示例在 Security Lake 中创建一个订阅用户，该订阅用户可以访问当前 Amazon 区域中为 Amazon 源指定的订阅用户身份数据。

```
aws securitylake create-subscriber \
    --access-types "S3" \
    --sources '[{"awsLogSource": {"sourceName": "VPC_FLOW","sourceVersion":
 "2.0"}}]' \
    --subscriber-name 'opensearch-s3' \
    --subscriber-identity '{"principal": "029189416600","externalId":
 "123456789012"}'
```

输出：

```
{
    "subscriber": {
        "accessTypes": [
            "S3"
        ],
        "createdAt": "2024-07-17T19:08:26.787000+00:00",
        "roleArn": "arn:aws:iam::773172568199:role/AmazonSecurityLake-896f218b-
cfba-40be-a255-8b49a65d0407",
        "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-1-
um632ufwpvxkyz0bc5hkb64atycnf3",
        "sources": [
            {
                "awsLogSource": {
                    "sourceName": "VPC_FLOW",
                    "sourceVersion": "2.0"
                }
            }
        ],
        "subscriberArn": "arn:aws:securitylake:us-
east-1:773172568199:subscriber/896f218b-cfba-40be-a255-8b49a65d0407",
        "subscriberId": "896f218b-cfba-40be-a255-8b49a65d0407",
        "subscriberIdentity": {
            "externalId": "123456789012",
            "principal": "029189416600"
        },
        "subscriberName": "opensearch-s3",
        "subscriberStatus": "ACTIVE",
        "updatedAt": "2024-07-17T19:08:27.133000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的创建具有数据访问权限的订阅用户。

示例 2：创建具有查询访问权限的订阅用户

以下 `create-subscriber` 示例在 Security Lake 中为指定的订阅用户身份创建一个在当前 Amazon 区域具有查询访问权限的订阅用户。

```
aws securitylake create-subscriber \
    --access-types "LAKEFORMATION" \
    --sources '[{"awsLogSource": {"sourceName": "VPC_FLOW","sourceVersion":
 "2.0"}}]' \
    --subscriber-name 'opensearch-s3' \
    --subscriber-identity '{"principal": "029189416600","externalId":
 "123456789012"}'
```

输出：

```
{
    "subscriber": {
        "accessTypes": [
            "LAKEFORMATION"
        ],
        "createdAt": "2024-07-18T01:05:55.853000+00:00",
        "resourceShareArn": "arn:aws:ram:us-east-1:123456789012:resource-
share/8c31da49-c224-4f1e-bb12-37ab756d6d8a",
        "resourceShareName": "LakeFormation-V2-NAMENAMENA-123456789012",
        "sources": [
            {
                "awsLogSource": {
                    "sourceName": "VPC_FLOW",
                    "sourceVersion": "2.0"
                }
            }
        ],
        "subscriberArn": "arn:aws:securitylake:us-east-1:123456789012:subscriber/
e762aabb-ce3d-4585-beab-63474597845d",
        "subscriberId": "e762aabb-ce3d-4585-beab-63474597845d",
        "subscriberIdentity": {
            "externalId": "123456789012",
            "principal": "029189416600"
        },
        "subscriberName": "opensearch-s3",
```

```
        "subscriberStatus": "ACTIVE",
        "updatedAt": "2024-07-18T01:05:58.393000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的创建具有查询访问权限的订阅用户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscriber。

delete-aws-log-source

以下代码示例演示了如何使用 delete-aws-log-source。

Amazon CLI

移除原生支持的 Amazon 服务

以下 delete-aws-logsource 示例在指定账户和区域中删除 VPC 流日志作为 Security Lake 源。

```
aws securitylake delete-aws-log-source \
    --sources '[{"regions": ["us-east-1"], "accounts": ["123456789012"],
 "sourceName": "SH_FINDINGS", "sourceVersion": "2.0"}]'
```

输出：

```
{
    "failed": [
        "123456789012"
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的移除 Amazon 服务作为源。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteAwsLogSource。

delete-custom-log-source

以下代码示例演示了如何使用 delete-custom-log-source。

Amazon CLI

移除自定义源

以下 delete-custom-logsource 示例在指定日志提供商账户和指定区域中删除自定义源。

```
aws securitylake delete-custom-log-source \
    --source-name "CustomSourceName"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的删除自定义源。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteCustomLogSource。

delete-data-lake-organization-configuration

以下代码示例演示了如何使用 delete-data-lake-organization-configuration。

Amazon CLI

停止在成员账户中自动收集源

以下 delete-data-lake-organization-configuration 示例停止从加入组织的新成员账户中自动收集 Amazon Security Hub 调查发现。只有委托的 Security Lake 管理员才能运行此命令。它可以防止新成员账户自动向数据湖提供数据。

```
aws securitylake delete-data-lake-organization-configuration \
    --auto-enable-new-account '[{"region":"us-east-1","sources":
[{"sourceName":"SH_FINDINGS"}]}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的使用 Amazon Organizations 管理多个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDataLakeOrganizationConfiguration。

delete-data-lake

以下代码示例演示了如何使用 delete-data-lake。

Amazon CLI

禁用您的数据湖

以下 delete-data-lake 示例在指定 Amazon 区域禁用您的数据湖。在指定的区域中，源不再向数据湖提供数据。对于利用 Amazon Organizations 的 Security Lake 部署，只有该组织的委托 Security Lake 管理员才能为组织中的账户禁用 Security Lake。

```
aws securitylake delete-data-lake \
    --regions "ap-northeast-1" "eu-central-1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的禁用 Amazon Security Lake。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDataLake。

delete-subscriber-notification

以下代码示例演示了如何使用 delete-subscriber-notification。

Amazon CLI

删除订阅用户通知

以下 delete-subscriber-notification 示例说明如何删除特定 Security Lake 订阅用户的订阅用户通知。

```
aws securitylake delete-subscriber-notification \
    --subscriber-id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubscriberNotification。

delete-subscriber

以下代码示例演示了如何使用 delete-subscriber。

Amazon CLI

删除订阅用户

以下 delete-subscriber 示例说明如果您不再希望某个订阅用户访问 Security Lake 中的数据，如何移除该订阅用户。

```
aws securitylake delete-subscriber \
    --subscriber-id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSubscriber。

get-data-lake-exception-subscription

以下代码示例演示了如何使用 get-data-lake-exception-subscription。

Amazon CLI

获取有关异常订阅的详细信息

以下 get-data-lake-exception-subscription 示例提供有关 Security Lake 异常订阅的详细信息。在此示例中，指定 Amazon 账户的用户会通过短信收到错误通知。异常消息将在指定时间段内保留在账户中。异常订阅会通过请求者的首选协议向 Security Lake 用户通报错误。

```
aws securitylake get-data-lake-exception-subscription
```

输出：

```
{
    "exceptionTimeToLive": 30,
    "notificationEndpoint": "123456789012",
    "subscriptionProtocol": "sms"
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的数据湖状态故障排除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataLakeExceptionSubscription。

get-data-lake-organization-configuration

以下代码示例演示了如何使用 get-data-lake-organization-configuration。

Amazon CLI

获取有关新组织账户配置的详细信息

以下 get-data-lake-organization-configuration 示例检索有关新组织账户在加入 Amazon Security Lake 后将发送的源日志的详细信息。

```
aws securitylake get-data-lake-organization-configuration
```

输出：

```
{
    "autoEnableNewAccount": [
        {
            "region": "us-east-1",
            "sources": [
                {
                    "sourceName": "VPC_FLOW",
                    "sourceVersion": "1.0"
                },
                {
                    "sourceName": "ROUTE53",
                    "sourceVersion": "1.0"
                },
                {
                    "sourceName": "SH_FINDINGS",
                    "sourceVersion": "1.0"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的使用 Amazon Organizations 管理多个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataLakeOrganizationConfiguration。

get-data-lake-sources

以下代码示例演示了如何使用 get-data-lake-sources。

Amazon CLI

获取日志收集的状态

以下 get-data-lake-sources 示例获取当前 Amazon 区域中指定账户的日志收集快照。该账户已启用 Amazon Security Lake。

```
aws securitylake get-data-lake-sources \
    --accounts "123456789012"
```

输出：

```
{
    "dataLakeSources": [
        {
            "account": "123456789012",
            "sourceName": "SH_FINDINGS",
            "sourceStatuses": [
                {
                    "resource": "vpc-1234567890abcdef0",
                    "status": "COLLECTING"
                }
            ]
        },
        {
            "account": "123456789012",
            "sourceName": "VPC_FLOW",
            "sourceStatuses": [
                {
                    "resource": "vpc-1234567890abcdef0",
                    "status": "NOT_COLLECTING"
                }
```

```
            ]
        },
        {
            "account": "123456789012",
            "sourceName": "LAMBDA_EXECUTION",
            "sourceStatuses": [
                {
                    "resource": "vpc-1234567890abcdef0",
                    "status": "COLLECTING"
                }
            ]
        },
        {
            "account": "123456789012",
            "sourceName": "ROUTE53",
            "sourceStatuses": [
                {
                    "resource": "vpc-1234567890abcdef0",
                    "status": "COLLECTING"
                }
            ]
        },
        {
            "account": "123456789012",
            "sourceName": "CLOUD_TRAIL_MGMT",
            "sourceStatuses": [
                {
                    "resource": "vpc-1234567890abcdef0",
                    "status": "COLLECTING"
                }
            ]
        }
    ],
    "dataLakeArn": null
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的从 Amazon 服务收集数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDataLakeSources。

`get-subscriber`

以下代码示例演示了如何使用 `get-subscriber`。

Amazon CLI

检索订阅信息

以下 get-subscriber 示例检索指定 Securiy Lake 订阅用户的订阅信息。

```
aws securitylake get-subscriber \
    --subscriber-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "subscriber": {
        "accessTypes": [
            "LAKEFORMATION"
        ],
        "createdAt": "2024-04-19T15:19:44.421803+00:00",
        "resourceShareArn": "arn:aws:ram:eu-west-2:123456789012:resource-share/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "resourceShareName": "LakeFormation-V3-TKJGBHCKTZ-123456789012",
        "sources": [
            {
                "awsLogSource": {
                    "sourceName": "LAMBDA_EXECUTION",
                    "sourceVersion": "1.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "EKS_AUDIT",
                    "sourceVersion": "2.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "ROUTE53",
                    "sourceVersion": "1.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "SH_FINDINGS",
                    "sourceVersion": "1.0"
```

```
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "VPC_FLOW",
                    "sourceVersion": "1.0"
                }
            },
            {
                "customLogSource": {
                    "attributes": {
                        "crawlerArn": "arn:aws:glue:eu-west-2:123456789012:crawler/
testCustom2",
                        "databaseArn": "arn:aws:glue:eu-
west-2:123456789012:database/amazon_security_lake_glue_db_eu_west_2",
                        "tableArn": "arn:aws:glue:eu-west-2:123456789012:table/
amazon_security_lake_table_eu_west_2_ext_testcustom2"
                    },
                    "provider": {
                        "location": "s3://aws-security-data-lake-eu-
west-2-8ugsus4ztnsfpjbldwbgf4vge98av9/ext/testCustom2/",
                        "roleArn": "arn:aws:iam::123456789012:role/
AmazonSecurityLake-Provider-testCustom2-eu-west-2"
                    },
                    "sourceName": "testCustom2"
                }
            },
            {
                "customLogSource": {
                    "attributes": {
                        "crawlerArn": "arn:aws:glue:eu-west-2:123456789012:crawler/
TestCustom",
                        "databaseArn": "arn:aws:glue:eu-
west-2:123456789012:database/amazon_security_lake_glue_db_eu_west_2",
                        "tableArn": "arn:aws:glue:eu-west-2:123456789012:table/
amazon_security_lake_table_eu_west_2_ext_testcustom"
                    },
                    "provider": {
                        "location": "s3://aws-security-data-lake-eu-
west-2-8ugsus4ztnsfpjbldwbgf4vge98av9/ext/TestCustom/",
                        "roleArn": "arn:aws:iam::123456789012:role/
AmazonSecurityLake-Provider-TestCustom-eu-west-2"
                    },
                    "sourceName": "TestCustom"
```

```
                }
            }
        ],
        "subscriberArn": "arn:aws:securitylake:eu-west-2:123456789012:subscriber/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "subscriberId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "subscriberIdentity": {
            "externalId": "123456789012",
            "principal": "123456789012"
        },
        "subscriberName": "test",
        "subscriberStatus": "ACTIVE",
        "updatedAt": "2024-04-19T15:19:55.230588+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubscriber。

list-data-lake-exceptions

以下代码示例演示了如何使用 list-data-lake-exceptions。

Amazon CLI

列出影响您数据湖的问题

以下 list-data-lake-exceptions 示例列出在过去 14 天内在指定 Amazon 区域中影响您数据湖的问题。

```
aws securitylake list-data-lake-exceptions \
    --regions "us-east-1" "eu-west-3"
```

输出：

```
{
    "exceptions": [
        {
            "exception": "The account does not have the required role permissions.
 Update your role permissions to use the new data source version.",
            "region": "us-east-1",
```

```
            "timestamp": "2024-02-29T12:24:15.641725+00:00"
        },
        {
            "exception": "The account does not have the required role permissions.
 Update your role permissions to use the new data source version.",
            "region": "eu-west-3",
            "timestamp": "2024-02-29T12:24:15.641725+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 故障排除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDataLakeExceptions。

`list-data-lakes`

以下代码示例演示了如何使用 `list-data-lakes`。

Amazon CLI

列出 Security Lake 配置对象

以下 `list-data-lakes` 示例列出指定 Amazon 区域的 Amazon Security Lake 配置对象。您可以使用此命令来确定是否在指定区域中启用了 Security Lake。

```
aws securitylake list-data-lakes \
    --regions "us-east-1"
```

输出：

```
{
    "dataLakes": [
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-1:123456789012:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "S3_MANAGED_KEY"
            },
            "lifecycleConfiguration": {
```

```
                "expiration": {
                    "days": 365
                },
                "transitions": [
                    {
                        "days": 60,
                        "storageClass": "ONEZONE_IA"
                    }
                ]
            },
            "region": "us-east-1",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:123456789012:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-
east-1-1234567890abcdef0",
            "updateStatus": {
                "exception": {
                    "code": "software.amazon.awssdk.services.s3.model.S3Exception",
                    "reason": ""
                },
                "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "status": "FAILED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的检查区域状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDataLakes。

list-log-sources

以下代码示例演示了如何使用 list-log-sources。

Amazon CLI

检索 Amazon Security Lake 日志源

以下 `list-log-sources` 示例列出指定账户中的 Amazon Security Lake 日志源。

```
aws securitylake list-log-sources \
    --accounts "123456789012"
```

输出：

```
{
    "account": "123456789012",
    "region": "xy-region-1",
    "sources": [
        {
               "awsLogSource": {
                "sourceName": "VPC_FLOW",
                "sourceVersion": "2.0"
            }
        },
        {
            "awsLogSource": {
                "sourceName": "SH_FINDINGS",
                "sourceVersion": "2.0"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的源管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLogSources。

`list-subscribers`

以下代码示例演示了如何使用 `list-subscribers`。

Amazon CLI

检索 Amazon Security Lake 订阅用户

以下 `list-subscribers` 示例列出指定账户中的所有 Amazon Security Lake 订阅用户。

```
aws securitylake list-subscribers
```

输出：

```
{
    "subscribers": [
        {
            "accessTypes": [
                "S3"
            ],
            "createdAt": "2024-06-04T15:02:28.921000+00:00",
            "roleArn": "arn:aws:iam::123456789012:role/AmazonSecurityLake-
E1WG1ZNPRXT0D4",
            "s3BucketArn": "amzn-s3-demo-bucket--usw2-az1--x-s3",
            "sources": [
                {
                    "awsLogSource": {
                        "sourceName": "CLOUD_TRAIL_MGMT",
                        "sourceVersion": "2.0"
                    }
                },
                {
                    "awsLogSource": {
                        "sourceName": "LAMBDA_EXECUTION",
                        "sourceVersion": "1.0"
                    }
                },
                {
                    "customLogSource": {
                        "attributes": {
                            "crawlerArn": "arn:aws:glue:eu-
west-2:123456789012:crawler/E1WG1ZNPRXT0D4",
                            "databaseArn": "arn:aws:glue:eu-
west-2:123456789012:database/E1WG1ZNPRXT0D4",
                            "tableArn": "arn:aws:glue:eu-west-2:123456789012:table/
E1WG1ZNPRXT0D4"
                        },
                        "provider": {
                            "location": "amzn-s3-demo-bucket--usw2-az1--x-s3",
                            "roleArn": "arn:aws:iam::123456789012:role/
AmazonSecurityLake-E1WG1ZNPRXT0D4"
                        },
                        "sourceName": "testCustom2"
                    }
                }
            ],
```

```
            "subscriberArn": "arn:aws:securitylake:eu-
west-2:123456789012:subscriber/E1WG1ZNPRXT0D4",
            "subscriberEndpoint": "arn:aws:sqs:eu-
west-2:123456789012:AmazonSecurityLake-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111-Main-
Queue",
            "subscriberId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "subscriberIdentity": {
                "externalId": "ext123456789012",
                "principal": "123456789012"
            },
            "subscriberName": "Test",
            "subscriberStatus": "ACTIVE",
            "updatedAt": "2024-06-04T15:02:35.617000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSubscribers。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出现有资源的标签

以下 `list-tags-for-resource` 示例列出指定 Amazon Security Lake 订阅用户的标签。在此示例中，“Owner”标签键没有关联的标签值。您也可以使用此操作列出其他现有 Security Lake 资源的标签。

```
aws securitylake list-tags-for-resource \
    --resource-arn "arn:aws:securitylake:us-
east-1:123456789012:subscriber/1234abcd-12ab-34cd-56ef-1234567890ab"
```

输出：

```
{
    "tags": [
```

```
        {
            "key": "Environment",
            "value": "Cloud"
        },
        {
            "key": "CostCenter",
            "value": "12345"
        },
        {
            "key": "Owner",
            "value": ""
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的标记 Amazon Security Lake 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`register-data-lake-delegated-administrator`

以下代码示例演示了如何使用 `register-data-lake-delegated-administrator`。

Amazon CLI

指定委托的管理员

以下 `register-data-lake-delegated-administrator` 示例会将指定 Amazon 账户指定为委托的 Amazon Security Lake 管理员。

```
aws securitylake register-data-lake-delegated-administrator \
    --account-id 123456789012
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的使用 Amazon Organizations 管理多个账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterDataLakeDelegatedAdministrator。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向现有资源添加标签

以下 `tag-resource` 示例向现有订阅用户资源添加标签。要创建新资源并为其添加一个或多个标签，请不要使用此操作。相反，请针对要创建的资源类型使用相应的“Create”操作。

```
aws securitylake tag-resource \
    --resource-arn "arn:aws:securitylake:us-
east-1:123456789012:subscriber/1234abcd-12ab-34cd-56ef-1234567890ab" \
    --tags key=Environment,value=Cloud
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的标记 Amazon Security Lake 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从现有资源中移除标签

以下 `untag-resource` 示例从现有订阅用户资源中移除指定标签。

```
aws securitylake untag-resource \
    --resource-arn "arn:aws:securitylake:us-
east-1:123456789012:subscriber/1234abcd-12ab-34cd-56ef-1234567890ab" \
    --tags Environment Owner
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的标记 Amazon Security Lake 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-data-lake-exception-subscription

以下代码示例演示了如何使用 update-data-lake-exception-subscription。

Amazon CLI

更新 Security Lake 异常的通知订阅

以下 update-data-lake-exception-subscription 示例更新通知用户 Security Lake 异常的通知订阅。

```
aws securitylake update-data-lake-exception-subscription \
    --notification-endpoint "123456789012" \
    --exception-time-to-live 30 \
    --subscription-protocol "email"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 故障排除。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDataLakeExceptionSubscription。

update-data-lake

以下代码示例演示了如何使用 update-data-lake。

Amazon CLI

示例 1：更新您的数据湖设置

以下 update-data-lake 示例更新您的 Amazon Security Lake 数据湖的设置。您可以使用此操作来指定数据加密、存储和汇总区域设置。

```
aws securitylake update-data-lake \
    --configurations '[{"encryptionConfiguration":
 {"kmsKeyId":"S3_MANAGED_KEY"},"region":"us-east-1","lifecycleConfiguration":
 {"expiration":{"days":365},"transitions":
```

```
[{"days":60,"storageClass":"ONEZONE_IA"}]}}, {"encryptionConfiguration":
 {"kmsKeyId":"S3_MANAGED_KEY"},"region":"us-east-2","lifecycleConfiguration":
 {"expiration":{"days":365},"transitions":
[{"days":60,"storageClass":"ONEZONE_IA"}]}}]' \
    --meta-store-manager-role-arn "arn:aws:iam:us-east-1:123456789012:role/service-
role/AmazonSecurityLakeMetaStoreManager"
```

输出：

```
{
    "dataLakes": [
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-1:522481757177:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "S3_MANAGED_KEY"
            },
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 365
                },
                "transitions": [
                    {
                        "days": 60,
                        "storageClass": "ONEZONE_IA"
                    }
                ]
            },
            "region": "us-east-1",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-1-
gnevt6s8z7bzby8oi3uiaysbr8v2ml",
            "updateStatus": {
                "exception": {},
                "requestId": "f20a6450-d24a-4f87-a6be-1d4c075a59c2",
                "status": "INITIALIZED"
```

```
            }
        },
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-2:522481757177:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "S3_MANAGED_KEY"
            },
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 365
                },
                "transitions": [
                    {
                        "days": 60,
                        "storageClass": "ONEZONE_IA"
                    }
                ]
            },
            "region": "us-east-2",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-2-
cehuifzl5rwmhm6m62h7zhvtseogr9",
            "updateStatus": {
                "exception": {},
                "requestId": "f20a6450-d24a-4f87-a6be-1d4c075a59c2",
                "status": "INITIALIZED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 入门。

示例 2：在单个区域配置数据湖

以下 create-data-lake 示例在单个 Amazon 区域启用 Amazon Security Lake 并配置您的数据湖。

```
aws securitylake create-data-lake \
    --configurations '[{"encryptionConfiguration":
 {"kmsKeyId":"1234abcd-12ab-34cd-56ef-1234567890ab"},"region":"us-
east-2","lifecycleConfiguration": {"expiration":{"days":500},"transitions":
[{"days":30,"storageClass":"GLACIER"}]}}]' \
    --meta-store-manager-role-arn "arn:aws:iam:us-east-1:123456789012:role/service-
role/AmazonSecurityLakeMetaStoreManager"
```

输出：

```
{
    "dataLakes": [
        {
            "createStatus": "COMPLETED",
            "dataLakeArn": "arn:aws:securitylake:us-east-2:522481757177:data-lake/
default",
            "encryptionConfiguration": {
                "kmsKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
            },
            "lifecycleConfiguration": {
                "expiration": {
                    "days": 500
                },
                "transitions": [
                    {
                        "days": 30,
                        "storageClass": "GLACIER"
                    }
                ]
            },
            "region": "us-east-2",
            "replicationConfiguration": {
                "regions": [
                    "ap-northeast-3"
                ],
                "roleArn": "arn:aws:securitylake:ap-northeast-3:522481757177:data-
lake/default"
            },
            "s3BucketArn": "arn:aws:s3:::aws-security-data-lake-us-east-2-
cehuifzl5rwmhm6m62h7zhvtseogr9",
```

```
            "updateStatus": {
                "exception": {},
                "requestId": "77702a53-dcbf-493e-b8ef-518e362f3003",
                "status": "INITIALIZED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的 Amazon Security Lake 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDataLake。

update-subscriber-notification

以下代码示例演示了如何使用 update-subscriber-notification。

Amazon CLI

更新订阅用户通知

以下 update-subscriber-notification 示例说明如何更新订阅用户的通知。

```
aws securitylake update-subscriber-notification \
    --subscriber-id "12345ab8-1a34-1c34-1bd4-12345ab9012" \
    --configuration '{"httpsNotificationConfiguration":
 {"targetRoleArn":"arn:aws:iam::XXX:role/service-role/RoleName",
 "endpoint":"https://account-management.$3.$2.securitylake.aws.dev/v1/datalake"}}'
```

输出：

```
{
    "subscriberEndpoint": [
        "https://account-management.$3.$2.securitylake.aws.dev/v1/datalake"
    ]
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscriberNotification。

update-subscriber

以下代码示例演示了如何使用 update-subscriber。

Amazon CLI

更新 Amazon Security Lake 订阅用户

以下 update-subscriber 示例更新特定 Security Lake 订阅用户的安全湖数据访问源。

```
aws securitylake update-subscriber \
    --subscriber-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "subscriber": {
        "accessTypes": [
            "LAKEFORMATION"
        ],
        "createdAt": "2024-04-19T15:19:44.421803+00:00",
        "resourceShareArn": "arn:aws:ram:eu-west-2:123456789012:resource-share/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "resourceShareName": "LakeFormation-V3-TKJGBHCKTZ-123456789012",
        "sources": [
            {
                "awsLogSource": {
                    "sourceName": "LAMBDA_EXECUTION",
                    "sourceVersion": "1.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "EKS_AUDIT",
                    "sourceVersion": "2.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "ROUTE53",
                    "sourceVersion": "1.0"
                }
            },
            {
```

```
                "awsLogSource": {
                    "sourceName": "SH_FINDINGS",
                    "sourceVersion": "1.0"
                }
            },
            {
                "awsLogSource": {
                    "sourceName": "VPC_FLOW",
                    "sourceVersion": "1.0"
                }
            },
            {
                "customLogSource": {
                    "attributes": {
                        "crawlerArn": "arn:aws:glue:eu-west-2:123456789012:crawler/
E1WG1ZNPRXT0D4",
                        "databaseArn": "arn:aws:glue:eu-
west-2:123456789012:database/E1WG1ZNPRXT0D4",
                        "tableArn": "arn:aws:glue:eu-west-2:123456789012:table/
E1WG1ZNPRXT0D4"
                    },
                    "provider": {
                        "location": "amzn-s3-demo-bucket--usw2-az1--x-s3",
                        "roleArn": "arn:aws:iam::123456789012:role/
AmazonSecurityLake-E1WG1ZNPRXT0D4"
                    },
                    "sourceName": "testCustom2"
                }
            }
        ],
        "subscriberArn": "arn:aws:securitylake:eu-west-2:123456789012:subscriber/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "subscriberId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "subscriberIdentity": {
            "externalId": "123456789012",
            "principal": "123456789012"
        },
        "subscriberName": "test",
        "subscriberStatus": "ACTIVE",
        "updatedAt": "2024-07-18T20:47:37.098000+00:00"
    }
}
```

有关更多信息，请参阅《Amazon Security Lake 用户指南》中的订阅用户管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscriber。

Amazon Serverless Application Repository使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Serverless Application Repository 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`put-application-policy`

以下代码示例演示了如何使用 `put-application-policy`。

Amazon CLI

示例 1：公开共享应用程序

以下 `put-application-policy` 将公开共享应用程序，让任何人都可以在 Amazon Serverless Application Repository 中找到并部署您的应用程序。

```
aws serverlessrepo put-application-policy \
    --application-id arn:aws:serverlessrepo:us-east-1:123456789012:applications/my-test-application \
    --statements Principals='*',Actions=Deploy
```

输出：

```
{
    "Statements": [
        {
```

```
            "Actions": [
                "Deploy"
            ],
            "Principals": [
                ""
            ],
            "StatementId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
        }
    ]
}
```

示例 2：私下共享应用程序

以下 put-application-policy 将私下共享应用程序，只有特定 Amazon 账户才可以在 Amazon Serverless Application Repository 中找到并部署应用程序。

```
aws serverlessrepo put-application-policy \
    --application-id arn:aws:serverlessrepo:us-east-1:123456789012:applications/my-test-application \
    --statements Principals=111111111111,222222222222,Actions=Deploy
```

输出：

```
{
    "Statements": [
        {
            "Actions": [
                "Deploy"
            ],
            "Principals": [
                "111111111111",
                "222222222222"
            ],
            "StatementId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Serverless Application Repository 开发人员指南》中的通过控制台共享应用程序。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutApplicationPolicy。

使用 Amazon CLI 的服务目录示例

以下代码示例演示如何通过将 Amazon Command Line Interface与服务目录结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`accept-portfolio-share`

以下代码示例演示了如何使用 `accept-portfolio-share`。

Amazon CLI

接受共享产品组合

以下 `accept-portfolio-share` 示例接受其他用户提出的共享指定产品组合的提议。

```
aws servicecatalog accept-portfolio-share \
    --portfolio-id port-2s6wuabcdefghijk
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AcceptPortfolioShare。

`associate-principal-with-portfolio`

以下代码示例演示了如何使用 `associate-principal-with-portfolio`。

Amazon CLI

将委托人与产品组合相关联

以下 associate-principal-with-portfolio 示例会将用户与特定产品组合相关联。

```
aws servicecatalog associate-principal-with-portfolio \
    --portfolio-id port-2s6abcdefwdh4 \
    --principal-arn arn:aws:iam::123456789012:user/usertest \
    --principal-type IAM
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociatePrincipalWithPortfolio。

associate-product-with-portfolio

以下代码示例演示了如何使用 associate-product-with-portfolio。

Amazon CLI

将产品与产品组合相关联

以下 associate-product-with-portfolio 示例会将指定产品与特定产品组合相关联。

```
aws servicecatalog associate-product-with-portfolio
     --product-id prod-3p5abcdef3oyk
     --portfolio-id port-2s6abcdef5wdh4
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateProductWithPortfolio。

associate-tag-option-with-resource

以下代码示例演示了如何使用 associate-tag-option-with-resource。

Amazon CLI

将 TagOption 与资源相关联

以下 associate-tag-option-with-resource 示例会将指定 TagOption 与指定资源相关联。

```
aws servicecatalog associate-tag-option-with-resource \
    --resource-id port-2s6abcdq5wdh4 \
    --tag-option-id tag-p3abc2pkpz5qc
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateTagOptionWithResource。

copy-product

以下代码示例演示了如何使用 copy-product。

Amazon CLI

复制产品

以下 copy-product 示例使用 JSON 文件传递参数，创建指定产品的副本。

```
aws servicecatalog copy-product --cli-input-json file://copy-product-input.json
```

的内容copy-product-input.json：

```
{
    "SourceProductArn": "arn:aws:catalog:us-west-2:123456789012:product/prod-
tcabcd3syn2xy",
    "TargetProductName": "copy-of-myproduct",
    "CopyOptions": [
        "CopyTags"
    ]
}
```

输出：

```
{
    "CopyProductToken": "copyproduct-abc5defgjkdji"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyProduct。

create-portfolio-share

以下代码示例演示了如何使用 create-portfolio-share。

Amazon CLI

与账户共享产品组合

以下 create-portfolio-share 示例与指定账户共享指定产品组合。

```
aws servicecatalog create-portfolio-share \
    --portfolio-id port-2s6abcdef5wdh4 \
    --account-id 794123456789
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePortfolioShare。

create-portfolio

以下代码示例演示了如何使用 create-portfolio。

Amazon CLI

创建产品组合

以下 create-portfolio 示例创建产品组合。

```
aws servicecatalog create-portfolio  \
    --provider-name my-provider \
    --display-name my-portfolio
```

输出：

```
{
    "PortfolioDetail": {
        "ProviderName": "my-provider",
        "DisplayName": "my-portfolio",
        "CreatedTime": 1571337221.555,
        "ARN": "arn:aws:catalog:us-east-2:123456789012:portfolio/
port-2s6xmplq5wdh4",
        "Id": "port-2s6xmplq5wdh4"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePortfolio。

create-product

以下代码示例演示了如何使用 create-product。

Amazon CLI

创建产品

以下 create-product 示例使用 JSON 文件传递参数，创建产品。

```
aws servicecatalog create-product \
    --cli-input-json file://create-product-input.json
```

的内容create-product-input.json：

```
{
    "AcceptLanguage": "en",
    "Name": "test-product",
    "Owner": "test-owner",
    "Description": "test-description",
    "Distributor": "test-distributor",
    "SupportDescription": "test-support",
    "SupportEmail": "test@amazon.com",
    "SupportUrl": "https://aws.amazon.com",
    "ProductType": "CLOUD_FORMATION_TEMPLATE",
    "Tags": [
        {
            "Key": "region",
            "Value": "us-east-1"
        }
    ],
    "ProvisioningArtifactParameters": {
        "Name": "test-version-name",
        "Description": "test-version-description",
        "Info": {
            "LoadTemplateFromURL": "https://s3-us-west-1.amazonaws.com/
cloudformation-templates-us-west-1/my-cfn-template.template"
        },
        "Type": "CLOUD_FORMATION_TEMPLATE"
    }
}
```

输出：

```
{
    "Tags": [
```

```
        {
            "Key": "region",
            "Value": "us-east-1"
        }
    ],
    "ProductViewDetail": {
        "CreatedTime": 1576025036.0,
        "ProductARN": "arn:aws:catalog:us-west-2:1234568542028:product/
prod-3p5abcdef3oyk",
        "Status": "CREATED",
        "ProductViewSummary": {
            "Type": "CLOUD_FORMATION_TEMPLATE",
            "Distributor": "test-distributor",
            "SupportUrl": "https://aws.amazon.com",
            "SupportEmail": "test@amazon.com",
            "Id": "prodview-abcd42wvx45um",
            "SupportDescription": "test-support",
            "ShortDescription": "test-description",
            "Owner": "test-owner",
            "Name": "test-product2",
            "HasDefaultPath": false,
            "ProductId": "prod-3p5abcdef3oyk"
        }
    },
    "ProvisioningArtifactDetail": {
        "CreatedTime": 1576025036.0,
        "Active": true,
        "Id": "pa-pq3p5lil12a34",
        "Description": "test-version-description",
        "Name": "test-version-name",
        "Type": "CLOUD_FORMATION_TEMPLATE"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProduct。

`create-provisioning-artifact`

以下代码示例演示了如何使用 `create-provisioning-artifact`。

Amazon CLI

创建预置构件

以下 create-provisioning-artifact 示例使用 JSON 文件传递参数，创建预置构件。

```
aws servicecatalog create-provisioning-artifact \
   --cli-input-json file://create-provisioning-artifact-input.json
```

的内容create-provisioning-artifact-input.json：

```
{
    "ProductId": "prod-nfi2abcdefghi",
    "Parameters": {
        "Name": "test-provisioning-artifact",
        "Description": "test description",
        "Info": {
            "LoadTemplateFromURL": "https://s3-us-west-1.amazonaws.com/
cloudformation-templates-us-west-1/my-cfn-template.template"
        },
        "Type": "CLOUD_FORMATION_TEMPLATE"
    }
}
```

输出：

```
{
    "Info": {
        "TemplateUrl": "https://s3-us-west-1.amazonaws.com/cloudformation-templates-
us-west-1/my-cfn-template.template"
    },
    "Status": "CREATING",
    "ProvisioningArtifactDetail": {
        "Id": "pa-bb4abcdefwnaio",
        "Name": "test-provisioning-artifact",
        "Description": "test description",
        "Active": true,
        "Type": "CLOUD_FORMATION_TEMPLATE",
        "CreatedTime": 1576022545.0
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProvisioningArtifact。

create-tag-option

以下代码示例演示了如何使用 create-tag-option。

Amazon CLI

创建 TagOption

以下 create-tag-option 示例创建 TagOption。

```
aws servicecatalog create-tag-option
    --key 1234
    --value name
```

输出：

```
{
    "TagOptionDetail": {
    "Id": "tag-iabcdn4fzjjms",
    "Value": "name",
    "Active": true,
    "Key": "1234"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTagOption。

delete-portfolio-share

以下代码示例演示了如何使用 delete-portfolio-share。

Amazon CLI

停止与账户共享产品组合

以下 delete-portfolio-share 示例停止与指定账户共享产品组合。

```
aws servicecatalog delete-portfolio-share \
    --portfolio-id port-2s6abcdq5wdh4 \
    --account-id 123456789012
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePortfolioShare。

delete-portfolio

以下代码示例演示了如何使用 delete-portfolio。

Amazon CLI

删除产品组合

以下 delete-portfolio 示例删除指定产品组合。

```
aws servicecatalog delete-portfolio \
    --id port-abcdlx4gox4do
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePortfolio。

delete-product

以下代码示例演示了如何使用 delete-product。

Amazon CLI

删除产品

以下 delete-product 示例删除指定产品。

```
aws servicecatalog delete-product \
    --id prod-abcdcek6yhbxi
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProduct。

delete-provisioning-artifact

以下代码示例演示了如何使用 delete-provisioning-artifact。

Amazon CLI

删除预置构件

以下 delete-provisioning-artifact 示例删除指定预置构件。

```
aws servicecatalog delete-provisioning-artifact \
    --product-id prod-abc2uebuplcpw \
    --provisioning-artifact-id pa-pqabcddii7ouc
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProvisioningArtifact。

delete-tag-option

以下代码示例演示了如何使用 delete-tag-option。

Amazon CLI

删除 TagOption

以下 delete-tag-option 示例删除指定 TagOption。

```
aws servicecatalog delete-tag-option \
    --id tag-iabcdn4fzjjms
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTagOption。

describe-copy-product-status

以下代码示例演示了如何使用 describe-copy-product-status。

Amazon CLI

描述复制产品操作的状态

以下 describe-copy-product-status 示例显示指定异步复制产品操作的当前状态。

```
aws servicecatalog describe-copy-product-status \
```

```
    --copy-product-token copyproduct-znn5tf5abcd3w
```

输出：

```
{
    "CopyProductStatus": "SUCCEEDED",
    "TargetProductId": "prod-os6hog7abcdt2"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCopyProductStatus。

describe-portfolio

以下代码示例演示了如何使用 describe-portfolio。

Amazon CLI

描述产品组合

以下 describe-portfolio 示例显示指定产品组合的详细信息。

```
aws servicecatalog describe-portfolio \
    --id port-2s6abcdq5wdh4
```

输出：

```
{
    "TagOptions": [],
    "PortfolioDetail": {
        "ARN": "arn:aws:catalog:us-west-2:687558541234:portfolio/
port-2s6abcdq5wdh4",
        "Id": "port-2s6wuzyq5wdh4",
        "CreatedTime": 1571337221.555,
        "DisplayName": "my-portfolio",
        "ProviderName": "my-provider"
    },
    "Tags": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePortfolio。

describe-product-as-admin

以下代码示例演示了如何使用 describe-product-as-admin。

Amazon CLI

以管理员身份描述产品

以下 describe-product-as-admin 示例使用管理员权限显示指定产品的详细信息。

```
aws servicecatalog describe-product-as-admin \
    --id prod-abcdcek6yhbxi
```

输出：

```
{
    "TagOptions": [],
    "ProductViewDetail": {
        "ProductARN": "arn:aws:catalog:us-west-2:687558542028:product/prod-
abcdcek6yhbxi",
        "ProductViewSummary": {
            "SupportEmail": "test@amazon.com",
            "Type": "CLOUD_FORMATION_TEMPLATE",
            "Distributor": "test-distributor",
            "ShortDescription": "test-description",
            "Owner": "test-owner",
            "Id": "prodview-wi3l2j4abc6vc",
            "SupportDescription": "test-support",
            "ProductId": "prod-abcdcek6yhbxi",
            "HasDefaultPath": false,
            "Name": "test-product3",
            "SupportUrl": "https://aws.amazon.com"
        },
        "CreatedTime": 1577136715.0,
        "Status": "CREATED"
    },
    "ProvisioningArtifactSummaries": [
        {
            "CreatedTime": 1577136715.0,
            "Description": "test-version-description",
            "ProvisioningArtifactMetadata": {
                "SourceProvisioningArtifactId": "pa-abcdxkkiv5fcm"
            },
            "Name": "test-version-name-3",
```

```
            "Id": "pa-abcdxkkiv5fcm"
        }
    ],
    "Tags": [
        {
            "Value": "iad",
            "Key": "region"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProductAsAdmin。

describe-provisioned-product

以下代码示例演示了如何使用 describe-provisioned-product。

Amazon CLI

描述预置产品

以下 describe-provisioned-product 示例显示指定预置产品的详细信息。

```
aws servicecatalog describe-provisioned-product \
    --id pp-dpom27bm4abcd
```

输出：

```
{
    "ProvisionedProductDetail": {
        "Status": "ERROR",
        "CreatedTime": 1577222793.358,
        "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/mytestppname3/
pp-dpom27bm4abcd",
        "Id": "pp-dpom27bm4abcd",
        "StatusMessage": "AmazonCloudFormationException  Parameters: [KeyName]
 must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:
 ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",
        "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",
        "LastRecordId": "rec-tfuawdjovzxge",
        "Type": "CFN_STACK",
        "Name": "mytestppname3"
    },
```

```
    "CloudWatchDashboards": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProvisionedProduct。

describe-provisioning-artifact

以下代码示例演示了如何使用 describe-provisioning-artifact。

Amazon CLI

描述预置构件

以下 describe-provisioning-artifact 示例显示指定预置构件的详细信息。

```
aws servicecatalog describe-provisioning-artifact \
    --provisioning-artifact-id pa-pcz347abcdcfm \
    --product-id prod-abcdfz3syn2rg
```

输出：

```
{
    "Info": {
        "TemplateUrl": "https://awsdocs.s3.amazonaws.com/servicecatalog/
myexampledevelopment-environment.template"
    },
    "ProvisioningArtifactDetail": {
        "Id": "pa-pcz347abcdcfm",
        "Active": true,
        "Type": "CLOUD_FORMATION_TEMPLATE",
        "Description": "updated description",
        "CreatedTime": 1562097906.0,
        "Name": "updated name"
    },
    "Status": "AVAILABLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProvisioningArtifact。

describe-tag-option

以下代码示例演示了如何使用 describe-tag-option。

Amazon CLI

描述 TagOption

以下 describe-tag-option 示例显示指定 TagOption 的详细信息。

```
aws servicecatalog describe-tag-option \
    --id tag-p3tej2abcd5qc
```

输出：

```
{
    "TagOptionDetail": {
        "Active": true,
        "Id": "tag-p3tej2abcd5qc",
        "Value": "value-3",
        "Key": "1234"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTagOption。

disassociate-principal-from-portfolio

以下代码示例演示了如何使用 disassociate-principal-from-portfolio。

Amazon CLI

解除主体与产品组合的关联

以下 disassociate-principal-from-portfolio 示例解除指定主体与产品组合的关联。

```
aws servicecatalog disassociate-principal-from-portfolio \
    --portfolio-id port-2s6abcdq5wdh4 \
    --principal-arn arn:aws:iam::123456789012:group/myendusers
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociatePrincipalFromPortfolio。

disassociate-product-from-portfolio

以下代码示例演示了如何使用 disassociate-product-from-portfolio。

Amazon CLI

解除产品与产品组合的关联

以下 disassociate-product-from-portfolio 示例解除指定产品与产品组合的关联。

```
aws servicecatalog disassociate-product-from-portfolio \
    --product-id prod-3p5abcdmu3oyk \
    --portfolio-id port-2s6abcdq5wdh4
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateProductFromPortfolio。

disassociate-tag-option-from-resource

以下代码示例演示了如何使用 disassociate-tag-option-from-resource。

Amazon CLI

解除 TagOption 与资源的关联

以下 disassociate-tag-option-from-resource 示例解除指定 TagOption 与资源的关联。

```
aws servicecatalog disassociate-tag-option-from-resource \
     --resource-id port-2s6abcdq5wdh4 \
     --tag-option-id tag-p3abc2pkpz5qc
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateTagOptionFromResource。

list-accepted-portfolio-shares

以下代码示例演示了如何使用 list-accepted-portfolio-shares。

Amazon CLI

列出已接受共享的产品组合

以下 list-accepted-portfolio-shares 示例列出该账户接受共享的所有产品组合，仅包括默认的服务目录产品组合。

```
aws servicecatalog list-accepted-portfolio-shares \
    --portfolio-share-type "AWS_SERVICECATALOG"
```

输出：

```
{
    "PortfolioDetails": [
        {
            "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/port-
d2abcd5dpkuma",
            "Description": "AWS Service Catalog Reference blueprints for often-used
 AWS services such as EC2, S3, RDS, VPC and EMR.",
            "CreatedTime": 1574456190.687,
            "ProviderName": "AWS Service Catalog",
            "DisplayName": "Reference Architectures",
            "Id": "port-d2abcd5dpkuma"
        },
        {
            "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/port-
abcdefaua7zpu",
            "Description": "AWS well-architected blueprints for high reliability
 applications.",
            "CreatedTime": 1574461496.092,
            "ProviderName": "AWS Service Catalog",
            "DisplayName": "High Reliability Architectures",
            "Id": "port-abcdefaua7zpu"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAcceptedPortfolioShares。

list-portfolio-access

以下代码示例演示了如何使用 list-portfolio-access。

Amazon CLI

列出有权访问产品组合的账户

以下 `list-portfolio-access` 示例列出有权访问指定产品组合的 Amazon 账户。

```
aws servicecatalog list-portfolio-access \
    --portfolio-id port-2s6abcdq5wdh4
```

输出：

```
{
    "AccountIds": [
        "123456789012"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPortfolioAccess。

list-portfolios-for-product

以下代码示例演示了如何使用 `list-portfolios-for-product`。

Amazon CLI

列出与产品关联的产品组合

以下 `list-portfolios-for-product` 示例列出与指定产品关联的产品组合。

```
aws servicecatalog list-portfolios-for-product \
    --product-id prod-abcdfz3syn2rg
```

输出：

```
{
    "PortfolioDetails": [
        {
            "CreatedTime": 1571337221.555,
            "Id": "port-2s6abcdq5wdh4",
            "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/
port-2s6abcdq5wdh4",
```

```
            "DisplayName": "my-portfolio",
            "ProviderName": "my-provider"
        },
        {
            "CreatedTime": 1559665256.348,
            "Id": "port-5abcd3e5st4ei",
            "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/
port-5abcd3e5st4ei",
            "DisplayName": "test",
            "ProviderName": "provider-name"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPortfoliosForProduct。

list-portfolios

以下代码示例演示了如何使用 list-portfolios。

Amazon CLI

列出产品组合

以下 list-portfolios 示例列出当前区域的服务目录产品组合。

```
aws servicecatalog list-portfolios
```

输出：

```
{
    "PortfolioDetails": [
        {
           "CreatedTime": 1559665256.348,
           "ARN": "arn:aws:catalog:us-east-2:123456789012:portfolio/
port-5pzcxmplst4ei",
           "DisplayName": "my-portfolio",
           "Id": "port-5pzcxmplst4ei",
           "ProviderName": "my-user"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPortfolios。

list-principals-for-portfolio

以下代码示例演示了如何使用 list-principals-for-portfolio。

Amazon CLI

列出产品组合的所有主体

以下 list-principals-for-portfolio 示例列出指定产品组合的所有主体。

```
aws servicecatalog list-principals-for-portfolio \
    --portfolio-id port-2s6abcdq5wdh4
```

输出：

```
{
    "Principals": [
        {
            "PrincipalARN": "arn:aws:iam::123456789012:user/usertest",
            "PrincipalType": "IAM"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPrincipalsForPortfolio。

list-provisioning-artifacts

以下代码示例演示了如何使用 list-provisioning-artifacts。

Amazon CLI

列出产品的所有预置构件

以下 list-provisioning-artifacts 示例列出指定产品的所有预置构件。

```
aws servicecatalog list-provisioning-artifacts \
    --product-id prod-nfi2abcdefgcpw
```

输出：

```
{
    "ProvisioningArtifactDetails": [
        {
            "Id": "pa-abcdef54ipm6z",
            "Description": "test-version-description",
            "Type": "CLOUD_FORMATION_TEMPLATE",
            "CreatedTime": 1576021147.0,
            "Active": true,
            "Name": "test-version-name"
        },
        {
            "Id": "pa-bb4zyxwwnaio",
            "Description": "test description",
            "Type": "CLOUD_FORMATION_TEMPLATE",
            "CreatedTime": 1576022545.0,
            "Active": true,
            "Name": "test-provisioning-artifact-2"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProvisioningArtifacts。

list-resources-for-tag-option

以下代码示例演示了如何使用 list-resources-for-tag-option。

Amazon CLI

列出与 TagOption 关联的资源

以下 list-resources-for-tag-option 示例列出与指定 TagOption 关联的资源。

```
aws servicecatalog list-resources-for-tag-option \
    --tag-option-id tag-p3tej2abcd5qc
```

输出：

```
{
```

```
    "ResourceDetails": [
        {
            "ARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-
abcdfz3syn2rg",
            "Name": "my product",
            "Description": "description",
            "CreatedTime": 1562097906.0,
            "Id": "prod-abcdfz3syn2rg"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourcesForTagOption。

list-tag-options

以下代码示例演示了如何使用 list-tag-options。

Amazon CLI

以下 list-tag-options 示例列出 TagOptions 的所有值。

```
aws servicecatalog list-tag-options
```

输出：

```
{
    "TagOptionDetails": [
        {
            "Value": "newvalue",
            "Active": true,
            "Id": "tag-iabcdn4fzjjms",
            "Key": "1234"
        },
        {
            "Value": "value1",
            "Active": true,
            "Id": "tag-e3abcdvmwvrzy",
            "Key": "key"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagOptions。

provision-product

以下代码示例演示了如何使用 provision-product。

Amazon CLI

置备产品

以下 provision-product 示例使用指定的预置构件置备指定的产品。

```
aws servicecatalog provision-product \
    --product-id prod-abcdfz3syn2rg \
    --provisioning-artifact-id pa-abc347pcsccfm \
    --provisioned-product-name "mytestppname3"
```

输出：

```
{
    "RecordDetail": {
        "RecordId": "rec-tfuawdabcdege",
        "CreatedTime": 1577222793.362,
        "ProvisionedProductId": "pp-abcd27bm4mldq",
        "PathId": "lpv2-abcdg3jp6t5k6",
        "RecordErrors": [],
        "ProductId": "prod-abcdfz3syn2rg",
        "UpdatedTime": 1577222793.362,
        "RecordType": "PROVISION_PRODUCT",
        "ProvisionedProductName": "mytestppname3",
        "ProvisioningArtifactId": "pa-pcz347abcdcfm",
        "RecordTags": [],
        "Status": "CREATED",
        "ProvisionedProductType": "CFN_STACK"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ProvisionProduct。

reject-portfolio-share

以下代码示例演示了如何使用 reject-portfolio-share。

Amazon CLI

拒绝分享产品组合

以下 reject-portfolio-share 示例拒绝分享给定产品组合的产品组合。

```
aws servicecatalog reject-portfolio-share \
    --portfolio-id port-2s6wuabcdefghijk
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RejectPortfolioShare。

scan-provisioned-products

以下代码示例演示了如何使用 scan-provisioned-products。

Amazon CLI

列出所有可用的预置产品

以下 scan-provisioned-products 示例列出可用的预置产品。

```
aws servicecatalog scan-provisioned-products
```

输出：

```
{
    "ProvisionedProducts": [
        {
            "Status": "ERROR",
            "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/
mytestppname3/pp-abcd27bm4mldq",
            "StatusMessage": "AmazonCloudFormationException  Parameters: [KeyName]
 must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:
 ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",
            "Id": "pp-abcd27bm4mldq",
            "Type": "CFN_STACK",
            "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",
            "CreatedTime": 1577222793.358,
            "Name": "mytestppname3",
            "LastRecordId": "rec-tfuawdabcdxge"
        }
```

```
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ScanProvisionedProducts。

search-products-as-admin

以下代码示例演示了如何使用 search-products-as-admin。

Amazon CLI

搜索具有管理员权限的产品

以下 search-products-as-admin 示例使用产品组合 ID 作为筛选条件搜索具有管理员权限的产品。

```
aws servicecatalog search-products-as-admin \
    --portfolio-id port-5abcd3e5st4ei
```

输出：

```
{
    "ProductViewDetails": [
        {
            "ProductViewSummary": {
                "Name": "my product",
                "Owner": "owner name",
                "Type": "CLOUD_FORMATION_TEMPLATE",
                "ProductId": "prod-abcdfz3syn2rg",
                "HasDefaultPath": false,
                "Id": "prodview-abcdmyuzv2dlu",
                "ShortDescription": "description"
            },
            "ProductARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-
abcdfz3syn2rg",
            "CreatedTime": 1562097906.0,
            "Status": "CREATED"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchProductsAsAdmin。

search-provisioned-products

以下代码示例演示了如何使用 search-provisioned-products。

Amazon CLI

搜索预置产品

以下 search-provisioned-products 示例使用 JSON 文件传递参数，搜索与指定产品 ID 匹配的预置产品。

```
aws servicecatalog search-provisioned-products \
    --cli-input-json file://search-provisioned-products-input.json
```

的内容search-provisioned-products-input.json：

```
{
    "Filters": {
        "SearchQuery": [
            "prod-tcjvfz3syn2rg"
        ]
    }
}
```

输出：

```
{
    "ProvisionedProducts": [
        {
            "ProvisioningArtifactId": "pa-pcz347abcdcfm",
            "Name": "mytestppname3",
            "CreatedTime": 1577222793.358,
            "Id": "pp-abcd27bm4mldq",
            "Status": "ERROR",
            "UserArn": "arn:aws:iam::123456789012:user/cliuser",
            "StatusMessage": "AmazonCloudFormationException  Parameters: [KeyName]
 must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:
 ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",
            "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/
mytestppname3/pp-abcd27bm4mldq",
            "Tags": [
```

```
                {
                    "Value": "arn:aws:catalog:us-west-2:123456789012:product/prod-
abcdfz3syn2rg",
                    "Key": "aws:servicecatalog:productArn"
                },
                {
                    "Value": "arn:aws:iam::123456789012:user/cliuser",
                    "Key": "aws:servicecatalog:provisioningPrincipalArn"
                },
                {
                    "Value": "value-3",
                    "Key": "1234"
                },
                {
                    "Value": "pa-pcz347abcdcfm",
                    "Key": "aws:servicecatalog:provisioningArtifactIdentifier"
                },
                {
                    "Value": "arn:aws:catalog:us-west-2:123456789012:portfolio/
port-2s6abcdq5wdh4",
                    "Key": "aws:servicecatalog:portfolioArn"
                },
                {
                    "Value": "arn:aws:servicecatalog:us-west-2:123456789012:stack/
mytestppname3/pp-abcd27bm4mldq",
                    "Key": "aws:servicecatalog:provisionedProductArn"
                }
            ],
            "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",
            "UserArnSession": "arn:aws:iam::123456789012:user/cliuser",
            "Type": "CFN_STACK",
            "LastRecordId": "rec-tfuawdabcdxge",
            "ProductId": "prod-abcdfz3syn2rg"
        }
    ],
    "TotalResultsCount": 1
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SearchProvisionedProducts。

update-portfolio

以下代码示例演示了如何使用 update-portfolio。

Amazon CLI

更新产品组合

以下 update-portfolio 示例更新指定产品组合的名称。

```
aws servicecatalog update-portfolio \
    --id port-5abcd3e5st4ei \
    --display-name "New portfolio name"
```

输出：

```
{
    "PortfolioDetail": {
        "DisplayName": "New portfolio name",
        "ProviderName": "provider",
        "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/
port-5abcd3e5st4ei",
        "Id": "port-5abcd3e5st4ei",
        "CreatedTime": 1559665256.348
    },
    "Tags": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePortfolio。

update-product

以下代码示例演示了如何使用 update-product。

Amazon CLI

更新产品

以下 update-product 示例更新指定产品的名称和所有人。

```
aws servicecatalog update-product \
    --id prod-os6abc7drqlt2 \
    --name "New product name" \
    --owner "Updated product owner"
```

输出：

```
{
    "Tags": [
        {
            "Value": "iad",
            "Key": "region"
        }
    ],
    "ProductViewDetail": {
        "ProductViewSummary": {
            "Owner": "Updated product owner",
            "ProductId": "prod-os6abc7drqlt2",
            "Distributor": "test-distributor",
            "SupportUrl": "https://aws.amazon.com",
            "Name": "New product name",
            "ShortDescription": "test-description",
            "HasDefaultPath": false,
            "Id": "prodview-6abcdgrfhvidy",
            "SupportDescription": "test-support",
            "SupportEmail": "test@amazon.com",
            "Type": "CLOUD_FORMATION_TEMPLATE"
        },
        "Status": "CREATED",
        "ProductARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-
os6abc7drqlt2",
        "CreatedTime": 1577136255.0
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProduct。

update-provisioning-artifact

以下代码示例演示了如何使用 update-provisioning-artifact。

Amazon CLI

更新预置构件

以下 update-provisioning-artifact 示例使用 JSON 文件传递参数，更新指定预置构件的名称和描述。

```
aws servicecatalog update-provisioning-artifact \
    --cli-input-json file://update-provisioning-artifact-input.json
```

的内容update-provisioning-artifact-input.json：

```
{
    "ProductId": "prod-abcdfz3syn2rg",
    "ProvisioningArtifactId": "pa-pcz347abcdcfm",
    "Name": "updated name",
    "Description": "updated description"
}
```

输出：

```
{
    "Info": {
        "TemplateUrl": "https://awsdocs.s3.amazonaws.com/servicecatalog/
myexampledevelopment-environment.template"
    },
    "Status": "AVAILABLE",
    "ProvisioningArtifactDetail": {
        "Active": true,
        "Description": "updated description",
        "Id": "pa-pcz347abcdcfm",
        "Name": "updated name",
        "Type": "CLOUD_FORMATION_TEMPLATE",
        "CreatedTime": 1562097906.0
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateProvisioningArtifact。

update-tag-option

以下代码示例演示了如何使用 update-tag-option。

Amazon CLI

更新 TagOption

以下 update-tag-option 示例使用指定 JSON 文件更新 TagOption 的值。

```
aws servicecatalog update-tag-option --cli-input-json file://update-tag-option-input.json
```

的内容update-tag-option-input.json：

```
{
    "Id": "tag-iabcdn4fzjjms",
    "Value": "newvalue",
    "Active": true
}
```

输出：

```
{
    "TagOptionDetail": {
        "Value": "newvalue",
        "Key": "1234",
        "Active": true,
        "Id": "tag-iabcdn4fzjjms"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateTagOption。

使用 Amazon CLI 的服务配额示例

以下代码示例演示如何通过将 Amazon Command Line Interface与服务配额结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

get-aws-default-service-quota

以下代码示例演示了如何使用 get-aws-default-service-quota。

Amazon CLI

描述默认服务配额

以下 get-aws-default-service-quota 示例显示指定配额的详细信息。

```
aws service-quotas get-aws-default-service-quota \
    --service-code ec2 \
    --quota-code L-1216C47A
```

输出：

```
{
    "Quota": {
        "ServiceCode": "ec2",
        "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
        "QuotaArn": "arn:aws:servicequotas:us-east-2::ec2/L-1216C47A",
        "QuotaCode": "L-1216C47A",
        "QuotaName": "Running On-Demand Standard (A, C, D, H, I, M, R, T, Z)
 instances",
        "Value": 5.0,
        "Unit": "None",
        "Adjustable": true,
        "GlobalQuota": false,
        "UsageMetric": {
            "MetricNamespace": "AWS/Usage",
            "MetricName": "ResourceCount",
            "MetricDimensions": {
                "Class": "Standard/OnDemand",
                "Resource": "vCPU",
                "Service": "EC2",
                "Type": "Resource"
            },
            "MetricStatisticRecommendation": "Maximum"
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAwsDefaultServiceQuota。

get-requested-service-quota-change

以下代码示例演示了如何使用 get-requested-service-quota-change。

Amazon CLI

描述服务配额提升请求

以下 get-requested-service-quota-change 示例描述指定配额的提升请求。

```
aws service-quotas get-requested-service-quota-change \
    --request-id d187537d15254312a9609aa51bbf7624u7W49tPO
```

输出：

```
{
    "RequestedQuota": {
        "Id": "d187537d15254312a9609aa51bbf7624u7W49tPO",
        "CaseId": "6780195351",
        "ServiceCode": "ec2",
        "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
        "QuotaCode": "L-20F13EBD",
        "QuotaName": "Running Dedicated c5n Hosts",
        "DesiredValue": 2.0,
        "Status": "CASE_OPENED",
        "Created": 1580446904.067,
        "LastUpdated": 1580446953.265,
        "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\"arn:aws:iam::123456789012:root\"}",
        "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-20F13EBD",
        "GlobalQuota": false,
        "Unit": "None"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRequestedServiceQuotaChange。

get-service-quota

以下代码示例演示了如何使用 get-service-quota。

Amazon CLI

描述服务配额

以下 get-service-quota 示例显示关于指定配额的详细信息。

```
aws service-quotas get-service-quota \
    --service-code ec2 \
    --quota-code L-1216C47A
```

输出：

```
{
    "Quota": {
        "ServiceCode": "ec2",
        "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
        "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-1216C47A",
        "QuotaCode": "L-1216C47A",
        "QuotaName": "Running On-Demand Standard (A, C, D, H, I, M, R, T, Z)
 instances",
        "Value": 1920.0,
        "Unit": "None",
        "Adjustable": true,
        "GlobalQuota": false,
        "UsageMetric": {
            "MetricNamespace": "AWS/Usage",
            "MetricName": "ResourceCount",
            "MetricDimensions": {
                "Class": "Standard/OnDemand",
                "Resource": "vCPU",
                "Service": "EC2",
                "Type": "Resource"
            },
            "MetricStatisticRecommendation": "Maximum"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceQuota。

list-aws-default-service-quotas

以下代码示例演示了如何使用 list-aws-default-service-quotas。

Amazon CLI

列出服务的默认配额

以下 list-aws-default-service-quotas 示例列出指定服务配额的默认值。

```
aws service-quotas list-aws-default-service-quotas \
    --service-code xray
```

输出：

```
{
    "Quotas": [
        {
            "ServiceCode": "xray",
            "ServiceName": "AWS X-Ray",
            "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-C6B6F05D",
            "QuotaCode": "L-C6B6F05D",
            "QuotaName": "Indexed annotations per trace",
            "Value": 50.0,
            "Unit": "None",
            "Adjustable": false,
            "GlobalQuota": false
        },
        {
            "ServiceCode": "xray",
            "ServiceName": "AWS X-Ray",
            "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-D781C0FD",
            "QuotaCode": "L-D781C0FD",
            "QuotaName": "Segment document size",
            "Value": 64.0,
            "Unit": "Kilobytes",
            "Adjustable": false,
            "GlobalQuota": false
        },
        {
            "ServiceCode": "xray",
            "ServiceName": "AWS X-Ray",
            "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-998BFF16",
```

```
            "QuotaCode": "L-998BFF16",
            "QuotaName": "Trace and service graph retention in days",
            "Value": 30.0,
            "Unit": "None",
            "Adjustable": false,
            "GlobalQuota": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAwsDefaultServiceQuotas。

`list-requested-service-quota-change-history-by-quota`

以下代码示例演示了如何使用 `list-requested-service-quota-change-history-by-quota`。

Amazon CLI

列出您的配额提升请求

以下 `list-requested-service-quota-change-history-by-quota` 示例列出指定配额的配额提升请求。

```
aws service-quotas list-requested-service-quota-change-history-by-quota \
    --service-code ec2 \
    --quota-code L-20F13EBD
```

输出：

```
{
    "RequestedQuotas": [
        {
            "Id": "d187537d15254312a9609aa51bbf7624u7W49tPO",
            "CaseId": "6780195351",
            "ServiceCode": "ec2",
            "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
            "QuotaCode": "L-20F13EBD",
            "QuotaName": "Running Dedicated c5n Hosts",
            "DesiredValue": 2.0,
            "Status": "CASE_OPENED",
```

```
            "Created": 1580446904.067,
            "LastUpdated": 1580446953.265,
            "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\"arn:aws:iam::123456789012:root\"}",
            "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/
L-20F13EBD",
            "GlobalQuota": false,
            "Unit": "None"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》 中的 ListRequestedServiceQuotaChangeHistoryByQuota。

list-requested-service-quota-change-history

以下代码示例演示了如何使用 list-requested-service-quota-change-history。

Amazon CLI

列出您的配额提升请求

以下 list-requested-service-quota-change-history 示例列出指定服务的配额提升请求。

```
aws service-quotas list-requested-service-quota-change-history \
    --service-code ec2
```

输出：

```
{
    "RequestedQuotas": [
        {
            "Id": "d187537d15254312a9609aa51bbf7624u7W49tPO",
            "CaseId": "6780195351",
            "ServiceCode": "ec2",
            "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
            "QuotaCode": "L-20F13EBD",
            "QuotaName": "Running Dedicated c5n Hosts",
            "DesiredValue": 2.0,
```

```
            "Status": "CASE_OPENED",
            "Created": 1580446904.067,
            "LastUpdated": 1580446953.265,
            "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\"arn:aws:iam::123456789012:root\"}",
            "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/
L-20F13EBD",
            "GlobalQuota": false,
            "Unit": "None"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRequestedServiceQuotaChangeHistory。

`list-service-quotas`

以下代码示例演示了如何使用 `list-service-quotas`。

Amazon CLI

列出服务的配额

以下 `list-service-quotas` 示例显示有关 Amazon CloudFormation 配额的详细信息。

```
aws service-quotas list-service-quotas \
    --service-code cloudformation
```

输出：

```
{
    "Quotas": [
        {
            "ServiceCode": "cloudformation",
            "ServiceName": "AWS CloudFormation",
            "QuotaArn": "arn:aws:servicequotas:us-
east-2:123456789012:cloudformation/L-87D14FB7",
            "QuotaCode": "L-87D14FB7",
            "QuotaName": "Output count in CloudFormation template",
            "Value": 60.0,
```

```
            "Unit": "None",
            "Adjustable": false,
            "GlobalQuota": false
        },
        {
            "ServiceCode": "cloudformation",
            "ServiceName": "AWS CloudFormation",
            "QuotaArn": "arn:aws:servicequotas:us-
east-2:123456789012:cloudformation/L-0485CB21",
            "QuotaCode": "L-0485CB21",
            "QuotaName": "Stack count",
            "Value": 200.0,
            "Unit": "None",
            "Adjustable": true,
            "GlobalQuota": false
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceQuotas。

list-services

以下代码示例演示了如何使用 `list-services`。

Amazon CLI

列出可用服务

以下命令将列出服务配额中可用的服务。

```
aws service-quotas list-services
```

输出：

```
{
    "Services": [
        {
            "ServiceCode": "AWSCloudMap",
            "ServiceName": "AWS Cloud Map"
        },
        {
```

```
            "ServiceCode": "access-analyzer",
            "ServiceName": "Access Analyzer"
        },
        {
            "ServiceCode": "acm",
            "ServiceName": "AWS Certificate Manager (ACM)"
        },

        ...truncated...

        {
            "ServiceCode": "xray",
            "ServiceName": "AWS X-Ray"
        }
    ]
}
```

您可以添加 `--query` 参数，筛选显示您感兴趣的信息。以下示例仅显示了服务代码。

```
aws service-quotas list-services \
    --query Services[*].ServiceCode
```

输出：

```
[
    "AWSCloudMap",
    "access-analyzer",
    "acm",
    "acm-pca",
    "amplify",
    "apigateway",
    "application-autoscaling",
        ...truncated...
    "xray"
]
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

request-service-quota-increase

以下代码示例演示了如何使用 `request-service-quota-increase`。

Amazon CLI

请求提升服务配额

以下 `request-service-quota-increase` 示例请求提升指定的服务配额。

```
aws service-quotas request-service-quota-increase \
    --service-code ec2 \
    --quota-code L-20F13EBD \
    --desired-value 2
```

输出：

```
{
    "RequestedQuota": {
        "Id": "d187537d15254312a9609aa51bbf7624u7W49tPO",
        "ServiceCode": "ec2",
        "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
        "QuotaCode": "L-20F13EBD",
        "QuotaName": "Running Dedicated c5n Hosts",
        "DesiredValue": 2.0,
        "Status": "PENDING",
        "Created": 1580446904.067,
        "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\"arn:aws:iam::123456789012:root\"}",
        "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-20F13EBD",
        "GlobalQuota": false,
        "Unit": "None"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RequestServiceQuotaIncrease。

使用Amazon CLI的 Amazon SES 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon SES 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`delete-identity`

以下代码示例演示了如何使用 `delete-identity`。

Amazon CLI

删除身份

以下示例使用 `delete-identity` 命令从通过 Amazon SES 验证的身份列表中删除身份：

```
aws ses delete-identity --identity user@example.com
```

有关已验证身份的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址和域”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteIdentity。

`get-identity-dkim-attributes`

以下代码示例演示了如何使用 `get-identity-dkim-attributes`。

Amazon CLI

获取身份列表的 Amazon SES Easy DKIM 属性

以下示例使用 `get-identity-dkim-attributes` 命令检索身份列表的 Amazon SES Easy DKIM 属性：

```
aws ses get-identity-dkim-attributes --identities "example.com" "user@example.com"
```

输出：

```
{
   "DkimAttributes": {
       "example.com": {
           "DkimTokens": [
               "EXAMPLEjcs5xoyqytjsotsijas7236gr",
               "EXAMPLEjr76cvoc6mysspnioorxsn6ep",
               "EXAMPLEkbmkqkhlm2lyz77ppkulerm4k"
           ],
           "DkimEnabled": true,
           "DkimVerificationStatus": "Success"
       },
       "user@example.com": {
           "DkimEnabled": false,
           "DkimVerificationStatus": "NotStarted"
       }
   }
}
```

如果调用此命令时，列表包含从未提交验证的身份，则该身份将不会出现在输出中。

有关 Easy DKIM 的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“Amazon SES 中的 Easy DKIM”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIdentityDkimAttributes。

`get-identity-notification-attributes`

以下代码示例演示了如何使用 `get-identity-notification-attributes`。

Amazon CLI

获取身份列表的 Amazon SES 通知属性

以下示例使用 `get-identity-notification-attributes` 命令检索身份列表的 Amazon SES 通知属性：

```
aws ses get-identity-notification-attributes --
identities "user1@example.com" "user2@example.com"
```

输出：

```
{
```

```
    "NotificationAttributes": {
        "user1@example.com": {
            "ForwardingEnabled": false,
            "ComplaintTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic",
            "BounceTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic",
            "DeliveryTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic"
        },
        "user2@example.com": {
            "ForwardingEnabled": true
        }
    }
}
```

此命令将返回电子邮件反馈转发的状态，如果适用，还将返回向其发送退信、投诉和送达通知的 Amazon SNS 主题的 Amazon 资源名称（ARN）。

如果调用此命令时，列表包含从未提交验证的身份，则该身份将不会出现在输出中。

有关通知的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“使用 Amazon SES 的通知”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIdentityNotificationAttributes。

`get-identity-verification-attributes`

以下代码示例演示了如何使用 `get-identity-verification-attributes`。

Amazon CLI

获取身份列表的 Amazon SES 验证状态

以下示例使用 `get-identity-verification-attributes` 命令检索身份列表的 Amazon SES 验证状态：

```
aws ses get-identity-verification-attributes --
identities "user1@example.com" "user2@example.com"
```

输出：

```
{
    "VerificationAttributes": {
```

```
        "user1@example.com": {
            "VerificationStatus": "Success"
        },
        "user2@example.com": {
            "VerificationStatus": "Pending"
        }
    }
}
```

如果调用此命令时，列表包含从未提交验证的身份，则该身份将不会出现在输出中。

有关已验证身份的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址和域”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetIdentityVerificationAttributes。

get-send-quota

以下代码示例演示了如何使用 get-send-quota。

Amazon CLI

获取 Amazon SES 发送限制

以下示例使用 get-send-quota 命令返回 Amazon SES 发送限制：

```
aws ses get-send-quota
```

输出：

```
{
    "Max24HourSend": 200.0,
    "SentLast24Hours": 1.0,
    "MaxSendRate": 1.0
}
```

Max24HourSend 是发送配额，即，您在 24 小时之内可发送的最大电子邮件数量。发送配额反映一个滚动的时段。每当您尝试发送电子邮件时，Amazon SES 都会检查您在过去 24 小时内发送的电子邮件数量。只要您发送电子邮件总数小于您的配额，发送请求就会被接受，并发送您的电子邮件。

SentLast24Hours 是您在过去 24 小时内已发送的电子邮件数量。

MaxSendRate 是您每秒最多可发送的电子邮件数量。

请注意，发送限制基于收件人而不是消息。例如，一封包含 10 个收件人的电子邮件占用 10 份发送配额。

有关更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“管理您的 Amazon SES 发送限制”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetSendQuota。

`get-send-statistics`

以下代码示例演示了如何使用 `get-send-statistics`。

Amazon CLI

获取 Amazon SES 发送统计信息

以下示例使用 `get-send-statistics` 命令返回 Amazon SES 发送统计信息

```
aws ses get-send-statistics
```

输出：

```
{
   "SendDataPoints": [
       {
           "Complaints": 0,
           "Timestamp": "2013-06-12T19:32:00Z",
           "DeliveryAttempts": 2,
           "Bounces": 0,
           "Rejects": 0
       },
       {
           "Complaints": 0,
           "Timestamp": "2013-06-12T00:47:00Z",
           "DeliveryAttempts": 1,
           "Bounces": 0,
           "Rejects": 0
       }
   ]
```

```
}
```

结果是数据点的列表，表示过去两周的发送活动。列表中的每个数据点都包含 15 分钟间隔的统计信息。

在此示例中，只有两个数据点，因为用户在过去两周内发送的唯一电子邮件是在两个 15 分钟间隔内发送的。

有关更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“监控您的 Amazon SES 使用情况统计信息”。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 GetSendStatistics。

`list-identities`

以下代码示例演示了如何使用 `list-identities`。

Amazon CLI

列出特定 Amazon 账户的所有身份（电子邮件地址和域）

以下示例使用 `list-identities` 命令列出已提交到 Amazon SES 进行验证的所有身份：

```
aws ses list-identities
```

输出：

```
{
    "Identities": [
      "user@example.com",
      "example.com"
    ]
}
```

返回的列表包含所有身份，无论验证状态如何（已验证、待验证、失败等）。

在此示例中，由于未指定 identity-type 参数，因此返回了电子邮件地址和 域。

有关验证的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址和域”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListIdentities。

send-email

以下代码示例演示了如何使用 send-email。

Amazon CLI

使用 Amazon SES 发送格式化的电子邮件

以下示例使用 send-email 命令发送格式化的电子邮件：

```
aws ses send-email --from sender@example.com --destination file://destination.json
 --message file://message.json
```

输出：

```
{
   "MessageId": "EXAMPLEf3a5efcd1-51adec81-d2a4-4e3f-9fe2-5d85c1b23783-000000"
}
```

目标和消息是 JSON 数据结构，保存在当前目录下的 .json 文件中。这些文件如下所示：

destination.json:

```
{
  "ToAddresses":  ["recipient1@example.com", "recipient2@example.com"],
  "CcAddresses":  ["recipient3@example.com"],
  "BccAddresses": []
}
```

message.json:

```
{
   "Subject": {
       "Data": "Test email sent using the AWS CLI",
       "Charset": "UTF-8"
   },
   "Body": {
       "Text": {
           "Data": "This is the message body in text format.",
           "Charset": "UTF-8"
       },
```

```
        "Html": {
            "Data": "This message body contains HTML formatting. It can, for example,
 contain links like this one: <a class=\"ulink\" href=\"http://docs.aws.amazon.com/
ses/latest/DeveloperGuide\" target=\"_blank\">Amazon SES Developer Guide</a>.",
            "Charset": "UTF-8"
        }
    }
}
```

请将发件人和收件人的电子邮件地址替换为您要使用的电子邮件地址。请注意，发件人的电子邮件地址必须已通过 Amazon SES 验证。在您获得 Amazon SES 的生产访问权限之前，您还必须验证每个收件人的电子邮件地址，除非收件人是 Amazon SES 邮箱模拟器。有关验证的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址和域”。

输出中的消息 ID 表示 send-email 调用成功。

如果您没有收到电子邮件，请检查垃圾邮件。

有关发送格式化电子邮件的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“使用 Amazon SES API 发送格式化的电子邮件”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SendEmail。

`send-raw-email`

以下代码示例演示了如何使用 `send-raw-email`。

Amazon CLI

使用 Amazon SES 发送原始电子邮件

以下示例使用 `send-raw-email` 命令发送带有 TXT 附件的电子邮件：

```
aws ses send-raw-email --raw-message file://message.json
```

输出：

```
{
    "MessageId": "EXAMPLEf3f73d99b-c63fb06f-d263-41f8-a0fb-d0dc67d56c07-000000"
}
```

原始消息是一个 JSON 数据结构，保存在当前目录下名为 message.json 的文件中。其中包含以下内容：

```
{
   "Data": "From: sender@example.com\nTo: recipient@example.com\nSubject: Test email
 sent using the AWS CLI (contains an attachment)\nMIME-Version: 1.0\nContent-type:
 Multipart/Mixed; boundary=\"NextPart\"\n\n--NextPart\nContent-Type: text/plain
\n\nThis is the message body.\n\n--NextPart\nContent-Type: text/plain;\nContent-
Disposition: attachment; filename=\"attachment.txt\"\n\nThis is the text in the
 attachment.\n\n--NextPart--"
}
```

如您所见，“Data”是一个长字符串，包含 MIME 格式的全部原始电子邮件内容，其中包括一个名为 attachment.txt 的附件。

请将 sender@example.com 和 recipient@example.com 替换为您要使用的地址。请注意，发件人的电子邮件地址必须已通过 Amazon SES 验证。在您获得 Amazon SES 的生产访问权限之前，您还必须验证收件人的电子邮件地址，除非收件人是 Amazon SES 邮箱模拟器。有关验证的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址和域”。

输出中的消息 ID 表示 send-raw-email 调用成功。

如果您没有收到电子邮件，请检查垃圾邮件。

有关发送原始电子邮件的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“使用 Amazon SES API 发送原始电子邮件”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SendRawEmail。

set-identity-dkim-enabled

以下代码示例演示了如何使用 set-identity-dkim-enabled。

Amazon CLI

为经 Amazon SES 验证的身份启用或禁用 Easy DKIM

以下示例使用 set-identity-dkim-enabled 命令禁用已验证电子邮件地址的 DKIM：

```
aws ses set-identity-dkim-enabled --identity user@example.com --no-dkim-enabled
```

有关 Easy DKIM 的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“Amazon SES 中的 Easy DKIM”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetIdentityDkimEnabled。

set-identity-feedback-forwarding-enabled

以下代码示例演示了如何使用 set-identity-feedback-forwarding-enabled。

Amazon CLI

为经 Amazon SES 验证的身份启用或禁用退信和投诉电子邮件反馈转发

以下示例使用 set-identity-feedback-forwarding-enabled 命令使经验证的电子邮件地址能够通过电子邮件接收退信和投诉通知：

```
aws ses set-identity-feedback-forwarding-enabled --identity user@example.com --
forwarding-enabled
```

您必须通过 Amazon SNS 或电子邮件反馈转发接收退信和投诉通知，因此，只有在为退信和投诉通知选择 Amazon SNS 主题时，才可以禁用电子邮件反馈转发。

有关通知的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“使用 Amazon SES 的通知”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetIdentityFeedbackForwardingEnabled。

set-identity-notification-topic

以下代码示例演示了如何使用 set-identity-notification-topic。

Amazon CLI

设置 Amazon SNS 主题，Amazon SES 将向其发布已验证身份的退信、投诉和/或送达通知

以下示例使用 set-identity-notification-topic 命令指定经验证的电子邮件地址将接收退信通知的 Amazon SNS 主题：

```
aws ses set-identity-notification-topic --identity user@example.com --notification-
type Bounce --sns-topic arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic
```

有关通知的更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“使用 Amazon SES 的通知”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetIdentityNotificationTopic。

verify-domain-dkim

以下代码示例演示了如何使用 verify-domain-dkim。

Amazon CLI

生成经验证的域的 DKIM 令牌，以便使用 Amazon SES 进行 DKIM 签名

以下示例使用 verify-domain-dkim 命令为已通过 Amazon SES 验证的域生成 DKIM 令牌：

```
aws ses verify-domain-dkim --domain example.com
```

输出：

```
{
   "DkimTokens": [
       "EXAMPLEq76owjnks3lnluwg65scbemvw",
       "EXAMPLEi3dnsj67hstzaj673klariwx2",
       "EXAMPLEwfbtcukvimehexktmdtaz6naj"
   ]
}
```

要设置 DKIM，您必须使用返回的 DKIM 令牌更新域名的 DNS 设置，使其具有指向 Amazon SES 托管的 DKIM 公钥的 CNAME 记录。有关更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“Amazon SES 中的 Easy DKIM”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 VerifyDomainDkim。

verify-domain-identity

以下代码示例演示了如何使用 verify-domain-identity。

Amazon CLI

使用 Amazon SES 验证域

以下示例使用 verify-domain-identity 命令验证域：

```
aws ses verify-domain-identity --domain example.com
```

输出：

```
{
    "VerificationToken": "eoEmxw+YaYhb3h3iVJHuXMJXqeu1q1/wwmvjuEXAMPLE"
}
```

要完成域验证，您必须在域的 DNS 设置中添加一条包含返回的验证令牌的 TXT 记录。有关更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证域”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 VerifyDomainIdentity。

verify-email-identity

以下代码示例演示了如何使用 verify-email-identity。

Amazon CLI

使用 Amazon SES 验证电子邮件地址

以下示例使用 verify-email-identity 命令验证电子邮件地址：

```
aws ses verify-email-identity --email-address user@example.com
```

在使用 Amazon SES 发送电子邮件之前，您必须验证从中发送电子邮件的地址或域，以证明您拥有该地址或域。如果您尚未获得生产访问权限，则还需要验证所有收件电子邮件地址，Amazon SES 邮箱模拟器提供的电子邮件地址除外。

调用 verify-email-identity 后，该电子邮件地址将收到一封验证电子邮件。用户必须单击此电子邮件中的链接才能完成验证过程。

有关更多信息，请参阅《Amazon Simple Email Service 开发人员指南》中的“在 Amazon SES 中验证电子邮件地址”。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 VerifyEmailIdentity。

使用 Amazon CLI 的 Shield 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Shield 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-drt-log-bucket`

以下代码示例演示了如何使用 `associate-drt-log-bucket`。

Amazon CLI

授权 DRT 访问 Amazon S3 存储桶

以下 `associate-drt-log-bucket` 示例在 DRT 和指定的 S3 存储桶之间创建关联。这允许 DRT 代表账户访问存储桶：

```
aws shield associate-drt-log-bucket \
    --log-bucket flow-logs-for-website-lb
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的授权 DDoS 响应团队。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateDrtLogBucket。

`associate-drt-role`

以下代码示例演示了如何使用 `associate-drt-role`。

Amazon CLI

授权 DRT 代表您缓解潜在攻击

以下 `associate-drt-role` 示例在 DRT 和指定角色之间创建关联。DRT 可以使用该角色访问和管理账户。

```
aws shield associate-drt-role \
    --role-arn arn:aws:iam::123456789012:role/service-role/DrtRole
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的授权 DDoS 响应团队。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateDrtRole。

create-protection

以下代码示例演示了如何使用 create-protection。

Amazon CLI

为单个 Amazon 资源启用 Amazon Shield Advanced 保护

以下 create-protection 示例为指定 Amazon CloudFront 分配启用 Shield Advanced 保护。

```
aws shield create-protection \
    --name "Protection for CloudFront distribution" \
    --resource-arn arn:aws:cloudfront::123456789012:distribution/E198WC25FXOWY8
```

输出：

```
{
    "ProtectionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的指定要保护的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateProtection。

create-subscription

以下代码示例演示了如何使用 create-subscription。

Amazon CLI

为账户启用 Amazon Shield Advanced 保护

以下 create-subscription 示例为账户启用 Shield Advanced 保护。

```
aws shield create-subscription
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的 Amazon Shield Advanced 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSubscription。

delete-protection

以下代码示例演示了如何使用 delete-protection。

Amazon CLI

从 Amazon 资源中移除 Amazon Shield Advanced 保护

以下 delete-protection 示例移除指定的 Amazon Shield Advanced 保护。

```
aws shield delete-protection \
    --protection-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的从 Amazon 资源中移除 Amazon Shield Advanced。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteProtection。

describe-attack

以下代码示例演示了如何使用 describe-attack。

Amazon CLI

检索攻击的详细描述

以下 describe-attack 示例显示指定攻击 ID 的 DDoS 攻击的详细信息。您可以通过运行 list-attacks 命令来获取攻击 ID。

```
aws shield describe-attack --attack-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

输出：

```
{
    "Attack": {
        "AttackId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "ResourceArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/testElb",
        "SubResources": [
            {
                "Type": "IP",
                "Id": "192.0.2.2",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 11786208.0,
                                "N": 12,
                                "Unit": "BPS"
                            }
                        ]
                    }
                ],
                "Counters": []
            },
            {
                "Type": "IP",
                "Id": "192.0.2.3",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 9821840.0,
                                "N": 10,
                                "Unit": "BPS"
                            }
                        ]
```

```
                    }
                ],
                "Counters": []
            },
            {
                "Type": "IP",
                "Id": "192.0.2.4",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 7857472.0,
                                "N": 8,
                                "Unit": "BPS"
                            }
                        ]
                    }
                ],
                "Counters": []
            },
            {
                "Type": "IP",
                "Id": "192.0.2.5",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 1964368.0,
                                "N": 2,
                                "Unit": "BPS"
                            }
                        ]
                    }
                ],
                "Counters": []
            },
```

```
            {
                "Type": "IP",
                "Id": "2001:DB8::bcde:4321:8765:0:0",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 1964368.0,
                                "N": 2,
                                "Unit": "BPS"
                            }
                        ]
                    }
                ],
                "Counters": []
            },
            {
                "Type": "IP",
                "Id": "192.0.2.6",
                "AttackVectors": [
                    {
                        "VectorType": "SYN_FLOOD",
                        "VectorCounters": [
                            {
                                "Name": "SYN_FLOOD_BPS",
                                "Max": 982184.0,
                                "Average": 982184.0,
                                "Sum": 1964368.0,
                                "N": 2,
                                "Unit": "BPS"
                            }
                        ]
                    }
                ],
                "Counters": []
            }
        ],
        "StartTime": 1576024927.457,
        "EndTime": 1576025647.457,
        "AttackCounters": [],
```

```
        "AttackProperties": [
            {
                "AttackLayer": "NETWORK",
                "AttackPropertyIdentifier": "SOURCE_IP_ADDRESS",
                "TopContributors": [
                    {
                        "Name": "198.51.100.5",
                        "Value": 2024475682
                    },
                    {
                        "Name": "198.51.100.8",
                        "Value": 1311380863
                    },
                    {
                        "Name": "203.0.113.4",
                        "Value": 900599855
                    },
                    {
                        "Name": "198.51.100.4",
                        "Value": 769417366
                    },
                    {
                        "Name": "203.1.113.13",
                        "Value": 757992847
                    }
                ],
                "Unit": "BYTES",
                "Total": 92773354841
            },
            {
                "AttackLayer": "NETWORK",
                "AttackPropertyIdentifier": "SOURCE_COUNTRY",
                "TopContributors": [
                    {
                        "Name": "United States",
                        "Value": 80938161764
                    },
                    {
                        "Name": "Brazil",
                        "Value": 9929864330
                    },
                    {
                        "Name": "Netherlands",
                        "Value": 1635009446
```

```
                    },
                    {
                        "Name": "Mexico",
                        "Value": 144832971
                    },
                    {
                        "Name": "Japan",
                        "Value": 45369000
                    }
                ],
                "Unit": "BYTES",
                "Total": 92773354841
            },
            {
                "AttackLayer": "NETWORK",
                "AttackPropertyIdentifier": "SOURCE_ASN",
                "TopContributors": [
                    {
                        "Name": "12345",
                        "Value": 74953625841
                    },
                    {
                        "Name": "12346",
                        "Value": 4440087595
                    },
                    {
                        "Name": "12347",
                        "Value": 1635009446
                    },
                    {
                        "Name": "12348",
                        "Value": 1221230000
                    },
                    {
                        "Name": "12349",
                        "Value": 1199425294
                    }
                ],
                "Unit": "BYTES",
                "Total": 92755479921
            }
        ],
        "Mitigations": []
    }
```

```
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的查看 DDoS 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAttack。

describe-drt-access

以下代码示例演示了如何使用 describe-drt-access。

Amazon CLI

检索 DRT 代表您缓解攻击的授权说明

以下 describe-drt-access 示例检索 DRT 拥有的角色和 S3 存储桶授权，这使其能够代表您应对潜在的攻击。

```
aws shield describe-drt-access
```

输出：

```
{
    "RoleArn": "arn:aws:iam::123456789012:role/service-role/DrtRole",
    "LogBucketList": [
        "flow-logs-for-website-lb"
    ]
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的授权 DDoS 响应团队。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDrtAccess。

describe-emergency-contact-settings

以下代码示例演示了如何使用 describe-emergency-contact-settings。

Amazon CLI

检索您在 DRT 存档的紧急电子邮件地址

以下 describe-emergency-contact-settings 示例检索 DRT 存档的账户电子邮件地址。这些是 DRT 在应对可疑攻击时应联系的地址。

```
aws shield describe-emergency-contact-settings
```

输出：

```
{
    "EmergencyContactList": [
        {
            "EmailAddress": "ops@example.com"
        },
        {
            "EmailAddress": "ddos-notifications@example.com"
       }
    ]
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的“Amazon Shield 工作原理 <https://docs.aws.amazon.com/waf/latest/developerguide/ddos-overview.html>”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeEmergencyContactSettings。

describe-protection

以下代码示例演示了如何使用 describe-protection。

Amazon CLI

检索 Amazon Shield Advanced 保护的详细信息

以下 describe-protection 示例显示指定 ID 的 Shield Advanced 保护的详细信息。您可以通过运行 list-protections 命令来获取保护 ID。

```
aws shield describe-protection \
    --protection-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "Protection": {
        "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
        "Name": "1.2.3.4",
        "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:eip-allocation/
eipalloc-0ac1537af40742a6d"
    }
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的指定要保护的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeProtection。

`describe-subscription`

以下代码示例演示了如何使用 `describe-subscription`。

Amazon CLI

检索账户的 Amazon Shield Advanced 保护的详细信息

以下 `describe-subscription` 示例显示为账户提供的 Shield Advanced 保护的详细信息：

```
aws shield describe-subscription
```

输出：

```
{
    "Subscription": {
        "StartTime": 1534368978.0,
        "EndTime": 1597613778.0,
        "TimeCommitmentInSeconds": 63244800,
        "AutoRenew": "ENABLED",
        "Limits": [
            {
                "Type": "GLOBAL_ACCELERATOR",
                "Max": 1000
            },
            {
                "Type": "ROUTE53_HOSTED_ZONE",
                "Max": 1000
            },
            {
                "Type": "CF_DISTRIBUTION",
                "Max": 1000
```

```
            },
            {
                "Type": "ELB_LOAD_BALANCER",
                "Max": 1000
            },
            {
                "Type": "EC2_ELASTIC_IP_ALLOCATION",
                "Max": 1000
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的 Amazon Shield 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSubscription。

disassociate-drt-log-bucket

以下代码示例演示了如何使用 disassociate-drt-log-bucket。

Amazon CLI

移除 DRT 代表您访问 Amazon S3 存储桶的授权

以下 disassociate-drt-log-bucket 示例移除 DRT 与指定 S3 存储桶之间的关联。在此命令完成后，DRT 将无法再代表该账户访问该存储桶。

```
aws shield disassociate-drt-log-bucket \
    --log-bucket flow-logs-for-website-lb
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的授权 DDoS 响应团队。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateDrtLogBucket。

disassociate-drt-role

以下代码示例演示了如何使用 disassociate-drt-role。

Amazon CLI

移除 DRT 代表您缓解潜在攻击的授权

以下 `disassociate-drt-role` 示例移除 DRT 与账户之间的关联。该调用结束后，DRT 将无法再访问或管理您的帐户。

```
aws shield disassociate-drt-role
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的授权 DDoS 响应团队。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateDrtRole。

`get-subscription-state`

以下代码示例演示了如何使用 `get-subscription-state`。

Amazon CLI

检索账户 Amazon Shield Advanced 订阅的当前状态

以下 `get-subscription-state` 示例检索账户 Shield Advanced 保护的状态。

```
aws shield get-subscription-state
```

输出：

```
{
    "SubscriptionState": "ACTIVE"
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的 Amazon Shield 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubscriptionState。

`list-attacks`

以下代码示例演示了如何使用 `list-attacks`。

Amazon CLI

从 Amazon Shield Advanced 中检索攻击摘要

以下 `list-attacks` 示例检索指定时间段内指定 Amazon CloudFront 分发的攻击摘要。响应包括攻击 ID，您可以将其提供给 `describe-attack` 命令以获取攻击的详细信息。

```
aws shield list-attacks \
    --resource-arns arn:aws:cloudfront::12345678910:distribution/E1PXMP22ZVFAOR \
    --start-time FromInclusive=1529280000,ToExclusive=1529300000
```

输出：

```
{
    "AttackSummaries": [
        {
            "AttackId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "ResourceArn": "arn:aws:cloudfront::123456789012:distribution/
E1PXMP22ZVFAOR",
            "StartTime": 1529280000.0,
            "EndTime": 1529449200.0,
            "AttackVectors": [
                {
                    "VectorType": "SYN_FLOOD"
                }
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的查看 DDoS 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAttacks。

`list-protections`

以下代码示例演示了如何使用 `list-protections`。

Amazon CLI

从 Amazon Shield Advanced 中检索保护摘要

以下 `list-protections` 示例检索为该账户启用的保护的摘要。

```
aws shield list-protections
```

输出：

```
{
    "Protections": [
        {
            "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Name": "Protection for CloudFront distribution",
            "ResourceArn": "arn:aws:cloudfront::123456789012:distribution/
E198WC25FXOWY8"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的指定要保护的资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListProtections。

update-emergency-contact-settings

以下代码示例演示了如何使用 update-emergency-contact-settings。

Amazon CLI

定义 DRT 存档中的紧急电子邮件地址

以下 update-emergency-contact-settings 示例定义 DRT 在响应可疑攻击时应联系的两个电子邮件地址。

```
aws shield update-emergency-contact-settings \
        --emergency-contact-list EmailAddress=ops@example.com EmailAddress=ddos-
notifications@example.com
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的 Amazon Shield 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateEmergencyContactSettings。

update-subscription

以下代码示例演示了如何使用 update-subscription。

Amazon CLI

修改账户的 Amazon Shield Advanced 订阅

以下 update-subscription 示例为该账户启用 Amazon Shield Advanced 订阅的自动续订。

```
aws shield update-subscription \
    --auto-renew ENABLED
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Shield Advanced 开发人员指南》中的 Amazon Shield 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSubscription。

使用 Amazon CLI 的 Signer 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Signer 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

cancel-signing-profile

以下代码示例演示了如何使用 cancel-signing-profile。

Amazon CLI

删除签名配置文件

以下 cancel-signing-profile 示例从 Amazon Signer 中移除现有签名配置文件。

```
aws signer cancel-signing-profile \
    --profile-name MyProfile1
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelSigningProfile。

describe-signing-job

以下代码示例演示了如何使用 describe-signing-job。

Amazon CLI

显示签名任务的详细信息

以下 describe-signing-job 示例显示关于指定签名任务的详细信息。

```
aws signer describe-signing-job \
    --job-id 2065c468-73e2-4385-a6c9-0123456789abc
```

输出：

```
{
    "status": "Succeeded",
    "completedAt": 1568412037,
    "platformId": "AmazonFreeRTOS-Default",
    "signingMaterial": {
        "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
    },
    "statusReason": "Signing Succeeded",
    "jobId": "2065c468-73e2-4385-a6c9-0123456789abc",
    "source": {
        "s3": {
            "version": "PNyFaUTgsQh5ZdMCcoCe6pT1gOpgB_M4",
            "bucketName": "signer-source",
            "key": "MyCode.rb"
```

```
        }
    },
    "profileName": "MyProfile2",
    "signedObject": {
        "s3": {
            "bucketName": "signer-destination",
            "key": "signed-2065c468-73e2-4385-a6c9-0123456789abc"
        }
    },
    "requestedBy": "arn:aws:iam::123456789012:user/maria",
    "createdAt": 1568412036
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSigningJob。

get-signing-platform

以下代码示例演示了如何使用 get-signing-platform。

Amazon CLI

显示签名平台的详细信息

以下 get-signing-platform 示例显示关于指定签名平台的详细信息。

```
aws signer get-signing-platform \
    --platform-id AmazonFreeRTOS-TI-CC3220SF
```

输出：

```
{
    "category": "AWS",
    "displayName": "Amazon FreeRTOS SHA1-RSA CC3220SF-Format",
    "target": "SHA1-RSA-TISHA1",
    "platformId": "AmazonFreeRTOS-TI-CC3220SF",
    "signingConfiguration": {
        "encryptionAlgorithmOptions": {
            "defaultValue": "RSA",
            "allowedValues": [
                "RSA"
            ]
        },
        "hashAlgorithmOptions": {
```

```
            "defaultValue": "SHA1",
            "allowedValues": [
                "SHA1"
            ]
        }
    },
    "maxSizeInMB": 16,
    "partner": "AmazonFreeRTOS",
    "signingImageFormat": {
        "defaultFormat": "JSONEmbedded",
        "supportedFormats": [
            "JSONEmbedded"
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSigningPlatform。

get-signing-profile

以下代码示例演示了如何使用 get-signing-profile。

Amazon CLI

显示签名配置文件的详细信息

以下 get-signing-profile 示例显示关于指定签名配置文件的详细信息。

```
aws signer get-signing-profile \
    --profile-name MyProfile3
```

输出：

```
{
    "platformId": "AmazonFreeRTOS-TI-CC3220SF",
    "profileName": "MyProfile3",
    "status": "Active",
    "signingMaterial": {
        "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSigningProfile。

list-signing-jobs

以下代码示例演示了如何使用 list-signing-jobs。

Amazon CLI

列出所有签名任务

以下 list-signing-jobs 示例显示账户所有签名任务的详细信息。

```
aws signer list-signing-jobs
```

在此示例中，返回了两个任务，一个成功，一个失败。

```
{
    "jobs": [
        {
            "status": "Succeeded",
            "signingMaterial": {
                "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
            },
            "jobId": "2065c468-73e2-4385-a6c9-0123456789abc",
            "source": {
                "s3": {
                    "version": "PNyFaUTgsQh5ZdMCcoCe6pT1gOpgB_M4",
                    "bucketName": "signer-source",
                    "key": "MyCode.rb"
                }
            },
            "signedObject": {
                "s3": {
                    "bucketName": "signer-destination",
                    "key": "signed-2065c468-73e2-4385-a6c9-0123456789abc"
                }
            },
            "createdAt": 1568412036
        },
        {
            "status": "Failed",
            "source": {
```

```
                "s3": {
                    "version": "PNyFaUTgsQh5ZdMCcoCe6pT1gOpgB_M4",
                    "bucketName": "signer-source",
                    "key": "MyOtherCode.rb"
                }
            },
            "signingMaterial": {
                "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
            },
            "createdAt": 1568402690,
            "jobId": "74d9825e-22fc-4a0d-b962-0123456789abc"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSigningJobs。

list-signing-platforms

以下代码示例演示了如何使用 list-signing-platforms。

Amazon CLI

列出所有签名平台

以下 list-signing-platforms 示例显示所有可用签名平台的详细信息。

```
aws signer list-signing-platforms
```

输出：

```
{
    "platforms": [
        {
            "category": "AWS",
            "displayName": "AWS IoT Device Management SHA256-ECDSA ",
            "target": "SHA256-ECDSA",
            "platformId": "AWSIoTDeviceManagement-SHA256-ECDSA",
            "signingConfiguration": {
                "encryptionAlgorithmOptions": {
                    "defaultValue": "ECDSA",
                    "allowedValues": [
```

```
                    "ECDSA"
                ]
            },
            "hashAlgorithmOptions": {
                "defaultValue": "SHA256",
                "allowedValues": [
                    "SHA256"
                ]
            }
        },
        "maxSizeInMB": 2048,
        "partner": "AWSIoTDeviceManagement",
        "signingImageFormat": {
            "defaultFormat": "JSONDetached",
            "supportedFormats": [
                "JSONDetached"
            ]
        }
    },
    {
        "category": "AWS",
        "displayName": "Amazon FreeRTOS SHA1-RSA CC3220SF-Format",
        "target": "SHA1-RSA-TISHA1",
        "platformId": "AmazonFreeRTOS-TI-CC3220SF",
        "signingConfiguration": {
            "encryptionAlgorithmOptions": {
                "defaultValue": "RSA",
                "allowedValues": [
                    "RSA"
                ]
            },
            "hashAlgorithmOptions": {
                "defaultValue": "SHA1",
                "allowedValues": [
                    "SHA1"
                ]
            }
        },
        "maxSizeInMB": 16,
        "partner": "AmazonFreeRTOS",
        "signingImageFormat": {
            "defaultFormat": "JSONEmbedded",
            "supportedFormats": [
                "JSONEmbedded"
```

```
                ]
            }
        },
        {
            "category": "AWS",
            "displayName": "Amazon FreeRTOS SHA256-ECDSA",
            "target": "SHA256-ECDSA",
            "platformId": "AmazonFreeRTOS-Default",
            "signingConfiguration": {
                "encryptionAlgorithmOptions": {
                    "defaultValue": "ECDSA",
                    "allowedValues": [
                        "ECDSA"
                    ]
                },
                "hashAlgorithmOptions": {
                    "defaultValue": "SHA256",
                    "allowedValues": [
                        "SHA256"
                    ]
                }
            },
            "maxSizeInMB": 16,
            "partner": "AmazonFreeRTOS",
            "signingImageFormat": {
                "defaultFormat": "JSONEmbedded",
                "supportedFormats": [
                    "JSONEmbedded"
                ]
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSigningPlatforms。

`list-signing-profiles`

以下代码示例演示了如何使用 `list-signing-profiles`。

Amazon CLI

列出所有签名配置文件

以下 list-signing-profiles 示例显示账户所有签名配置文件的详细信息。

```
aws signer list-signing-profiles
```

输出：

```
{
    "profiles": [
        {
            "platformId": "AmazonFreeRTOS-TI-CC3220SF",
            "profileName": "MyProfile4",
            "status": "Active",
            "signingMaterial": {
                "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
            }
        },
        {
            "platformId": "AWSIoTDeviceManagement-SHA256-ECDSA",
            "profileName": "MyProfile5",
            "status": "Active",
            "signingMaterial": {
                "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSigningProfiles。

put-signing-profile

以下代码示例演示了如何使用 put-signing-profile。

Amazon CLI

创建签名配置文件

以下 put-signing-profile 示例使用指定证书和平台创建签名配置文件。

```
aws signer put-signing-profile \
```

```
    --profile-name MyProfile6 \
    --signing-material certificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc \
    --platform AmazonFreeRTOS-TI-CC3220SF
```

输出：

```
{
    "arn": "arn:aws:signer:us-west-2:123456789012:/signing-profiles/MyProfile6"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutSigningProfile。

start-signing-job

以下代码示例演示了如何使用 start-signing-job。

Amazon CLI

启动签名任务

以下 start-signing-job 示例对在指定源中找到的代码执行签名任务。它使用指定的配置文件进行签名，并将签名的代码放置在指定目的地。

```
aws signer start-signing-job \
    --source 's3={bucketName=signer-
source,key=MyCode.rb,version=PNyFaUTgsQh5ZdMCcoCe6pT1gOpgB_M4}' \
    --destination 's3={bucketName=signer-destination,prefix=signed-}' \
    --profile-name MyProfile7
```

输出是签名任务的 ID。

```
{
    "jobId": "2065c468-73e2-4385-a6c9-0123456789abc"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSigningJob。

使用 Amazon CLI 的 Snowball Edge 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Snowball Edge 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`get-snowball-usage`

以下代码示例演示了如何使用 `get-snowball-usage`。

Amazon CLI

获取有关账户 Snowball 服务限制的信息

以下 `get-snowball-usage` 示例显示有关您账户 Snowball 服务限制的信息，以及您账户已使用的 Snowball 的数量。

```
aws snowball get-snowball-usage
```

输出：

```
{
    "SnowballLimit": 1,
    "SnowballsInUse": 0
}
```

有关更多信息，请参阅《Amazon Snowball 开发人员指南》中的 Amazon Snowball Edge 限制。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSnowballUsage。

`list-jobs`

以下代码示例演示了如何使用 `list-jobs`。

Amazon CLI

列出你账户中的当前 Snowball 任务

以下 `list-jobs` 示例显示一个 `JobListEntry` 对象数组。在此示例中，仅列出了单个任务。

```
aws snowball list-jobs
```

输出：

```
{
    "JobListEntries": [
        {
            "CreationDate": 2016-09-27T14:50Z,
             "Description": "Important Photos 2016-08-11",
             "IsMaster": TRUE,
             "JobId": "ABCd1e324fe-022f-488e-a98b-3b0566063db1",
             "JobState": "Complete",
             "JobType": "IMPORT",
             "SnowballType": "EDGE"
        }
   ]
}
```

有关更多信息，请参阅《Amazon Snowball 开发人员指南》中的 Amazon Snowball Edge 设备的任务。

- 有关 API 的详细信息，请参阅《Amazon CLI 命令参考》中的 ListJobs。

使用Amazon CLI的 Amazon SNS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon SNS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作
- 场景

操作

`add-permission`

以下代码示例演示了如何使用 `add-permission`。

Amazon CLI

向主题添加权限

以下 `add-permission` 示例为 Amazon 账户 987654321098 添加在 Amazon 账户 123456789012 下使用指定主题的 `Publish` 操作的权限。

```
aws sns add-permission \
    --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --label Publish-Permission \
    --aws-account-id 987654321098 \
    --action-name Publish
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddPermission。

`check-if-phone-number-is-opted-out`

以下代码示例演示了如何使用 `check-if-phone-number-is-opted-out`。

Amazon CLI

检查电话号码的 SMS 消息退出

以下 check-if-phone-number-is-opted-out 示例将检查指定电话号码是否退出了当前 Amazon 账户的 SMS 消息接收。

```
aws sns check-if-phone-number-is-opted-out \
    --phone-number +1555550100
```

输出：

```
{
    "isOptedOut": false
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CheckIfPhoneNumberIsOptedOut。

confirm-subscription

以下代码示例演示了如何使用 confirm-subscription。

Amazon CLI

确认订阅

以下 confirm-subscription 命令将完成订阅名为 my-topic 的 SNS 主题时启动的确认过程。--token 参数来自发送到订阅调用中指定的通知端点的确认消息。

```
aws sns confirm-subscription \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \
    --
token 2336412f37fb687f5d51e6e241d7700ae02f7124d8268910b858cb4db727ceeb2474bb937929d3bdd7ce5d
```

输出：

```
{
    "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ConfirmSubscription。

create-platform-application

以下代码示例演示了如何使用 create-platform-application。

Amazon CLI

创建平台应用程序

以下 create-platform-application 示例使用指定的平台凭证创建一个 Google Firebase 平台应用程序。

```
aws sns create-platform-application \
    --name MyApplication \
    --platform GCM \
    --attributes PlatformCredential=EXAMPLEabcd12345jklm67890stuv12345bcdef
```

输出：

```
{
    "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/GCM/
MyApplication"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePlatformApplication。

create-topic

以下代码示例演示了如何使用 create-topic。

Amazon CLI

创建 SNS 主题

以下 create-topic 示例将创建名为 my-topic 的 SNS 主题。

```
aws sns create-topic \
    --name my-topic
```

输出：

```
{
```

```
    "ResponseMetadata": {
        "RequestId": "1469e8d7-1642-564e-b85d-a19b4b341f83"
    },
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

有关更多信息，请参阅《Amazon Command Line Interface 用户指南》中的将 Amazon Command Line Interface与 Amazon SQS 和 Amazon SNS 结合使用。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateTopic。

delete-endpoint

以下代码示例演示了如何使用 delete-endpoint。

Amazon CLI

删除平台应用程序端点

以下 delete-endpoint 示例删除指定的平台应用程序端点。

```
aws sns delete-endpoint \
    --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteEndpoint。

delete-platform-application

以下代码示例演示了如何使用 delete-platform-application。

Amazon CLI

删除平台应用程序

以下 delete-platform-application 示例删除指定的平台应用程序。

```
aws sns delete-platform-application \
    --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/ADM/
MyApplication
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePlatformApplication。

delete-topic

以下代码示例演示了如何使用 delete-topic。

Amazon CLI

删除 SNS 主题

以下 delete-topic 示例将删除指定的 SNS 主题。

```
aws sns delete-topic \
    --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteTopic。

get-endpoint-attributes

以下代码示例演示了如何使用 get-endpoint-attributes。

Amazon CLI

列出平台应用程序端点的属性

以下 get-endpoint-attributes 示例列出指定平台应用程序端点的属性。

```
aws sns get-endpoint-attributes \
    --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234
```

输出：

```
{
    "Attributes": {
        "Enabled": "true",
        "Token": "EXAMPLE12345..."
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEndpointAttributes。

get-platform-application-attributes

以下代码示例演示了如何使用 get-platform-application-attributes。

Amazon CLI

列出平台应用程序的属性

以下 get-platform-application-attributes 示例列出指定平台应用程序的属性。

```
aws sns get-platform-application-attributes \
    --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/MPNS/
MyApplication
```

输出：

```
{
    "Attributes": {
        "Enabled": "true",
        "SuccessFeedbackSampleRate": "100"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPlatformApplicationAttributes。

get-sms-attributes

以下代码示例演示了如何使用 get-sms-attributes。

Amazon CLI

列出默认 SMS 消息属性

以下 get-sms-attributes 示例将列出发送 SMS 消息的默认属性。

```
aws sns get-sms-attributes
```

输出：

```
{
    "attributes": {
        "DefaultSenderID": "MyName"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetSMSAttributes。

get-subscription-attributes

以下代码示例演示了如何使用 get-subscription-attributes。

Amazon CLI

检索主题的订阅属性

以下 get-subscription-attributes 将显示指定订阅的属性。您可以从 list-subscriptions 命令输出中获取 subscription-arn。

```
aws sns get-subscription-attributes \
    --subscription-arn "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
```

输出：

```
{
    "Attributes": {
        "Endpoint": "my-email@example.com",
        "Protocol": "email",
        "RawMessageDelivery": "false",
        "ConfirmationWasAuthenticated": "false",
        "Owner": "123456789012",
        "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f",
        "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSubscriptionAttributes。

get-topic-attributes

以下代码示例演示了如何使用 get-topic-attributes。

Amazon CLI

检索主题的属性

以下 get-topic-attributes 示例将显示指定主题的属性。

```
aws sns get-topic-attributes \
    --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

输出：

```
{
    "Attributes": {
        "SubscriptionsConfirmed": "1",
        "DisplayName": "my-topic",
        "SubscriptionsDeleted": "0",
        "EffectiveDeliveryPolicy": "{\"http\":{\"defaultHealthyRetryPolicy\":
{\"minDelayTarget\":20,\"maxDelayTarget\":20,\"numRetries\":3,\"numMaxDelayRetries
\":0,\"numNoDelayRetries\":0,\"numMinDelayRetries\":0,\"backoffFunction\":\"linear
\"},\"disableSubscriptionOverrides\":false}}",
        "Owner": "123456789012",
        "Policy": "{\"Version\":\"2008-10-17\",\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"*\"},\"Action\":[\"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic
\",\"SNS:DeleteTopic\",\"SNS:GetTopicAttributes\",\"SNS:Publish\",
\"SNS:RemovePermission\",\"SNS:AddPermission\",\"SNS:SetTopicAttributes\"],
\"Resource\":\"arn:aws:sns:us-west-2:123456789012:my-topic\",\"Condition\":
{\"StringEquals\":{\"AWS:SourceOwner\":\"0123456789012\"}}}]}",
        "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
        "SubscriptionsPending": "0"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetTopicAttributes。

list-endpoints-by-platform-application

以下代码示例演示了如何使用 list-endpoints-by-platform-application。

Amazon CLI

列出平台应用程序的端点

以下 `list-endpoints-by-platform-application` 示例列出指定平台应用程序的端点和端点属性。

```
aws sns list-endpoints-by-platform-application \
    --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/
MyApplication
```

输出：

```
{
    "Endpoints": [
        {
            "Attributes": {
                "Token": "EXAMPLE12345...,
                "Enabled": "true"
            },
            "EndpointArn": "arn:aws:sns:us-west-2:123456789012:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListEndpointsByPlatformApplication。

list-phone-numbers-opted-out

以下代码示例演示了如何使用 `list-phone-numbers-opted-out`。

Amazon CLI

列出 SMS 消息退出

以下 `list-phone-numbers-opted-out` 示例将列出退出 SMS 消息接收的电话号码。

```
aws sns list-phone-numbers-opted-out
```

输出：

```
{
    "phoneNumbers": [
        "+15555550100"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListPhoneNumbersOptedOut。

list-platform-applications

以下代码示例演示了如何使用 list-platform-applications。

Amazon CLI

列出平台应用程序

以下 list-platform-applications 示例列出 ADM 和 MPNS 的平台应用程序。

```
aws sns list-platform-applications
```

输出：

```
{
    "PlatformApplications": [
        {
            "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/ADM/
MyApplication",
            "Attributes": {
                "SuccessFeedbackSampleRate": "100",
                "Enabled": "true"
            }
        },
        {
            "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/MPNS/
MyOtherApplication",
            "Attributes": {
                "SuccessFeedbackSampleRate": "100",
                "Enabled": "true"
```

```
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPlatformApplications。

list-subscriptions-by-topic

以下代码示例演示了如何使用 list-subscriptions-by-topic。

Amazon CLI

列出与主题关联的订阅

以下 list-subscriptions-by-topic 将检索与指定主题关联的 SNS 订阅列表。

```
aws sns list-subscriptions-by-topic \
    --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

输出：

```
{
    "Subscriptions": [
        {
            "Owner": "123456789012",
            "Endpoint": "my-email@example.com",
            "Protocol": "email",
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
            "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListSubscriptionsByTopic。

list-subscriptions

以下代码示例演示了如何使用 list-subscriptions。

Amazon CLI

列出 SNS 订阅

以下 `list-subscriptions` 示例将显示 Amazon 账户中的 SNS 订阅的列表。

```
aws sns list-subscriptions
```

输出：

```
{
    "Subscriptions": [
        {
            "Owner": "123456789012",
            "Endpoint": "my-email@example.com",
            "Protocol": "email",
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
            "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListSubscriptions。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

列出主题的标签

以下 `list-tags-for-resource` 示例列出指定 Amazon SNS 主题的标签。

```
aws sns list-tags-for-resource \
    --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic
```

输出：

```
{
```

```
    "Tags": [
        {
            "Key": "Team",
            "Value": "Alpha"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-topics`

以下代码示例演示了如何使用 `list-topics`。

Amazon CLI

列出 SNS 主题

以下 `list-topics` 示例将列出 Amazon 账户中的所有 SNS 主题。

```
aws sns list-topics
```

输出：

```
{
    "Topics": [
        {
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListTopics。

`opt-in-phone-number`

以下代码示例演示了如何使用 `opt-in-phone-number`。

Amazon CLI

选择接收 SMS 消息

以下 opt-in-phone-number 示例选择指定的电话号码接收 SMS 消息。

```
aws sns opt-in-phone-number \
    --phone-number +15555550100
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 OptInPhoneNumber。

publish

以下代码示例演示了如何使用 publish。

Amazon CLI

示例 1：向主题发布消息

以下 publish 示例将指定消息发布到指定 SNS 主题。该消息来自一个文本文件，您可以在该文件中包含换行符。

```
aws sns publish \
    --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic" \
    --message file://message.txt
```

的内容message.txt：

```
Hello World
Second Line
```

输出：

```
{
    "MessageId": "123a45b6-7890-12c3-45d6-111122223333"
}
```

示例 2：向电话号码发布 SMS 消息

以下 publish 示例将消息 Hello world! 发布到电话号码 +1-555-555-0100。

```
aws sns publish \
    --message "Hello world!" \
```

```
    --phone-number +1-555-555-0100
```

输出：

```
{
    "MessageId": "123a45b6-7890-12c3-45d6-333322221111"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 Publish。

put-data-protection-policy

以下代码示例演示了如何使用 put-data-protection-policy。

Amazon CLI

设置数据保护策略

示例 1：拒绝发布者发布带有 CreditCardNumber 的消息

以下 put-data-protection-policy 示例拒绝发布者发布带有 CreditCardNumber 的消息。

```
aws sns put-data-protection-policy \
    --resource-arn arn:aws:sns:us-east-1:123456789012:mytopic \
    --data-protection-policy "{\"Name\":\"data_protection_policy\",\"Description
\":\"Example data protection policy\",\"Version\":\"2021-06-01\",\"Statement
\":[{\"DataDirection\":\"Inbound\",\"Principal\":[\"*\"],\"DataIdentifier\":
[\"arn:aws:dataprotection::aws:data-identifier/CreditCardNumber\"],\"Operation\":
{\"Deny\":{}}}]}"
```

此命令不生成任何输出。

示例 2：从文件加载参数

以下 put-data-protection-policy 将从文件加载参数。

```
aws sns put-data-protection-policy \
    --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --data-protection-policy file://policy.json
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutDataProtectionPolicy。

remove-permission

以下代码示例演示了如何使用 remove-permission。

Amazon CLI

从主题中移除权限

以下 remove-permission 示例从指定主题中移除权限 Publish-Permission。

```
aws sns remove-permission \
    --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --label Publish-Permission
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemovePermission。

set-endpoint-attributes

以下代码示例演示了如何使用 set-endpoint-attributes。

Amazon CLI

设置端点属性

以下 set-endpoint-attributes 示例禁用指定的平台应用程序端点。

```
aws sns set-endpoint-attributes \
    --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234 \
    --attributes Enabled=false
```

输出：

```
{
    "Attributes": {
        "Enabled": "false",
        "Token": "EXAMPLE12345..."
```

```
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetEndpointAttributes。

set-platform-application-attributes

以下代码示例演示了如何使用 set-platform-application-attributes。

Amazon CLI

设置平台应用程序属性

以下 set-platform-application-attributes 示例会将指定平台应用程序的 EventDeliveryFailure 属性设置为指定 Amazon SNS 主题的 ARN。

```
aws sns set-platform-application-attributes \
    --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/
MyApplication \
    --attributes EventDeliveryFailure=arn:aws:sns:us-
west-2:123456789012:AnotherTopic
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetPlatformApplicationAttributes。

set-sms-attributes

以下代码示例演示了如何使用 set-sms-attributes。

Amazon CLI

设置 SMS 消息属性

以下 set-sms-attributes 示例将 SMS 消息的默认发件人 ID 设置为 MyName。

```
aws sns set-sms-attributes \
    --attributes DefaultSenderID=MyName
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SetSMSAttributes。

set-subscription-attributes

以下代码示例演示了如何使用 set-subscription-attributes。

Amazon CLI

设置订阅属性

以下 set-subscription-attributes 示例将 RawMessageDelivery 属性设置为 SQS 订阅。

```
aws sns set-subscription-attributes \
    --subscription-arn arn:aws:sns:us-east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
    --attribute-name RawMessageDelivery \
    --attribute-value true
```

此命令不生成任何输出。

以下 set-subscription-attributes 示例将 FilterPolicy 属性设置为 SQS 订阅。

```
aws sns set-subscription-attributes \
    --subscription-arn arn:aws:sns:us-east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
    --attribute-name FilterPolicy \
    --attribute-value "{ \"anyMandatoryKey\": [\"any\", \"of\", \"these\"] }"
```

此命令不生成任何输出。

以下 set-subscription-attributes 示例从 SQS 订阅中移除 FilterPolicy 属性。

```
aws sns set-subscription-attributes \
    --subscription-arn arn:aws:sns:us-east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \
    --attribute-name FilterPolicy \
    --attribute-value "{}"
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SetSubscriptionAttributes。

set-topic-attributes

以下代码示例演示了如何使用 set-topic-attributes。

Amazon CLI

为主题设置属性

以下 set-topic-attributes 示例为指定主题设置 DisplayName 属性。

```
aws sns set-topic-attributes \
    --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --attribute-name DisplayName \
    --attribute-value MyTopicDisplayName
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SetTopicAttributes。

subscribe

以下代码示例演示了如何使用 subscribe。

Amazon CLI

订阅主题

以下 subscribe 命令将电子邮件地址订阅到指定主题。

```
aws sns subscribe \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \
    --protocol email \
    --notification-endpoint my-email@example.com
```

输出：

```
{
    "SubscriptionArn": "pending confirmation"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Subscribe。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

为主题添加标签

以下 tag-resource 示例将元数据标签添加到指定 Amazon SNS 主题。

```
aws sns tag-resource \
    --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --tags Key=Team,Value=Alpha
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

unsubscribe

以下代码示例演示了如何使用 unsubscribe。

Amazon CLI

从主题取消订阅

以下 unsubscribe 示例将从主题删除指定的订阅。

```
aws sns unsubscribe \
    --subscription-arn arn:aws:sns:us-west-2:0123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Unsubscribe。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

从主题中移除标签

以下 `untag-resource` 示例从指定的 Amazon SNS 主题中移除任何带有指定键的标签。

```
aws sns untag-resource \
    --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \
    --tag-keys Team
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

场景

为推送通知创建平台终端节点

以下代码示例演示如何为 Amazon SNS 推送通知创建平台端点。

Amazon CLI

创建平台应用程序端点

以下 `create-platform-endpoint` 示例使用指定令牌为指定平台应用程序创建端点。

```
aws sns create-platform-endpoint \
    --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/
MyApplication \
    --token EXAMPLE12345...
```

输出：

```
{
      "EndpointArn": "arn:aws:sns:us-west-2:1234567890:endpoint/GCM/
MyApplication/12345678-abcd-9012-efgh-345678901234"
}
```

使用Amazon CLI的 Amazon SQS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon SQS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-permission`

以下代码示例演示了如何使用 `add-permission`。

Amazon CLI

向队列添加权限

以下示例允许指定的 Amazon 账户向指定队列发送消息。

命令:

```
aws sqs add-permission --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --label SendMessagesFromMyQueue --aws-account-ids 12345EXAMPLE --actions SendMessage
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddPermission。

`cancel-message-move-task`

以下代码示例演示了如何使用 `cancel-message-move-task`。

Amazon CLI

取消消息移动任务

以下 `cancel-message-move-task` 示例取消指定的消息移动任务。

```
aws sqs cancel-message-move-task \
```

```
    --task-handle AQEB6nR4...HzlvZQ==
```

输出：

```
{
    "ApproximateNumberOfMessagesMoved": 102
}
```

有关更多信息，请参阅《开发人员指南》中的 Amazon SQS API 权限：操作和资源参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelMessageMoveTask。

change-message-visibility-batch

以下代码示例演示了如何使用 change-message-visibility-batch。

Amazon CLI

批量更改多条消息的超时可见性

此示例 2 条指定消息的超时可见性更改为 10 小时（10 小时 * 60 分钟 * 60 秒）。

命令:

```
aws sqs change-message-visibility-batch --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://change-message-
visibility-batch.json
```

输入文件（change-message-visibility-batch.json）：

```
[
  {
    "Id": "FirstMessage",
        "ReceiptHandle": "AQEBhz2q...Jf3kaw==",
        "VisibilityTimeout": 36000
  },
  {
    "Id": "SecondMessage",
        "ReceiptHandle": "AQEBkTUH...HifSnw==",
        "VisibilityTimeout": 36000
  }
]
```

输出：

```
{
  "Successful": [
    {
      "Id": "SecondMessage"
    },
    {
      "Id": "FirstMessage"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangeMessageVisibilityBatch。

change-message-visibility

以下代码示例演示了如何使用 change-message-visibility。

Amazon CLI

更改消息的超时可见性

此示例将指定消息的超时可见性更改为 10 小时（10 小时 * 60 分钟 * 60 秒）。

命令:

```
aws sqs change-message-visibility --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --receipt-handle AQEBTpyI...t6HyQg== --
visibility-timeout 36000
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ChangeMessageVisibility。

create-queue

以下代码示例演示了如何使用 create-queue。

Amazon CLI

创建队列

此示例使用指定名称创建队列，将消息保留期设置为 3 天（3 天 * 24 小时 * 60 分钟 * 60 秒），并将队列的死信队列设置为指定队列，其最大接收数量为 1,000 条消息。

命令:

```
aws sqs create-queue --queue-name MyQueue --attributes file://create-queue.json
```

输入文件（create-queue.json）：

```
{
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
east-1:80398EXAMPLE:MyDeadLetterQueue\",\"maxReceiveCount\":\"1000\"}",
  "MessageRetentionPeriod": "259200"
}
```

输出：

```
{
  "QueueUrl": "https://queue.amazonaws.com/80398EXAMPLE/MyQueue"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateQueue。

delete-message-batch

以下代码示例演示了如何使用 delete-message-batch。

Amazon CLI

批量删除多条消息

此示例将删除指定消息。

命令:

```
aws sqs delete-message-batch --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://delete-message-batch.json
```

输入文件（delete-message-batch.json）：

```
[
  {
        "Id": "FirstMessage",
        "ReceiptHandle": "AQEB1mgl...Z4GuLw=="
  },
  {
    "Id": "SecondMessage",
        "ReceiptHandle": "AQEBLsYM...VQubAA=="
  }
]
```

输出：

```
{
  "Successful": [
    {
      "Id": "FirstMessage"
    },
    {
      "Id": "SecondMessage"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMessageBatch。

delete-message

以下代码示例演示了如何使用 delete-message。

Amazon CLI

删除消息

此示例将删除指定消息。

命令:

```
aws sqs delete-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --receipt-handle AQEBRXTo...q2doVA==
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMessage。

delete-queue

以下代码示例演示了如何使用 delete-queue。

Amazon CLI

删除队列

此示例将删除指定队列。

命令:

```
aws sqs delete-queue --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewerQueue
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteQueue。

get-queue-attributes

以下代码示例演示了如何使用 get-queue-attributes。

Amazon CLI

获取队列的属性

此示例将获取指定队列的所有属性。

命令:

```
aws sqs get-queue-attributes --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names All
```

输出：

```
{
  "Attributes": {
    "ApproximateNumberOfMessagesNotVisible": "0",
    "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
east-1:80398EXAMPLE:MyDeadLetterQueue\",\"maxReceiveCount\":1000}",
    "MessageRetentionPeriod": "345600",
    "ApproximateNumberOfMessagesDelayed": "0",
    "MaximumMessageSize": "262144",
    "CreatedTimestamp": "1442426968",
    "ApproximateNumberOfMessages": "0",
    "ReceiveMessageWaitTimeSeconds": "0",
    "DelaySeconds": "0",
    "VisibilityTimeout": "30",
    "LastModifiedTimestamp": "1442426968",
    "QueueArn": "arn:aws:sqs:us-east-1:80398EXAMPLE:MyNewQueue"
  }
}
```

此示例仅获取指定队列的最大消息大小和可见性超时属性。

命令:

```
aws sqs get-queue-attributes --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyNewQueue --attribute-
names MaximumMessageSize VisibilityTimeout
```

输出：

```
{
  "Attributes": {
    "VisibilityTimeout": "30",
    "MaximumMessageSize": "262144"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueueAttributes。

get-queue-url

以下代码示例演示了如何使用 get-queue-url。

Amazon CLI

获取队列 URL

此示例将获取指定队列的 URL。

命令:

```
aws sqs get-queue-url --queue-name MyQueue
```

输出：

```
{
  "QueueUrl": "https://queue.amazonaws.com/80398EXAMPLE/MyQueue"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetQueueUrl。

`list-dead-letter-source-queues`

以下代码示例演示了如何使用 `list-dead-letter-source-queues`。

Amazon CLI

列出死信源队列

此示例列出与指定死信源队列关联的队列。

命令:

```
aws sqs list-dead-letter-source-queues --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

输出：

```
{
  "queueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDeadLetterSourceQueues。

list-message-move-tasks

以下代码示例演示了如何使用 list-message-move-tasks。

Amazon CLI

列出消息移动任务

以下 list-message-move-tasks 示例列出指定队列中最近的 2 个消息移动任务。

```
aws sqs list-message-move-tasks \
    --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue \
    --max-results 2
```

输出：

```
{
    "Results": [
        {
            "TaskHandle": "AQEB6nR4...HzlvZQ==",
            "Status": "RUNNING",
            "SourceArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1",
            "DestinationArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2",
            "MaxNumberOfMessagesPerSecond": 50,
            "ApproximateNumberOfMessagesMoved": 203,
            "ApproximateNumberOfMessagesToMove": 30,
            "StartedTimestamp": 1442428276921
        },

        {
            "Status": "COMPLETED",
            "SourceArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1",
            "DestinationArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2",
            "ApproximateNumberOfMessagesMoved": 29,
            "ApproximateNumberOfMessagesToMove": 0,
            "StartedTimestamp": 1342428272093
        }
    ]
}
```

有关更多信息，请参阅《开发人员指南》中的 Amazon SQS API 权限：操作和资源参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMessageMoveTasks。

`list-queue-tags`

以下代码示例演示了如何使用 `list-queue-tags`。

Amazon CLI

列出队列的所有成本分配标签

以下 `list-queue-tags` 示例显示与指定队列关联的所有成本分配标签。

```
aws sqs list-queue-tags \
    --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue
```

输出：

```
{
    "Tags": {
        "Team": "Alpha"
    }
}
```

有关更多信息，请参阅《Amazon Simple Queue Service 开发人员指南》中的列出成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueueTags。

`list-queues`

以下代码示例演示了如何使用 `list-queues`。

Amazon CLI

列出队列

此示例将列出所有队列。

命令:

```
aws sqs list-queues
```

输出：

```
{
  "QueueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/TestQueue1",
        "https://queue.amazonaws.com/80398EXAMPLE/TestQueue2"
  ]
}
```

此示例仅列出以“My”开头的队列。

命令:

```
aws sqs list-queues --queue-name-prefix My
```

输出：

```
{
  "QueueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue"
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListQueues。

purge-queue

以下代码示例演示了如何使用 purge-queue。

Amazon CLI

清除队列

此示例删除指定队列中的所有消息。

命令:

```
aws sqs purge-queue --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewQueue
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PurgeQueue。

receive-message

以下代码示例演示了如何使用 receive-message。

Amazon CLI

接收消息

此示例最多接收 10 条可用消息，返回所有可用属性。

命令:

```
aws sqs receive-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names All --message-attribute-names All --max-number-of-messages 10
```

输出：

```
{
  "Messages": [
    {
      "Body": "My first message.",
      "ReceiptHandle": "AQEBzbVv...fqNzFw==",
      "MD5OfBody": "1000f835...a35411fa",
      "MD5OfMessageAttributes": "9424c491...26bc3ae7",
      "MessageId": "d6790f8d-d575-4f01-bc51-40122EXAMPLE",
      "Attributes": {
        "ApproximateFirstReceiveTimestamp": "1442428276921",
        "SenderId": "AIDAIAZKMSNQ7TEXAMPLE",
        "ApproximateReceiveCount": "5",
        "SentTimestamp": "1442428276921"
      },
```

```
      "MessageAttributes": {
        "PostalCode": {
          "DataType": "String",
          "StringValue": "ABC123"
        },
        "City": {
          "DataType": "String",
          "StringValue": "Any City"
        }
      }
    }
  ]
}
```

此示例接收下一条可用消息，仅返回 SenderId 和 SentTimestamp 属性以及 PostalCode 消息属性。

命令:

```
aws sqs receive-message --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names SenderId SentTimestamp
 --message-attribute-names PostalCode
```

输出：

```
{
  "Messages": [
    {
      "Body": "My first message.",
      "ReceiptHandle": "AQEB6nR4...HzlvZQ==",
      "MD5OfBody": "1000f835...a35411fa",
      "MD5OfMessageAttributes": "b8e89563...e088e74f",
      "MessageId": "d6790f8d-d575-4f01-bc51-40122EXAMPLE",
      "Attributes": {
        "SenderId": "AIDAIAZKMSNQ7TEXAMPLE",
        "SentTimestamp": "1442428276921"
      },
      "MessageAttributes": {
        "PostalCode": {
          "DataType": "String",
          "StringValue": "ABC123"
        }
      }
```

```
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReceiveMessage。

`remove-permission`

以下代码示例演示了如何使用 `remove-permission`。

Amazon CLI

删除权限

此示例从指定队列中移除具有指定标签的权限。

命令:

```
aws sqs remove-permission --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --label SendMessagesFromMyQueue
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemovePermission。

`send-message-batch`

以下代码示例演示了如何使用 `send-message-batch`。

Amazon CLI

批量发送多条消息

此示例向指定队列发送 2 条具有指定消息正文、延迟时间和消息属性的消息。

命令:

```
aws sqs send-message-batch --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://send-message-batch.json
```

输入文件（send-message-batch.json）：

```
[
  {
    "Id": "FuelReport-0001-2015-09-16T140731Z",
        "MessageBody": "Fuel report for account 0001 on 2015-09-16 at 02:07:31 PM.",
        "DelaySeconds": 10,
        "MessageAttributes": {
          "SellerName": {
            "DataType": "String",
                "StringValue": "Example Store"
      },
          "City": {
        "DataType": "String",
        "StringValue": "Any City"
      },
          "Region": {
            "DataType": "String",
                "StringValue": "WA"
      },
          "PostalCode": {
            "DataType": "String",
                "StringValue": "99065"
          },
          "PricePerGallon": {
            "DataType": "Number",
                "StringValue": "1.99"
      }
        }
  },
  {
    "Id": "FuelReport-0002-2015-09-16T140930Z",
        "MessageBody": "Fuel report for account 0002 on 2015-09-16 at 02:09:30 PM.",
        "DelaySeconds": 10,
        "MessageAttributes": {
          "SellerName": {
            "DataType": "String",
                "StringValue": "Example Fuels"
      },
          "City": {
        "DataType": "String",
        "StringValue": "North Town"
      },
          "Region": {
```

```
            "DataType": "String",
                "StringValue": "WA"
      },
          "PostalCode": {
            "DataType": "String",
                "StringValue": "99123"
          },
          "PricePerGallon": {
            "DataType": "Number",
                "StringValue": "1.87"
      }
        }
  }
]
```

输出：

```
{
  "Successful": [
    {
      "MD5OfMessageBody": "203c4a38...7943237e",
      "MD5OfMessageAttributes": "10809b55...baf283ef",
      "Id": "FuelReport-0001-2015-09-16T140731Z",
      "MessageId": "d175070c-d6b8-4101-861d-adeb3EXAMPLE"
    },
    {
      "MD5OfMessageBody": "2cf0159a...c1980595",
      "MD5OfMessageAttributes": "55623928...ae354a25",
      "Id": "FuelReport-0002-2015-09-16T140930Z",
      "MessageId": "f9b7d55d-0570-413e-b9c5-a9264EXAMPLE"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendMessageBatch。

send-message

以下代码示例演示了如何使用 send-message。

Amazon CLI

发送邮件

此示例向指定队列发送一条具有指定消息正文、延迟时间和消息属性的消息。

命令:

```
aws sqs send-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --message-body "Information about the largest city in Any Region." --delay-seconds 10 --message-attributes file://send-message.json
```

输入文件（send-message.json）：

```
{
  "City": {
    "DataType": "String",
    "StringValue": "Any City"
  },
  "Greeting": {
    "DataType": "Binary",
    "BinaryValue": "Hello, World!"
  },
  "Population": {
    "DataType": "Number",
    "StringValue": "1250800"
  }
}
```

输出：

```
{
  "MD5OfMessageBody": "51b0a325...39163aa0",
  "MD5OfMessageAttributes": "00484c68...59e48f06",
  "MessageId": "da68f62c-0c07-4bee-bf5f-7e856EXAMPLE"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendMessage。

`set-queue-attributes`

以下代码示例演示了如何使用 `set-queue-attributes`。

Amazon CLI

设置队列属性

此示例将指定队列的传输延迟设置为 10 秒，最大消息大小为 128 KB（128 KB * 1,024 字节），消息保留期为 3 天（3 天 * 24 小时 * 60 分钟 * 60 秒），接收消息等待时间为 20 秒，默认可见性超时为 60 秒。此示例还将指定的死信队列与最大接收数 1,000 条消息相关联。

命令:

```
aws sqs set-queue-attributes --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewQueue --attributes file://set-queue-attributes.json
```

输入文件（set-queue-attributes.json）：

```
{
  "DelaySeconds": "10",
  "MaximumMessageSize": "131072",
  "MessageRetentionPeriod": "259200",
  "ReceiveMessageWaitTimeSeconds": "20",
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:80398EXAMPLE:MyDeadLetterQueue\",\"maxReceiveCount\":\"1000\"}",
  "VisibilityTimeout": "60"
}
```

输出：

```
None.
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SetQueueAttributes。

start-message-move-task

以下代码示例演示了如何使用 start-message-move-task。

Amazon CLI

示例 1：*启动消息移动任务*

以下 start-message-move-task 示例启动消息移动任务，将消息从指定的死信队列重新传输到源队列。

```
aws sqs start-message-move-task \
```

```
    --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue
```

输出：

```
{
    "TaskHandle": "AQEB6nR4...HzlvZQ=="
}
```

有关更多信息，请参阅《指南名称》中的主题标题。

示例 2：*以最高速率启动消息移动任务*

以下 start-message-move-task 示例启动消息移动任务，以每秒 50 条消息的最大速率将消息从指定的死信队列重新传输到指定的目标队列。

```
aws sqs start-message-move-task \
    --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1 \
    --destination-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2 \
    --max-number-of-messages-per-second 50
```

输出：

```
{
    "TaskHandle": "AQEB6nR4...HzlvZQ=="
}
```

有关更多信息，请参阅《开发人员指南》中的 Amazon SQS API 权限：操作和资源参考。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartMessageMoveTask。

tag-queue

以下代码示例演示了如何使用 tag-queue。

Amazon CLI

向队列添加成本分配标签

以下 tag-queue 示例会将成本分配标签添加到指定的 Amazon SQS 队列。

```
aws sqs tag-queue \
```

```
    --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue \
    --tags Priority=Highest
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Queue Service 开发人员指南》中的添加成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagQueue。

untag-queue

以下代码示例演示了如何使用 untag-queue。

Amazon CLI

从队列中移除成本分配标签

以下 untag-queue 示例从指定的 Amazon SQS 队列中移除成本分配标签。

```
aws sqs untag-queue \
    --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue \
    --tag-keys "Priority"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Simple Queue Service 开发人员指南》中的添加成本分配标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagQueue。

使用 Amazon CLI 的 Storage Gateway 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Storage Gateway 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`describe-gateway-information`

以下代码示例演示了如何使用 `describe-gateway-information`。

Amazon CLI

描述网关

以下 `describe-gateway-information` 命令将返回有关指定网关的元数据。在命令中使用网关的 Amazon 资源名称（ARN）指定要描述的网关。

此示例指定账户 123456789012 中 ID 为 sgw-12A3456B 的网关：

```
aws storagegateway describe-gateway-information --gateway-
arn "arn:aws:storagegateway:us-west-2:123456789012:gateway/sgw-12A3456B"
```

此命令将输出一个 JSON 数据块，其中包含有关网关的元数据，如名称、网络接口、已配置时区和状态（网关运行与否）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGatewayInformation。

`list-file-shares`

以下代码示例演示了如何使用 `list-file-shares`。

Amazon CLI

列出文件共享

以下 `command-name` 示例列出您的 Amazon 账户中可用的小组件。

```
aws storagegateway list-file-shares \
    --gateway-arn arn:aws:storagegateway:us-east-1:209870788375:gateway/sgw-FB02E292
```

输出：

```
{
    "FileShareInfoList": [
        {
            "FileShareType": "NFS",
```

```
            "FileShareARN": "arn:aws:storagegateway:us-east-1:111122223333:share/
share-2FA12345",
            "FileShareId": "share-2FA12345",
            "FileShareStatus": "AVAILABLE",
            "GatewayARN": "arn:aws:storagegateway:us-east-1:111122223333:gateway/
sgw-FB0AAAAA"
        }
    ],
    "Marker": null
}
```

有关更多信息，请参阅《Amazon Storage Gateway 服务 API 参考》中的 ListFileShares。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListFileShares。

`list-gateways`

以下代码示例演示了如何使用 `list-gateways`。

Amazon CLI

列出账户的网关

以下 `list-gateways` 命令将列出为账户定义的所有网关：

```
aws storagegateway list-gateways
```

此命令将输出包含网关 Amazon 资源名称（ARN）列表的 JSON 数据块。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGateways。

`list-volumes`

以下代码示例演示了如何使用 `list-volumes`。

Amazon CLI

列出为网关配置的卷

以下 `list-volumes` 命令将返回为指定网关配置的卷列表。在命令中使用网关的 Amazon 资源名称（ARN）指定要描述的网关。

此示例指定账户 123456789012 中 ID 为 sgw-12A3456B 的网关：

```
aws storagegateway list-volumes --gateway-arn "arn:aws:storagegateway:us-
west-2:123456789012:gateway/sgw-12A3456B"
```

此命令将输出一个 JSON 数据块，其中包含带有每个卷的类型和 ARN 的卷列表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVolumes。

`refresh-cache`

以下代码示例演示了如何使用 `refresh-cache`。

Amazon CLI

刷新文件共享缓存

以下 `refresh-cache` 示例刷新指定文件共享的缓存。

```
aws storagegateway refresh-cache \
    --file-share-arn arn:aws:storagegateway:us-east-1:111122223333:share/
share-2FA12345
```

输出：

```
{
    "FileShareARN": "arn:aws:storagegateway:us-east-1:111122223333:share/
share-2FA12345",
    "NotificationId": "4954d4b1-abcd-ef01-1234-97950a7d3483"
}
```

有关更多信息，请参阅《Amazon Storage Gateway 服务 API 参考》中的 ListFileShares。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RefreshCache。

Amazon STS使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon STS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

assume-role-with-saml

以下代码示例演示了如何使用 assume-role-with-saml。

Amazon CLI

获取通过 SAML 进行身份验证的角色的短期凭证

以下 assume-role-with-saml 命令将检索 IAM 角色 TestSaml 的一组短期凭证。此示例中的请求是通过使用身份提供商在您验证身份时提供的 SAML 断言进行验证的。

```
aws sts assume-role-with-saml \
    --role-arn arn:aws:iam::123456789012:role/TestSaml \
    --principal-arn arn:aws:iam::123456789012:saml-provider/SAML-test \
    --saml-
assertion "VERYLONGENCODEDASSERTIONEXAMPLExzYW1sOkF1ZGllbmNlPmJsYW5rPC9zYW1sOkF1ZGllbmNlPjwv
+PHNhbWw6TmFtZUlEIEZvcm1hdD0idXJuOm9hc2lzOm5hbWVzOnRjOlNBTUw6Mi4wOm5hbWVpZC1mb3JtYXQ6dHJhbnN
+PHNhbWw6U3ViamVjdENvbmZpcm1hdGlvbiBNZXRob2Q9InVybjpvYXNpczpuYW1lczp0YzpTQU1MOjIuMDpjbTpiZWF
```

输出：

```
{
    "Issuer": "https://integ.example.com/idp/shibboleth</Issuer",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/TestSaml",
        "AssumedRoleId": "ARO456EXAMPLE789:TestSaml"
    },
    "Credentials": {
        "AccessKeyId": "ASIAV3ZUEFP6EXAMPLE",
        "SecretAccessKey": "8P+SQvWIuLnKhh8d++jpw0nNmQRBZvNEXAMPLEKEY",
        "SessionToken": "IQoJb3JpZ2luX2VjEOz////////////////////
wEXAMPLEtMSJHMEUCIDoKK3JH9uGQE1z0sINr5M4jk
+Na8KHDcCYRVjJCZEvOAiEA3OvJGtw1EcViOleS2vhs8VdCKFJQWPQrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburED
```

```
+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
+scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==",
        "Expiration": "2019-11-01T20:26:47Z"
    },
    "Audience": "https://signin.aws.amazon.com/saml",
    "SubjectType": "transient",
    "PackedPolicySize": "6",
    "NameQualifier": "SbdGOnUkh1i4+EXAMPLExL/jEvs=",
    "Subject": "SamlExample"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的请求临时安全凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssumeRoleWithSaml。

assume-role-with-web-identity

以下代码示例演示了如何使用 assume-role-with-web-identity。

Amazon CLI

获取通过 Web 身份（OAuth 2."0）进行身份验证的角色的短期凭证

以下 assume-role-with-web-identity 命令将检索 IAM 角色 app1 的一组短期凭证。使用由指定 Web 身份提供者提供的 Web 身份令牌对请求进行身份验证。两个附加策略应用于会话，以进一步限制用户可以执行的操作。返回的凭证在生成一个小时后过期。

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/webidentitydemopolicy2"
 \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCzyoC
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

输出：

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A",
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    },
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的请求临时安全凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssumeRoleWithWebIdentity。

assume-role

以下代码示例演示了如何使用 assume-role。

Amazon CLI

要代入角色

以下 assume-role 命令将检索 IAM 角色 s3-access-example 的一组短期凭证。

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

输出：

```
{
    "AssumedRoleUser": {
```

```
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-access-
example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/qwjzP2iEXAMPLEbw/
m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8BRi2
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

该命令的输出包含访问密钥、私有密钥和会话令牌，您可以使用它们对 Amazon 进行身份验证。

要使用 Amazon CLI，则可以设置与角色关联的命名配置文件。使用配置文件时，Amazon CLI 将调用 assume-role 并为您管理凭证。有关更多信息，请参阅《Amazon CLI 用户指南》中的在 Amazon CLI 中使用 IAM 角色。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssumeRole。

`assume-root`

以下代码示例演示了如何使用 `assume-root`。

Amazon CLI

启动特权会话

以下 `assume-root` 命令检索一组短期凭证，您可以使用这些凭证移除组织中成员账户的配置错误的 Amazon S3 存储桶策略。

```
aws sts assume-root \
    --duration-seconds 900 \
    --target-principal 111122223333 \
    --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy
```

输出：

```
{
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/qwjzP2iEXAMPLEbw/
m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8BRi2
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2024-11-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    },
    "SourceIdentity": "Alice",
}
```

该命令的输出包含访问密钥、私有密钥和会话令牌，您可以使用它们在成员账户中执行特权操作。有关更多信息，请参阅《Amazon IAM 用户指南》中的在 Amazon Organizations 成员账户上执行特权任务。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 AssumeRoot。

decode-authorization-message

以下代码示例演示了如何使用 decode-authorization-message。

Amazon CLI

解码为响应请求而返回的编码授权消息

以下 decode-authorization-message 示例从为响应 Amazon Web Services 请求而返回的编码消息中解码有关请求授权状态的其他信息。

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3Ewi
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK44E
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgoqCz
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9zeB
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKEXAM
```

输出：


```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,\"matchedStatements
\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect\":\"DENY\",\"principals
\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE\"}]},\"principalGroups
\":{\"items\":[]},\"actions\":{\"items\":[{\"value\":\"ec2:RunInstances
\"}]},\"resources\":{\"items\":[{\"value\":\"*\"}]},\"conditions\":{\"items
\":[]}}]},\"failures\":{\"items\":[]},\"context\":{\"principal\":{\"id\":
\"AROA123456789EXAMPLE:Ana\",\"arn\":\"arn:aws:sts::111122223333:assumed-role/
Developer/Ana\"},\"action\":\"RunInstances\",\"resource\":\"arn:aws:ec2:us-
east-1:111122223333:instance/*\",\"conditions\":{\"items\":[{\"key\":
\"ec2:MetadataHttpPutResponseHopLimit\",\"values\":{\"items\":[{\"value\":
\"2\"}]}},{\"key\":\"ec2:InstanceMarketType\",\"values\":{\"items\":[{\"value
\":\"on-demand\"}]}},{\"key\":\"aws:Resource\",\"values\":{\"items\":[{\"value
\":\"instance/*\"}]}},{\"key\":\"aws:Account\",\"values\":{\"items\":[{\"value
\":\"111122223333\"}]}},{\"key\":\"ec2:AvailabilityZone\",\"values\":{\"items\":
[{\"value\":\"us-east-1f\"}]}},{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items
\":[{\"value\":\"false\"}]}},{\"key\":\"ec2:IsLaunchTemplateResource\",\"values
\":{\"items\":[{\"value\":\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":
{\"items\":[{\"value\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values
\":{\"items\":[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values
\":{\"items\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":
{\"items\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items
\":[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items\":
[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items\":
[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":[{\"value\":
\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```


有关更多信息，请参阅《Amazon IAM 用户指南》中的策略评估逻辑。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DecodeAuthorizationMessage。

get-caller-identity

以下代码示例演示了如何使用 get-caller-identity。

Amazon CLI

获取有关当前 IAM 身份的详细信息

以下 `get-caller-identity` 命令将显示有关用于验证请求的 IAM 身份的信息。调用方是 IAM 用户。

```
aws sts get-caller-identity
```

输出：

```
{
    "UserId": "AIDASAMPLEUSERID",
    "Account": "123456789012",
    "Arn": "arn:aws:iam::123456789012:user/DevAdmin"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCallerIdentity。

`get-federation-token`

以下代码示例演示了如何使用 `get-federation-token`。

Amazon CLI

使用 IAM 用户访问密钥凭证返回一组临时的安全凭证

以下 `get-federation-token` 示例返回用户的一组临时安全凭证（由访问密钥 ID、秘密访问密钥和安全令牌组成）。您必须使用 IAM 用户的长期安全凭证调用 `GetFederationToken` 操作。

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

的内容`myfile.json`：

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

输出：

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
```

```
    },
    "PackedPolicySize": 36
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的请求临时安全凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFederationToken。

get-session-token

以下代码示例演示了如何使用 get-session-token。

Amazon CLI

要获取 IAM 身份的一组短期凭证

以下 get-session-token 命令将检索进行调用的 IAM 身份的一组短期凭证。生成的凭证可用于策略要求多重身份验证（MFA）的请求。凭证在生成 15 分钟后过期。

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

输出：

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

有关更多信息，请参阅《Amazon IAM 用户指南》中的请求临时安全凭证。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSessionToken。

Amazon Web Services 支持使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Web Services 支持 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-attachments-to-set`

以下代码示例演示了如何使用 `add-attachments-to-set`。

Amazon CLI

向集合添加附件

以下 `add-attachments-to-set` 示例向集合添加一张图片，您可以在账户 Amazon 中为支持案例指定该图片。

```
aws support add-attachments-to-set \
    --attachment-set-id "as-2f5a6faa2a4a1e600-mu-nk5xQlBr70-
G1cUos5LZkd38KOAHZa9BMDVzNEXAMPLE" \
    --attachments fileName=troubleshoot-screenshot.png,data=base64-encoded-string
```

输出：

```
{
    "attachmentSetId": "as-2f5a6faa2a4a1e600-mu-nk5xQlBr70-
G1cUos5LZkd38KOAHZa9BMDVzNEXAMPLE",
    "expiryTime": "2020-05-14T17:04:40.790+0000"
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddAttachmentsToSet。

add-communication-to-case

以下代码示例演示了如何使用 add-communication-to-case。

Amazon CLI

向案例添加通信

以下 add-communication-to-case 示例向您 Amazon 账户中的支持案例添加通信。

```
aws support add-communication-to-case \
    --case-id "case-12345678910-2013-c4c1d2bf33c5cf47" \
    --communication-body "I'm attaching a set of images to this case." \
    --cc-email-addresses "myemail@example.com" \
    --attachment-set-id "as-2f5a6faa2a4a1e600-mu-nk5xQlBr70-
G1cUos5LZkd38KOAHZa9BMDVzNEXAMPLE"
```

输出：

```
{
    "result": true
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddCommunicationToCase。

create-case

以下代码示例演示了如何使用 create-case。

Amazon CLI

创建案例

以下 create-case 示例为您的 Amazon 账户创建了一个支持案例。

```
aws support create-case \
    --category-code "using-aws" \
    --cc-email-addresses "myemail@example.com" \
```

```
    --communication-body "I want to learn more about an AWS service." \
    --issue-type "technical" \
    --language "en" \
    --service-code "general-info" \
    --severity-code "low" \
    --subject "Question about my account"
```

输出：

```
{
    "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47"
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCase。

describe-attachment

以下代码示例演示了如何使用 describe-attachment。

Amazon CLI

描述附件

以下 describe-attachment 示例返回有关带指定 ID 的附件的信息。

```
aws support describe-attachment \
    --attachment-id "attachment-KBnjRNrePd9D6Jx0-Mm00xZuDEaL2JAj_0-
gJv9qqDooTipsz3V1Nb19rCfkZneeQeDPgp8X1iVJyHH7UuhZDdNeqGoduZsPrAhyMakqlc60-
iJjL5HqyYGiT1FG8EXAMPLE"
```

输出：

```
{
    "attachment": {
        "fileName": "troubleshoot-screenshot.png",
        "data": "base64-blob"
    }
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAttachment。

describe-cases

以下代码示例演示了如何使用 describe-cases。

Amazon CLI

描述案例

以下 describe-cases 示例返回有关您 Amazon 账户中指定支持案例的信息。

```
aws support describe-cases \
    --display-id "1234567890" \
    --after-time "2020-03-23T21:31:47.774Z" \
    --include-resolved-cases \
    --language "en" \
    --no-include-communications \
    --max-item 1
```

输出：

```
{
    "cases": [
        {
            "status": "resolved",
            "ccEmailAddresses": [],
            "timeCreated": "2020-03-23T21:31:47.774Z",
            "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47",
            "severityCode": "low",
            "language": "en",
            "categoryCode": "using-aws",
            "serviceCode": "general-info",
            "submittedBy": "myemail@example.com",
            "displayId": "1234567890",
            "subject": "Question about my account"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCases。

describe-communications

以下代码示例演示了如何使用 describe-communications。

Amazon CLI

描述案例的最新通信

以下 describe-communications 示例返回您 Amazon 账户中指定支持案例的最新通信。

```
aws support describe-communications \
    --case-id "case-12345678910-2013-c4c1d2bf33c5cf47" \
    --after-time "2020-03-23T21:31:47.774Z" \
    --max-item 1
```

输出：

```
{
    "communications": [
        {
            "body": "I want to learn more about an AWS service.",
            "attachmentSet": [],
            "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47",
            "timeCreated": "2020-05-12T23:12:35.000Z",
            "submittedBy": "Amazon Web Services"
        }
    ],
    "NextToken": "eyJuZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiEXAMPLE=="
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeCommunications。

describe-services

以下代码示例演示了如何使用 describe-services。

Amazon CLI

列出 Amazon 服务和服务类别

以下 describe-services 示例列出了用于请求一般信息的可用服务类别。

```
aws support describe-services \
    --service-code-list "general-info"
```

输出：

```
{
    "services": [
        {
            "code": "general-info",
            "name": "General Info and Getting Started",
            "categories": [
                {
                    "code": "charges",
                    "name": "How Will I Be Charged?"
                },
                {
                    "code": "gdpr-queries",
                    "name": "Data Privacy Query"
                },
                {
                    "code": "reserved-instances",
                    "name": "Reserved Instances"
                },
                {
                    "code": "resource",
                    "name": "Where is my Resource?"
                },
                {
                    "code": "using-aws",
                    "name": "Using AWS & Services"
                },
                {
                    "code": "free-tier",
                    "name": "Free Tier"
                },
                {
                    "code": "security-and-compliance",
                    "name": "Security & Compliance"
                },
                {
                    "code": "account-structure",
                    "name": "Account Structure"
                }
```

```
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeServices。

describe-severity-levels

以下代码示例演示了如何使用 describe-severity-levels。

Amazon CLI

列出可用的严重性级别

以下 describe-severity-levels 示例列出了支持案例的可用严重性级别。

```
aws support describe-severity-levels
```

输出：

```
{
    "severityLevels": [
        {
            "code": "low",
            "name": "Low"
        },
        {
            "code": "normal",
            "name": "Normal"
        },
        {
            "code": "high",
            "name": "High"
        },
        {
            "code": "urgent",
            "name": "Urgent"
        },
        {
```

```
            "code": "critical",
            "name": "Critical"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的选择严重性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSeverityLevels。

describe-trusted-advisor-check-refresh-statuses

以下代码示例演示了如何使用 describe-trusted-advisor-check-refresh-statuses。

Amazon CLI

列出 Amazon Trusted Advisor 检查的刷新状态

以下 describe-trusted-advisor-check-refresh-statuses 示例列出两个 Trusted Advisor 检查的刷新状态：Amazon S3 存储桶权限和 IAM 使用。

```
aws support describe-trusted-advisor-check-refresh-statuses \
    --check-id "Pfx0RwqBli" "zXCkfM1nI3"
```

输出：

```
{
    "statuses": [
        {
            "checkId": "Pfx0RwqBli",
            "status": "none",
            "millisUntilNextRefreshable": 0
        },
        {
            "checkId": "zXCkfM1nI3",
            "status": "none",
            "millisUntilNextRefreshable": 0
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的 Amazon Trusted Advisor。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrustedAdvisorCheckRefreshStatuses。

describe-trusted-advisor-check-result

以下代码示例演示了如何使用 describe-trusted-advisor-check-result。

Amazon CLI

列出 Amazon Trusted Advisor 检查的结果

以下 describe-trusted-advisor-check-result 示例列出 IAM 使用检查的结果。

```
aws support describe-trusted-advisor-check-result \
    --check-id "zXCkfM1nI3"
```

输出：

```
{
    "result": {
        "checkId": "zXCkfM1nI3",
        "timestamp": "2020-05-13T21:38:05Z",
        "status": "ok",
        "resourcesSummary": {
            "resourcesProcessed": 1,
            "resourcesFlagged": 0,
            "resourcesIgnored": 0,
            "resourcesSuppressed": 0
        },
        "categorySpecificSummary": {
            "costOptimizing": {
                "estimatedMonthlySavings": 0.0,
                "estimatedPercentMonthlySavings": 0.0
            }
        },
        "flaggedResources": [
            {
                "status": "ok",
                "resourceId": "47DEQpj8HBSa-_TImW-5JCeuQeRkm5NMpJWZEXAMPLE",
                "isSuppressed": false
            }
        ]
```

```
    }
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的 Amazon Trusted Advisor。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrustedAdvisorCheckResult。

`describe-trusted-advisor-check-summaries`

以下代码示例演示了如何使用 `describe-trusted-advisor-check-summaries`。

Amazon CLI

列出 Amazon Trusted Advisor 检查的摘要

以下 `describe-trusted-advisor-check-summaries` 示例列出两个 Trusted Advisor 检查的结果：Amazon S3 存储桶权限和 IAM 使用。

```
aws support describe-trusted-advisor-check-summaries \
    --check-ids "Pfx0RwqBli" "zXCkfM1nI3"
```

输出：

```
{
    "summaries": [
        {
            "checkId": "Pfx0RwqBli",
            "timestamp": "2020-05-13T21:38:12Z",
            "status": "ok",
            "hasFlaggedResources": true,
            "resourcesSummary": {
                "resourcesProcessed": 44,
                "resourcesFlagged": 0,
                "resourcesIgnored": 0,
                "resourcesSuppressed": 0
            },
            "categorySpecificSummary": {
                "costOptimizing": {
                    "estimatedMonthlySavings": 0.0,
                    "estimatedPercentMonthlySavings": 0.0
                }
```

```
            }
        },
        {
            "checkId": "zXCkfM1nI3",
            "timestamp": "2020-05-13T21:38:05Z",
            "status": "ok",
            "hasFlaggedResources": true,
            "resourcesSummary": {
                "resourcesProcessed": 1,
                "resourcesFlagged": 0,
                "resourcesIgnored": 0,
                "resourcesSuppressed": 0
            },
            "categorySpecificSummary": {
                "costOptimizing": {
                    "estimatedMonthlySavings": 0.0,
                    "estimatedPercentMonthlySavings": 0.0
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的 Amazon Trusted Advisor。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrustedAdvisorCheckSummaries。

`describe-trusted-advisor-checks`

以下代码示例演示了如何使用 `describe-trusted-advisor-checks`。

Amazon CLI

列出可用的 Amazon Trusted Advisor 检查

以下 `describe-trusted-advisor-checks` 示例列出您的 Amazon 账户中可用的 Trusted Advisor 检查。这些信息包括检查名称、ID、描述、类别和元数据。请注意，为便于阅读，输出已缩短。

```
aws support describe-trusted-advisor-checks \
    --language "en"
```

输出：

```
{
    "checks": [
        {
            "id": "zXCkfM1nI3",
            "name": "IAM Use",
            "description": "Checks for your use of AWS Identity and Access
 Management (IAM). You can use IAM to create users, groups, and roles in AWS, and
 you can use permissions to control access to AWS resources. \n<br>\n<br>\n<b>Alert
 Criteria</b><br>\nYellow: No IAM users have been created for this account.\n<br>
\n<br>\n<b>Recommended Action</b><br>\nCreate one or more IAM users and groups in
 your account. You can then create additional users whose permissions are limited
 to perform specific tasks in your AWS environment. For more information, see <a
 href=\"https://docs.aws.amazon.com/IAM/latest/UserGuide/IAMGettingStarted.html\"
 target=\"_blank\">Getting Started</a>. \n<br><br>\n<b>Additional Resources</b><br>
\n<a href=\"https://docs.aws.amazon.com/IAM/latest/UserGuide/IAM_Introduction.html\"
 target=\"_blank\">What Is IAM?</a>",
            "category": "security",
            "metadata": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的 Amazon Trusted Advisor。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTrustedAdvisorChecks。

`refresh-trusted-advisor-check`

以下代码示例演示了如何使用 `refresh-trusted-advisor-check`。

Amazon CLI

刷新 Amazon Trusted Advisor 检查

以下 `refresh-trusted-advisor-check` 示例刷新您 Amazon 账户中的 Amazon S3 存储桶权限 Trusted Advisor 检查。

```
aws support refresh-trusted-advisor-check \
    --check-id "Pfx0RwqBli"
```

输出：

```
{
    "status": {
        "checkId": "Pfx0RwqBli",
        "status": "enqueued",
        "millisUntilNextRefreshable": 3599992
    }
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的 Amazon Trusted Advisor。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RefreshTrustedAdvisorCheck。

`resolve-case`

以下代码示例演示了如何使用 `resolve-case`。

Amazon CLI

处理支持案例

以下 `resolve-case` 示例演示了如何处理您 Amazon 账户中的支持案例。

```
aws support resolve-case \
    --case-id "case-12345678910-2013-c4c1d2bf33c5cf47"
```

输出：

```
{
    "finalCaseStatus": "resolved",
    "initialCaseStatus": "work-in-progress"
}
```

有关更多信息，请参阅《Amazon Support 用户指南》中的案例管理。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResolveCase。

使用 Amazon CLI 的 Amazon SWF 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon SWF 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`count-closed-workflow-executions`

以下代码示例演示了如何使用 `count-closed-workflow-executions`。

Amazon CLI

统计已关闭的工作流执行次数

您可以使用 `swf count-closed-workflow-executions` 检索给定域的已关闭工作流执行次数。您可以指定筛选器来统计指定的执行类别。

`--domain` 和 `--close-time-filter` 或 `--start-time-filter` 参数为必填项。所有其他参数都为可选项。

```
aws swf count-closed-workflow-executions \
    --domain DataFrobtzz \
    --close-time-filter "{ \"latestDate\" : 1377129600, \"oldestDate\" :
 1370044800 }"
```

输出：

```
{
    "count": 2,
    "truncated": false
}
```

如果“truncated”为 `true`，则“count”表示 Amazon SWF 可以返回的最大数量。任何其他结果都会被截断。

要减少返回结果的数量，您可以：

修改 --close-time-filter 或 --start-time-filter 值以缩小搜索的时间范围。其中每一个参数都是互斥的：您只能在请求中指定其中一个。使用 --close-status-filter、--execution-filter、--tag-filter 或 --type-filter 参数可进一步筛选结果。但是，这些参数也是相互排斥的。

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 CountClosedWorkflowExecutions

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CountClosedWorkflowExecutions。

count-open-workflow-executions

以下代码示例演示了如何使用 count-open-workflow-executions。

Amazon CLI

统计打开的工作流执行次数

您可以使用 swf count-open-workflow-executions 检索给定域的打开的工作流执行次数。您可以指定筛选器来统计指定的执行类别。

--domain 和 --start-time-filter 是必填参数。所有其他参数都为可选项。

```
aws swf count-open-workflow-executions \
    --domain DataFrobtzz \
    --start-time-filter "{ \"latestDate\" : 1377129600, \"oldestDate\" :
 1370044800 }"
```

输出：

```
{
    "count": 4,
    "truncated": false
}
```

如果“truncated”为 true，则“count”表示 Amazon SWF 可以返回的最大数量。任何其他结果都会被截断。

要减少返回结果的数量，您可以：

修改 --start-time-filter 值以缩小搜索的时间范围。使用--close-status-filter、--execution-filter、--tag-filter 或 --type-filter 参数进一步筛选结果。其中每一参数都是互斥的：您只能在请求中指定其中一个。

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的“CountOpenWorkflowExecutions”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CountOpenWorkflowExecutions。

deprecate-domain

以下代码示例演示了如何使用 deprecate-domain。

Amazon CLI

弃用域

要弃用域（您仍可以看到它，但不能在它上面创建新工作流执行或注册类型），请使用 swf deprecate-domain。它只有一个必需参数 --name，此参数用于指定要弃用的域的名称。

```
aws swf deprecate-domain \
    --name MyNeatNewDomain ""
```

与 register-domain 一样，不会返回任何输出。不过，如果您使用 list-domains 查看已注册的域，则会看到该域已弃用，不会再显示在返回的数据中。

```
aws swf list-domains \
    --registration-status REGISTERED
        {
            "domainInfos": [
                {
                    "status": "REGISTERED",
                    "name": "DataFrobotz"
                },
                {
                    "status": "REGISTERED",
                    "name": "erontest"
                }
            ]
        }
```

如果您将 list-domains 与 --registration-status DEPRECATED 一起使用，则会看到已弃用的域。

```
aws swf list-domains \
```

```
    --registration-status DEPRECATED
        {
            "domainInfos": [
                {
                    "status": "DEPRECATED",
                    "name": "MyNeatNewDomain"
                }
            ]
        }
```

您还可以使用 describe-domain 获取有关已弃用域的信息。

```
aws swf describe-domain \
    --name MyNeatNewDomain
        {
            "domainInfo": {
                "status": "DEPRECATED",
                "name": "MyNeatNewDomain"
            },
            "configuration": {
                "workflowExecutionRetentionPeriodInDays": "0"
            }
        }
```

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 DeprecateDomain

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeprecateDomain。

describe-domain

以下代码示例演示了如何使用 describe-domain。

Amazon CLI

获取有关域的信息

要获取有关特定域的详细信息，请使用 swf describe-domain 命令。有一个必需参数：--name，此参数用于指定您要获取其信息的域的名称。

```
aws swf describe-domain \
    --name DataFrobotz
        {
```

```
            "domainInfo": {
                "status": "REGISTERED",
                "name": "DataFrobotz"
            },
            "configuration": {
                "workflowExecutionRetentionPeriodInDays": "1"
            }
        }
```

您还可以使用 describe-domain 获取有关已弃用域的信息。

```
aws swf describe-domain \
    --name MyNeatNewDomain
        {
            "domainInfo": {
                "status": "DEPRECATED",
                "name": "MyNeatNewDomain"
            },
            "configuration": {
                "workflowExecutionRetentionPeriodInDays": "0"
            }
        }
```

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 DescribeDomain

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDomain。

list-activity-types

以下代码示例演示了如何使用 list-activity-types。

Amazon CLI

列出活动类型

要获取域的活动类型列表，请使用 swf list-activity-types。--domain 和 --registration-status 是必填参数。

```
aws swf list-activity-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED
```

输出：

```
{
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.451,
            "activityType": {
                "version": "1",
                "name": "confirm-user-email"
            },
            "description": "subscribe confirm-user-email activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.709,
            "activityType": {
                "version": "1",
                "name": "confirm-user-phone"
            },
            "description": "subscribe confirm-user-phone activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454149.871,
            "activityType": {
                "version": "1",
                "name": "get-subscription-info"
            },
            "description": "subscribe get-subscription-info activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.909,
            "activityType": {
                "version": "1",
                "name": "send-subscription-success"
            },
            "description": "subscribe send-subscription-success activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.085,
            "activityType": {
```

```
                "version": "1",
                "name": "subscribe-user-sns"
            },
            "description": "subscribe subscribe-user-sns activity"
        }
    ]
}
```

您可以使用 --name 参数仅选择具有特定名称的活动类型：

```
aws swf list-activity-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED \
    --name "send-subscription-success"
```

输出：

```
{
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.909,
            "activityType": {
                "version": "1",
                "name": "send-subscription-success"
            },
            "description": "subscribe send-subscription-success activity"
        }
    ]
}
```

要在页面中检索结果，可以设置 --maximum-page-size 参数。如果返回的结果多于结果页所能容纳的数量，则会在结果集中返回“nextPageToken”：

```
aws swf list-activity-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED \
    --maximum-page-size 2
```

输出：

```
{
```

```
    "nextPageToken": "AAAAKgAAAAEAAAAAAAAAA1Gp1BelJq
+PmHvAnDxJYbup8+0R4LVtbXLDl7QNY7C3OpHo9Sz06D/GuFz1OyC73umBQ1tOPJ/gC/
aYpzDMqUIWIA1T9W0s2DryyZX4OC/6Lhk9/
o5kdsuWMSBkHhgaZjgwp3WJINIFJFdaSMxY2vYAX7AtRtpcqJuBDDRE9RaRqDGYqIYUMltarkiqpSY1ZVveBasBvlvyU
WGAaqehiDz7/JzLT/wWNNUMOd+Nhe",
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.451,
            "activityType": {
                "version": "1",
                "name": "confirm-user-email"
            },
            "description": "subscribe confirm-user-email activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.709,
            "activityType": {
                "version": "1",
                "name": "confirm-user-phone"
            },
            "description": "subscribe confirm-user-phone activity"
        }
    ]
}
```

您可以将 nextPageToken 值传递给 `--next-page-token` 参数中的下一个调用 `list-activity-types`，检索下一页的结果：

```
aws swf list-activity-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED \
    --maximum-page-size 2 \
    --next-page-token "AAAAKgAAAAEAAAAAAAAAA1Gp1BelJq
+PmHvAnDxJYbup8+0R4LVtbXLDl7QNY7C3OpHo9Sz06D/GuFz1OyC73umBQ1tOPJ/gC/
aYpzDMqUIWIA1T9W0s2DryyZX4OC/6Lhk9/
o5kdsuWMSBkHhgaZjgwp3WJINIFJFdaSMxY2vYAX7AtRtpcqJuBDDRE9RaRqDGYqIYUMltarkiqpSY1ZVveBasBvlvyU
WGAaqehiDz7/JzLT/wWNNUMOd+Nhe"
```

输出：

```
{
```

```
    "nextPageToken": "AAAAKgAAAAEAAAAAAAAAAw+7LZ4GRZPzTqBHsp2wBxWB8m1sgLCclgCuq3J+h/
m3+vOfFqtkcjLwV5cc4OjNAzTCuq/
XcylPumGwkjbajtqpZpbqOcVNfjFxGoi0LB2Olbvv0krbUISBvlpFPmSWpDSZJsxg5UxCcweteSlFn1PNSZ/
MoinBZo8OTkjMuzcsTuKOzH9wCaR8ITcALJ3SaqHU3pyIRS5hPmFA3OLIc8zaAepjlaujo6hntNSCruB4"
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454149.871,
            "activityType": {
                "version": "1",
                "name": "get-subscription-info"
            },
            "description": "subscribe get-subscription-info activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.909,
            "activityType": {
                "version": "1",
                "name": "send-subscription-success"
            },
            "description": "subscribe send-subscription-success activity"
        }
    ]
}
```

如果还有更多结果要返回，“nextPageToken”将与结果一起返回。当没有更多的结果页要返回时，“nextPageToken”将不会在结果集中返回。

您可以使用 `--reverse-order` 参数来颠倒返回结果的顺序。这也会影响分页结果。

```
aws swf list-activity-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED \
    --maximum-page-size 2 \
    --reverse-order
```

输出：

```
{
    "nextPageToken": "AAAAKgAAAAEAAAAAAAAAwXcpu5ePSyQkrC
+8WMbmSrenuZC2ZkIXQYBPB/b9xIOVkj+bMEFhGj0KmmJ4rF7iddhjf7UMYCsfGkEn7mk
```

```
+yMCgVc1JxDWmB0EH46bhcmcLmYNQihMDmUWocpr7To6/R7CLu0St1gkFayxOidJXErQW0zdNfQaIWAnF/
cwioBbXlkz1fQzmDeU3M5oYGMPQIrUqkPq7pMEW0q0lK5eDN97NzFYdZZ/rlcLDWPZhUjY",
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.085,
            "activityType": {
                "version": "1",
                "name": "subscribe-user-sns"
            },
            "description": "subscribe subscribe-user-sns activity"
        },
        {
            "status": "REGISTERED",
            "creationDate": 1371454150.909,
            "activityType": {
                "version": "1",
                "name": "send-subscription-success"
            },
            "description": "subscribe send-subscription-success activity"
        }
    ]
}
```

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListActivityTypes

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListActivityTypes。

list-domains

以下代码示例演示了如何使用 `list-domains`。

Amazon CLI

示例 1：列出您注册的域

以下 `list-domains` 命令示例列出已为您账户注册的 REGISTERED SWF 域。

```
aws swf list-domains \
    --registration-status REGISTERED
```

输出：

```
{
  "domainInfos": [
    {
      "status": "REGISTERED",
      "name": "DataFrobotz"
    },
    {
      "status": "REGISTERED",
      "name": "erontest"
    }
  ]
}
```

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListDomains。

示例 2：列出已弃用的域

以下 `list-domains` 命令示例列出已为您账户注册的 DEPRECATED SWF 域。已弃用的域是指无法注册新工作流或活动，但仍可以查询的域。

```
aws swf list-domains \
    --registration-status DEPRECATED
```

输出：

```
{
  "domainInfos": [
    {
      "status": "DEPRECATED",
      "name": "MyNeatNewDomain"
    }
  ]
}
```

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListDomains。

示例 3：列出已注册域的第一页

以下 `list-domains` 命令示例列出使用 `--maximum-page-size` 选项已为您账户注册的第一页 REGISTERED SWF 域。

```
aws swf list-domains \
    --registration-status REGISTERED \
    --maximum-page-size 1
```

输出：

```
{
    "domainInfos": [
        {
            "status": "REGISTERED",
            "name": "DataFrobotz"
        }
    ],
"nextPageToken": "AAAAKgAAAAEAAAAAAAAAA2QJKNtidVgd49TTeNwYcpD
+QKT2ynuEbibcQWe2QKrslMGe63gpS0MgZGpcpoKttL4OCXRFn98Xif557it
+wSZUsvUDtImjDLvguyuyyFdIZtvIxIKEOPm3k2r4OjAGaFsGOuVbrKljvla7wdU7FYH3OlkNCP8b7PBj9SBkUyGoiAg
}
```

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListDomains。

示例 4：列出已注册域的指定一页

以下 list-domains 命令示例列出使用 --maximum-page-size 选项已为您账户注册的第一页 REGISTERED SWF 域。

再次调用时，如果在 --next-page-token 参数中提供 nextPageToken 的值，那么您将会得到另外一页结果。

```
aws swf list-domains \
    --registration-status REGISTERED \
    --maximum-page-size 1 \
    --next-page-token "AAAAKgAAAAEAAAAAAAAAA2QJKNtidVgd49TTeNwYcpD
+QKT2ynuEbibcQWe2QKrslMGe63gpS0MgZGpcpoKttL4OCXRFn98Xif557it
+wSZUsvUDtImjDLvguyuyyFdIZtvIxIKEOPm3k2r4OjAGaFsGOuVbrKljvla7wdU7FYH3OlkNCP8b7PBj9SBkUyGoiAg
```

输出：

```
{
    "domainInfos": [
        {
            "status": "REGISTERED",
```

```
            "name": "erontest"
        }
    ]
}
```

当没有要检索的其他结果页时，nextPageToken 不会在结果中返回。

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListDomains。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDomains。

list-workflow-types

以下代码示例演示了如何使用 list-workflow-types。

Amazon CLI

列出工作流类型

要获取域的工作流类型列表，请使用 swf list-workflow-types。--domain 和 --registration-status 是必填参数。下面是一个简单的示例。

```
aws swf list-workflow-types \
    --domain DataFrobtzz \
    --registration-status REGISTERED
```

输出：

```
{
    "typeInfos": [
        {
            "status": "REGISTERED",
            "creationDate": 1371454149.598,
            "description": "DataFrobtzz subscribe workflow",
            "workflowType": {
                "version": "v3",
                "name": "subscribe"
            }
        }
    ]
}
```

与 list-activity-types 一样，您可以使用 --name 参数仅选择具有特定名称的工作流类型，并将 --next-page-token 与 --maximum-page-size 参数结合使用来查看结果。要颠倒返回结果的顺序，请使用 --reverse-order。

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 ListWorkflowTypes

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWorkflowTypes。

register-domain

以下代码示例演示了如何使用 register-domain。

Amazon CLI

注册域

您可以使用 Amazon CLI 注册新域。使用 swf register-domain 命令。有两个必需参数：--name 和 --workflow-execution-retention-period-in-days，前者指定域名，后者使用一个整数指定在该域上保留工作流执行数据的天数，最长 90 天（有关更多信息，请参阅 SWF 常见问题 <https://aws.amazon.com/swf/faqs/#retain_limit>）。在指定的天数过后，系统将不会保留工作流执行数据。

```
aws swf register-domain \
    --name MyNeatNewDomain \
    --workflow-execution-retention-period-in-days 0
    ""
```

注册域时，不会返回任何内容 ("")，但您可以使用 swf list-domains 或 swf describe-domain 查看新域。

```
aws swf list-domains \
    --registration-status REGISTERED
        {
            "domainInfos": [
                {
                    "status": "REGISTERED",
                    "name": "DataFrobotz"
                },
                {
                    "status": "REGISTERED",
                    "name": "MyNeatNewDomain"
                },
```

```
            {
                "status": "REGISTERED",
                "name": "erontest"
            }
        ]
    }
```

使用 `swf describe-domain`：

```
aws swf describe-domain --
name MyNeatNewDomain
{
    "domainInfo": {
        "status": "REGISTERED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"

  }
}
```

另请参阅《Amazon Simple Workflow 服务 API 参考》中的 RegisterDomain

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterDomain。

register-workflow-type

以下代码示例演示了如何使用 `register-workflow-type`。

Amazon CLI

注册工作流类型

要使用 Amazon CLI 注册工作流类型，请使用 `swf register-workflow-type` 命令。

```
aws swf register-workflow-type \
    --domain DataFrobtzz \
    --name "MySimpleWorkflow" \
    --workflow-version "v1"
```

如果成功，此命令不会产生任何输出。

如果出现错误（例如，两次尝试注册相同的工作流类型，或指定一个不存在的域），您将收到一个 JSON 格式的响应。

```
{
    "message": "WorkflowType=[name=MySimpleWorkflow, version=v1]",
    "__type": "com.amazonaws.swf.base.model#TypeAlreadyExistsFault"
}
```

`--domain`、`--name` 和 `--workflow-version` 为必填。您还可以设置工作流描述、超时和子工作流策略。

有关更多信息，请参阅《Amazon Simple Workflow 服务 API 参考》中的 RegisterWorkflowType

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterWorkflowType。

使用 Amazon CLI 的 Systems Manager 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Systems Manager 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`add-tags-to-resource`

以下代码示例演示了如何使用 `add-tags-to-resource`。

Amazon CLI

示例 1：向维护时段添加标签

以下 `add-tags-to-resource` 示例向指定的维护时段添加标签。

```
aws ssm add-tags-to-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "mw-03eb9db428EXAMPLE" \
    --tags "Key=Stack,Value=Production"
```

此命令不生成任何输出。

示例 2：向参数添加标签

以下 add-tags-to-resource 示例向指定参数添加两个标签。

```
aws ssm add-tags-to-resource \
    --resource-type "Parameter" \
    --resource-id "My-Parameter" \
    --tags '[{"Key":"Region","Value":"East"},{"Key":"Environment",
 "Value":"Production"}]'
```

此命令不生成任何输出。

示例 3：向 SSM 文档添加标签

以下 add-tags-to-resource 示例向指定文档添加标签。

```
aws ssm add-tags-to-resource \
    --resource-type "Document" \
    --resource-id "My-Document" \
    --tags "Key=Quarter,Value=Q322"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的标记 Systems Manager 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddTagsToResource。

associate-ops-item-related-item

以下代码示例演示了如何使用 associate-ops-item-related-item。

Amazon CLI

关联相关项目

以下 associate-ops-item-related-item 示例会将相关项目与 OpsItem 关联。

```
aws ssm associate-ops-item-related-item \
    --ops-item-id "oi-649fExample" \
    --association-type "RelatesTo" \
    --resource-type "AWS::SSMIncidents::IncidentRecord" \
    --resource-uri "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/c2bde883-f7d5-343a-b13a-bf5fe9ea689f"
```

输出：

```
{
    "AssociationId": "61d7178d-a30d-4bc5-9b4e-a9e74EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 OpsCenter 中处理 Incident Manager 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateOpsItemRelatedItem。

cancel-command

以下代码示例演示了如何使用 cancel-command。

Amazon CLI

示例 1：取消所有实例的命令

以下 cancel-command 示例尝试取消已对所有实例运行的指定命令。

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

此命令不生成任何输出。

示例 2：取消特定实例的命令

以下 cancel-command 示例仅尝试取消指定实例的命令。

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
    --instance-ids "i-02573cafcfEXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的标记 Systems Manager 参数。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 CancelCommand。

cancel-maintenance-window-execution

以下代码示例演示了如何使用 cancel-maintenance-window-execution。

Amazon CLI

取消执行维护时段

此 cancel-maintenance-window-execution 示例停止执行已在进行的指定维护时段。

```
aws ssm cancel-maintenance-window-execution \
    --window-execution-id j2l8d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE
```

输出：

```
{
    "WindowExecutionId": "j2l8d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Systems Manager 维护时段教程（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CancelMaintenanceWindowExecution。

create-activation

以下代码示例演示了如何使用 create-activation。

Amazon CLI

创建托管式实例激活

以下 create-activation 示例创建托管式实例激活。

```
aws ssm create-activation \
```

```
    --default-instance-name "HybridWebServers" \
    --iam-role "HybridWebServersRole" \
    --registration-limit 5
```

输出：

```
{
    "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
    "ActivationCode": "dRmgnYaFv567vEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的步骤 4：为混合环境创建托管式实例激活。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateActivation。

create-association-batch

以下代码示例演示了如何使用 create-association-batch。

Amazon CLI

创建多个关联

此示例将一个配置文档与多个实例相关联。如果适用，输出将返回成功和失败操作的列表。

命令:

```
aws ssm create-association-batch --entries "Name=AWS-
UpdateSSMAgent,InstanceId=i-1234567890abcdef0" "Name=AWS-
UpdateSSMAgent,InstanceId=i-9876543210abcdef0"
```

输出：

```
{
  "Successful": [
      {
          "Name": "AWS-UpdateSSMAgent",
          "InstanceId": "i-1234567890abcdef0",
          "AssociationVersion": "1",
          "Date": 1550504725.007,
          "LastUpdateAssociationDate": 1550504725.007,
          "Status": {
```

```
            "Date": 1550504725.007,
            "Name": "Associated",
            "Message": "Associated with AWS-UpdateSSMAgent"
        },
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-1234567890abcdef0"
                ]
            }
        ]
    },
    {
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-9876543210abcdef0",
        "AssociationVersion": "1",
        "Date": 1550504725.057,
        "LastUpdateAssociationDate": 1550504725.057,
        "Status": {
            "Date": 1550504725.057,
            "Name": "Associated",
            "Message": "Associated with AWS-UpdateSSMAgent"
        },
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "9c9f7f20-5154-4fed-a83e-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-9876543210abcdef0"
                ]
            }
        ]
```

```
        }
    ],
    "Failed": []
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateAssociationBatch。

create-association

以下代码示例演示了如何使用 create-association。

Amazon CLI

示例 1：使用实例 ID 关联文档

此示例使用实例 ID 将配置文档与实例关联起来。

```
aws ssm create-association \
    --instance-id "i-0cb2b964d3e14fd9f" \
    --name "AWS-UpdateSSMAgent"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
```

```
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager API Reference》中的 CreateAssociation。

示例 2：使用目标关联文档

此示例使用目标将配置文档与实例关联起来。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
```

```
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager API Reference》中的 CreateAssociation。

示例 3：创建仅运行一次的关联

此示例创建一个仅在指定日期和时间运行一次的新关联。使用过去或现在的日期创建的关联（处理关联时该日期已过去）会立即运行。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
    --schedule-expression "at(2020-05-14T15:55:00)"  \
    --apply-only-at-cron-interval
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
```

```
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager API Reference》中的 CreateAssociation 或《Amazon Systems Manager 用户指南》中的参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateAssociation。

`create-document`

以下代码示例演示了如何使用 `create-document`。

Amazon CLI

创建文档

以下 `create-document` 示例创建一个 Systems Manager 文档。

```
aws ssm create-document \
    --content file://exampleDocument.yml \
    --name "Example" \
    --document-type "Automation" \
    --document-format YAML
```

输出：

```
{
    "DocumentDescription": {
        "Hash": "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
        "HashType": "Sha256",
        "Name": "Example",
        "Owner": "29884EXAMPLE",
        "CreatedDate": 1583256349.452,
        "Status": "Creating",
        "DocumentVersion": "1",
        "Description": "Document Example",
        "Parameters": [
            {
```

```
                "Name": "AutomationAssumeRole",
                "Type": "String",
                "Description": "(Required) The ARN of the role that allows
 Automation to perform the actions on your behalf. If no role is specified, Systems
 Manager Automation uses your IAM permissions to execute this document.",
                "DefaultValue": ""
            },
            {
                "Name": "InstanceId",
                "Type": "String",
                "Description": "(Required) The ID of the Amazon EC2 instance.",
                "DefaultValue": ""
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux"
        ],
        "DocumentType": "Automation",
        "SchemaVersion": "0.3",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "YAML",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateDocument。

`create-maintenance-window`

以下代码示例演示了如何使用 `create-maintenance-window`。

Amazon CLI

示例 1：创建维护时段

以下 `create-maintenance-window` 示例创建一个新的维护时段，每五分钟执行一次，最多持续两个小时（根据需要），防止新任务在维护时段执行结束后的一小时内启动，允许未关联的目标（您尚未向维护时段注册的实例），并通过使用自定义标签表明其创建者打算在教程中进行使用。

```
aws ssm create-maintenance-window \
    --name "My-Tutorial-Maintenance-Window" \
    --schedule "rate(5 minutes)" \
    --duration 2 --cutoff 1 \
    --allow-unassociated-targets \
    --tags "Key=Purpose,Value=Tutorial"
```

输出：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE"
}
```

示例 2：创建仅运行一次的维护时段

以下 create-maintenance-window 示例创建了一个仅在指定日期和时间运行一次的新维护时段。

```
aws ssm create-maintenance-window \
    --name My-One-Time-Maintenance-Window \
    --schedule "at(2020-05-14T15:55:00)" \
    --duration 5 \
    --cutoff 2 \
    --allow-unassociated-targets \
    --tags "Key=Environment,Value=Production"
```

输出：

```
{
    "WindowId": "mw-01234567890abcdef"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的维护时段。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateMaintenanceWindow。

create-ops-item

以下代码示例演示了如何使用 create-ops-item。

Amazon CLI

创建 OpsItem

以下 `create-ops-item` 示例在 OperationalData 中使用 /aws/resources 键创建具有 Amazon DynamoDB 相关资源的 OpsItem。

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
    --description "Log clean up may have failed which caused the disk to be full" \
    --priority 2 \
    --source ec2 \
    --operational-data '{"/aws/resources":{"Value":"[{\"arn
\": \"arn:aws:dynamodb:us-west-2:12345678:table/OpsItems
\"}]","Type":"SearchableString"}}' \
    --notifications Arn="arn:aws:sns:us-west-2:12345678:TestUser"
```

输出：

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 OpsItem。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateOpsItem。

create-patch-baseline

以下代码示例演示了如何使用 `create-patch-baseline`。

Amazon CLI

示例 1：创建具有自动批准功能的补丁基准

以下 `create-patch-baseline` 示例创建一个 Windows Server 的补丁基准，该基准在 Microsoft 发布补丁 7 天后批准生产环境的补丁。

```
aws ssm create-patch-baseline \
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
```

```
    --approval-
rules "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Impo
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},Approv
 \
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

示例 2：创建带有批准截止日期的补丁基准

以下 create-patch-baseline 示例为 Windows Server 创建补丁基准，其批准 2020 年 7 月 7 日或之前在生产环境中发布的所有补丁。

```
aws ssm create-patch-baseline \
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
    --approval-
rules "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Impo
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},Approv
 \
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

示例 3：创建批准规则存储在 JSON 文件中的补丁基准

以下 create-patch-baseline 示例为 Amazon Linux 2017.09 创建补丁基准，其将在补丁发布 7 天后批准生产环境的补丁，指定补丁基准的批准规则，并指定补丁的自定义存储库。

```
aws ssm create-patch-baseline \
    --cli-input-json file://my-amazon-linux-approval-rules-and-repo.json
```

的内容my-amazon-linux-approval-rules-and-repo.json：

```
{
    "Name": "Amazon-Linux-2017.09-Production-Baseline",
    "Description": "My approval rules patch baseline for Amazon Linux 2017.09
 instances",
    "OperatingSystem": "AMAZON_LINUX",
    "Tags": [
        {
            "Key": "Environment",
            "Value": "Production"
        }
    ],
    "ApprovalRules": {
        "PatchRules": [
            {
                "ApproveAfterDays": 7,
                "EnableNonSecurity": true,
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "SEVERITY",
                            "Values": [
                                "Important",
                                "Critical"
                            ]
                        },
                        {
                            "Key": "CLASSIFICATION",
                            "Values": [
                                "Security",
                                "Bugfix"
                            ]
                        },
                        {
                            "Key": "PRODUCT",
                            "Values": [
                                "AmazonLinux2017.09"
                            ]
                        }
                    ]
                }
            }
        ]
```

```
    },
    "Sources": [
        {
            "Name": "My-AL2017.09",
            "Products": [
                "AmazonLinux2017.09"
            ],
            "Configuration": "[amzn-main] \nname=amzn-main-Base
\nmirrorlist=http://repo./$awsregion./$awsdomain//$releasever/main/mirror.list //
nmirrorlist_expire=300//nmetadata_expire=300 \npriority=10 \nfailovermethod=priority
 \nfastestmirror_enabled=0 \ngpgcheck=1 \ngpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-
KEY-amazon-ga \nenabled=1 \nretries=3 \ntimeout=5\nreport_instanceid=yes"
        }
    ]
}
```

示例 4：创建指定已批准和已拒绝补丁的补丁基准

以下 `create-patch-baseline` 示例明确指定要批准和拒绝的补丁，作为默认批准规则的例外情况。

```
aws ssm create-patch-baseline \
    --name "Amazon-Linux-2017.09-Alpha-Baseline" \
    --description "My custom approve/reject patch baseline for Amazon Linux 2017.09
 instances" \
    --operating-system "AMAZON_LINUX" \
    --approved-patches "CVE-2018-1234567,example-pkg-EE-2018*.amzn1.noarch" \
    --approved-patches-compliance-level "HIGH" \
    --approved-patches-enable-non-security \
    --tags "Key=Environment,Value=Alpha"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建自定义补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreatePatchBaseline。

create-resource-data-sync

以下代码示例演示了如何使用 `create-resource-data-sync`。

Amazon CLI

创建资源数据同步

此示例创建资源数据同步。如果此命令成功，则无任何输出。

命令:

```
aws ssm create-resource-data-sync --sync-name "ssm-resource-data-sync" --s3-destination "BucketName=ssm-bucket,Prefix=inventory,SyncFormat=JsonSerDe,Region=us-east-1"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResourceDataSync。

delete-activation

以下代码示例演示了如何使用 delete-activation。

Amazon CLI

删除托管式实例激活

以下 delete-activation 示例删除托管式实例激活。

```
aws ssm delete-activation \
    --activation-id "aa673477-d926-42c1-8757-1358cEXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的为混合环境设置 Amazon Systems Manager。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteActivation。

delete-association

以下代码示例演示了如何使用 delete-association。

Amazon CLI

示例 1：使用关联 ID 删除关联

以下 delete-association 示例删除指定关联 ID 的关联。如果此命令成功，则无任何输出。

```
aws ssm delete-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：删除关联

以下 delete-association 示例删除实例和文档之间的关联。如果此命令成功，则无任何输出。

```
aws ssm delete-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteAssociation。

delete-document

以下代码示例演示了如何使用 delete-document。

Amazon CLI

删除文档

以下 delete-document 示例删除一个 Systems Manager 文档。

```
aws ssm delete-document \
    --name "Example"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDocument。

delete-inventory

以下代码示例演示了如何使用 delete-inventory。

Amazon CLI

删除自定义清单类型

此示例删除自定义清单架构。

命令:

```
aws ssm delete-inventory --type-name "Custom:RackInfo" --schema-delete-
option "DeleteSchema"
```

输出：

```
{
  "DeletionId": "d72ac9e8-1f60-4d40-b1c6-bf8c78c68c4d",
  "TypeName": "Custom:RackInfo",
  "DeletionSummary": {
      "TotalCount": 1,
      "RemainingCount": 1,
      "SummaryItems": [
          {
              "Version": "1.0",
              "Count": 1,
              "RemainingCount": 1
          }
      ]
  }
}
```

禁用自定义清单类型

此示例禁用自定义清单架构。

命令:

```
aws ssm delete-inventory --type-name "Custom:RackInfo" --schema-delete-
option "DisableSchema"
```

输出：

```
{
  "DeletionId": "6961492a-8163-44ec-aa1e-923364dd0850",
```

```
  "TypeName": "Custom:RackInformation",
  "DeletionSummary": {
      "TotalCount": 0,
      "RemainingCount": 0,
      "SummaryItems": []
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteInventory。

delete-maintenance-window

以下代码示例演示了如何使用 delete-maintenance-window。

Amazon CLI

删除维护时段

此 delete-maintenance-window 示例删除指定的维护时段。

```
aws ssm delete-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9"
```

输出：

```
{
    "WindowId":"mw-1a2b3c4d5e6f7g8h9"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的删除维护时段（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteMaintenanceWindow。

delete-parameter

以下代码示例演示了如何使用 delete-parameter。

Amazon CLI

删除参数

以下 delete-parameter 示例将删除指定的一个参数。

```
aws ssm delete-parameter \
    --name "MyParameter"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteParameter。

delete-parameters

以下代码示例演示了如何使用 delete-parameters。

Amazon CLI

删除参数列表

以下 delete-parameters 示例删除指定的参数。

```
aws ssm delete-parameters \
    --names "MyFirstParameter" "MySecondParameter" "MyInvalidParameterName"
```

输出：

```
{
    "DeletedParameters": [
        "MyFirstParameter",
        "MySecondParameter"
    ],
    "InvalidParameters": [
        "MyInvalidParameterName"
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteParameters。

delete-patch-baseline

以下代码示例演示了如何使用 delete-patch-baseline。

Amazon CLI

删除补丁基准

以下 delete-patch-baseline 示例将删除指定的补丁基准。

```
aws ssm delete-patch-baseline \
    --baseline-id "pb-045f10b4f382baeda"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f382baeda"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新或删除补丁基准（控制台）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeletePatchBaseline。

delete-resource-data-sync

以下代码示例演示了如何使用 delete-resource-data-sync。

Amazon CLI

删除资源数据同步

此示例删除资源数据同步。如果此命令成功，则无任何输出。

命令:

```
aws ssm delete-resource-data-sync --sync-name "ssm-resource-data-sync"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResourceDataSync。

deregister-managed-instance

以下代码示例演示了如何使用 deregister-managed-instance。

Amazon CLI

取消注册托管式实例

以下 deregister-managed-instance 示例取消注册指定的托管式实例。

```
aws ssm deregister-managed-instance \
    --instance-id 'mi-08ab247cdfEXAMPLE'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在混合和多云环境中取消注册托管式节点。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeregisterManagedInstance。

deregister-patch-baseline-for-patch-group

以下代码示例演示了如何使用 deregister-patch-baseline-for-patch-group。

Amazon CLI

从补丁基准取消注册补丁组

以下 deregister-patch-baseline-for-patch-group 示例从指定的补丁基准中取消注册指定的补丁组。

```
aws ssm deregister-patch-baseline-for-patch-group \
    --patch-group "Production" \
    --baseline-id "pb-0ca44a362fEXAMPLE"
```

输出：

```
{
  "PatchGroup":"Production",
  "BaselineId":"pb-0ca44a362fEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的将补丁组添加到补丁基准。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 DeregisterPatchBaselineForPatchGroup。

deregister-target-from-maintenance-window

以下代码示例演示了如何使用 deregister-target-from-maintenance-window。

Amazon CLI

从维护时段删除目标

以下 deregister-target-from-maintenance-window 示例从指定的维护时段中删除指定的目标。

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-target-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

输出：

```
{
    "WindowId":"mw-ab12cd34ef56gh78",
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新维护时段（Amazon CLI）。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 DeregisterTargetFromMaintenanceWindow。

deregister-task-from-maintenance-window

以下代码示例演示了如何使用 deregister-task-from-maintenance-window。

Amazon CLI

从维护时段删除任务

以下 deregister-task-from-maintenance-window 示例从指定的维护时段中删除指定的任务。

```
aws ssm deregister-task-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
```

```
    --window-task-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c"
```

输出：

```
{
    "WindowTaskId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c",
    "WindowId":"mw-ab12cd34ef56gh78"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Systems Manager 维护时段教程（Amazon CLI）。

- 有关 API 的详细信息，请参阅《Amazon CLI Command Reference》中的 DeregisterTaskFromMaintenanceWindow。

describe-activations

以下代码示例演示了如何使用 describe-activations。

Amazon CLI

描述激活

以下 describe-activations 示例列出有关您 Amazon 账户中激活的详细信息。

```
aws ssm describe-activations
```

输出：

```
{
    "ActivationList": [
        {
            "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
            "Description": "Example1",
            "IamRole": "HybridWebServersRole,
            "RegistrationLimit": 5,
            "RegistrationsCount": 5,
            "ExpirationDate": 1584316800.0,
            "Expired": false,
            "CreatedDate": 1581954699.792
        },
        {
```

```
            "ActivationId": "3ee0322b-f62d-40eb-b672-13ebfEXAMPLE",
            "Description": "Example2",
            "IamRole": "HybridDatabaseServersRole",
            "RegistrationLimit": 5,
            "RegistrationsCount": 5,
            "ExpirationDate": 1580515200.0,
            "Expired": true,
            "CreatedDate": 1578064132.002
        },
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的步骤 4：为混合环境创建托管式实例激活。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeActivations。

describe-association-execution-targets

以下代码示例演示了如何使用 describe-association-execution-targets。

Amazon CLI

获取关联执行的详细信息

以下 describe-association-execution-targets 示例描述指定的关联执行。

```
aws ssm describe-association-execution-targets \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --execution-id "7abb6378-a4a5-4f10-8312-0123456789ab"
```

输出：

```
{
    "AssociationExecutionTargets": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "ResourceId": "i-1234567890abcdef0",
            "ResourceType": "ManagedInstance",
            "Status": "Success",
            "DetailedStatus": "Success",
```

```
            "LastExecutionDate": 1550505538.497,
            "OutputSource": {
                "OutputSourceId": "97fff367-fc5a-4299-aed8-0123456789ab",
                "OutputSourceType": "RunCommand"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看关联历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAssociationExecutionTargets。

describe-association-executions

以下代码示例演示了如何使用 describe-association-executions。

Amazon CLI

示例 1：获取关联所有执行的详细信息

以下 describe-association-executions 示例描述指定关联的所有执行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
    "AssociationExecutions": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505827.119,
            "ResourceCountByStatus": "{Success=1}"
        },
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
```

```
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505536.843,
            "ResourceCountByStatus": "{Success=1}"
        },
        ...
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看关联历史记录。

示例 2：获取特定日期和时间之后关联的所有执行的详细信息

以下 `describe-association-executions` 示例描述指定日期和时间之后关联的所有执行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --filters "Key=CreatedTime,Value=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

输出：

```
{
    "AssociationExecutions": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505827.119,
            "ResourceCountByStatus": "{Success=1}"
        },
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505536.843,
            "ResourceCountByStatus": "{Success=1}"
        },
        ...
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看关联历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAssociationExecutions。

describe-association

以下代码示例演示了如何使用 describe-association。

Amazon CLI

示例 1：获取关联的详细信息

以下 describe-association 示例描述指定关联 ID 的关联。

```
aws ssm describe-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-GatherSoftwareInventory",
        "AssociationVersion": "1",
        "Date": 1534864780.995,
        "LastUpdateAssociationDate": 1543235759.81,
        "Overview": {
            "Status": "Success",
            "AssociationStatusAggregatedCount": {
                "Success": 2
            }
        },
        "DocumentVersion": "$DEFAULT",
        "Parameters": {
            "applications": [
                "Enabled"
            ],
            "awsComponents": [
                "Enabled"
            ],
```

```
            "customInventory": [
                "Enabled"
            ],
            "files": [
                ""
            ],
            "instanceDetailedInformation": [
                "Enabled"
            ],
            "networkConfig": [
                "Enabled"
            ],
            "services": [
                "Enabled"
            ],
            "windowsRegistry": [
                ""
            ],
            "windowsRoles": [
                "Enabled"
            ],
            "windowsUpdates": [
                "Enabled"
            ]
        },
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "*"
                ]
            }
        ],
        "ScheduleExpression": "rate(24 hours)",
        "LastExecutionDate": 1550501886.0,
        "LastSuccessfulExecutionDate": 1550501886.0,
        "AssociationName": "Inventory-Association"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：获取特定实例和文档的关联的详细信息

以下 describe-association 示例描述实例和文档之间的关联。

```
aws ssm describe-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487876122.564,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-1234567890abcdef0",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Associated",
            "AssociationStatusAggregatedCount": {
                "Pending": 1
            }
        },
        "AssociationId": "d8617c07-2079-4c18-9847-1234567890ab",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487876122.564,
        "Date": 1487876122.564,
        "Targets": [
            {
                "Values": [
                    "i-1234567890abcdef0"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编辑和创建关联的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAssociation。

describe-automation-executions

以下代码示例演示了如何使用 describe-automation-executions。

Amazon CLI

描述自动化执行

以下 describe-automation-executions 示例显示了有关自动化执行的详细信息。

```
aws ssm describe-automation-executions \
    --filters Key=ExecutionId,Values=73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：

```
{
    "AutomationExecutionMetadataList": [
        {
            "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",
            "DocumentName": "AWS-StartEC2Instance",
            "DocumentVersion": "1",
            "AutomationExecutionStatus": "Success",
            "ExecutionStartTime": 1583737233.748,
            "ExecutionEndTime": 1583737234.719,
            "ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/mw_service_role/
OrchestrationService",
            "LogFile": "",
            "Outputs": {},
            "Mode": "Auto",
            "Targets": [],
            "ResolvedTargets": {
                "ParameterValues": [],
                "Truncated": false
            },
            "AutomationType": "Local"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的运行简单的自动化工作流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAutomationExecutions。

describe-automation-step-executions

以下代码示例演示了如何使用 describe-automation-step-executions。

Amazon CLI

示例 1：描述自动化执行的所有步骤

以下 describe-automation-step-executions 示例显示有关自动化执行步骤的详细信息。

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：

```
{
    "StepExecutions": [
        {
            "StepName": "startInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1583737234.134,
            "ExecutionEndTime": 1583737234.672,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"running\"",
                "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "running"
                ]
            },
            "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
            "OverriddenParameters": {}
        }
    ]
}
```

示例 2：描述自动化执行的特定步骤

以下 describe-automation-step-executions 示例显示了有关自动化执行中特定步骤的详细信息。

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
    --filters Key=StepExecutionId,Values=95e70479-cf20-4d80-8018-7e4e2EXAMPLE
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的分步运行自动化工作流（命令行）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAutomationStepExecutions。

describe-available-patches

以下代码示例演示了如何使用 describe-available-patches。

Amazon CLI

获取可用补丁

以下 describe-available-patches 示例检索有关 MSRC 严重性为“严重”的所有 Windows Server 2019 可用补丁的详细信息。

```
aws ssm describe-available-patches \
    --
filters "Key=PRODUCT,Values=WindowsServer2019" "Key=MSRC_SEVERITY,Values=Critical"
```

输出：

```
{
    "Patches": [
        {
            "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",
            "ReleaseDate": 1544047205.0,
            "Title": "2018-11 Update for Windows Server 2019 for x64-based Systems
 (KB4470788)",
            "Description": "Install this update to resolve issues in Windows. For a
 complete listing of the issues that are included in this update, see the associated
 Microsoft Knowledge Base article for more information. After you install this item,
 you may have to restart your computer.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2019",
```

```
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
            "KbNumber": "KB4470788",
            "MsrcNumber": "",
            "Language": "All"
        },
        {
            "Id": "c96115e1-5587-4115-b851-22baa46a3f11",
            "ReleaseDate": 1549994410.0,
            "Title": "2019-02 Security Update for Adobe Flash Player for Windows
 Server 2019 for x64-based Systems (KB4487038)",
            "Description": "A security issue has been identified in a Microsoft
 software product that could affect your system. You can help protect your system
 by installing this update from Microsoft. For a complete listing of the issues that
 are included in this update, see the associated Microsoft Knowledge Base article.
 After you install this update, you may have to restart your system.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4487038",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2019",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
            "KbNumber": "KB4487038",
            "MsrcNumber": "",
            "Language": "All"
        },
        ...
    ]
}
```

获取特定补丁的详细信息

以下 describe-available-patches 示例将检索有关指定补丁的详细信息。

```
aws ssm describe-available-patches \
    --filters "Key=PATCH_ID,Values=KB4480979"
```

输出：

```
{
    "Patches": [
        {
            "Id": "680861e3-fb75-432e-818e-d72e5f2be719",
```

```
            "ReleaseDate": 1546970408.0,
            "Title": "2019-01 Security Update for Adobe Flash Player for Windows
 Server 2016 for x64-based Systems (KB4480979)",
            "Description": "A security issue has been identified in a Microsoft
 software product that could affect your system. You can help protect your system
 by installing this update from Microsoft. For a complete listing of the issues that
 are included in this update, see the associated Microsoft Knowledge Base article.
 After you install this update, you may have to restart your system.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4480979",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2016",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
            "KbNumber": "KB4480979",
            "MsrcNumber": "",
            "Language": "All"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Patch Manager 工作原理。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeAvailablePatches。

describe-document-permission

以下代码示例演示了如何使用 describe-document-permission。

Amazon CLI

描述文档权限

以下 describe-document-permission 示例显示有关公开共享 Systems Manager 文档的权限详细信息。

```
aws ssm describe-document-permission \
    --name "Example" \
    --permission-type "Share"
```

输出：

```
{
    "AccountIds": [
        "all"
    ],
    "AccountSharingInfoList": [
        {
            "AccountId": "all",
            "SharedDocumentVersion": "$DEFAULT"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的共享 Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeDocumentPermission。

describe-document

以下代码示例演示了如何使用 describe-document。

Amazon CLI

显示文档的详细信息

以下 describe-document 示例显示有关您 Amazon 账户中 Systems Manager 文档的详细信息。

```
aws ssm describe-document \
    --name "Example"
```

输出：

```
{
    "Document": {
        "Hash": "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
        "HashType": "Sha256",
        "Name": "Example",
        "Owner": "29884EXAMPLE",
        "CreatedDate": 1583257938.266,
        "Status": "Active",
```

```
        "DocumentVersion": "1",
        "Description": "Document Example",
        "Parameters": [
            {
                "Name": "AutomationAssumeRole",
                "Type": "String",
                "Description": "(Required) The ARN of the role that allows
 Automation to perform the actions on your behalf. If no role is specified, Systems
 Manager Automation uses your IAM permissions to execute this document.",
                "DefaultValue": ""
            },
            {
                "Name": "InstanceId",
                "Type": "String",
                "Description": "(Required) The ID of the Amazon EC2 instance.",
                "DefaultValue": ""
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux"
        ],
        "DocumentType": "Automation",
        "SchemaVersion": "0.3",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "YAML",
        "Tags": []
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeDocument。

describe-effective-instance-associations

以下代码示例演示了如何使用 describe-effective-instance-associations。

Amazon CLI

获取实例有效关联的详细信息

以下 describe-effective-instance-associations 示例检索有关实例有效关联的详细信息。

命令:

```
aws ssm describe-effective-instance-associations --instance-id "i-1234567890abcdef0"
```

输出：

```
{
    "Associations": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "InstanceId": "i-1234567890abcdef0",
            "Content": "{\n    \"schemaVersion\": \"1.2\",\n    \"description\":
 \"Update the Amazon SSM Agent to the latest version or specified version.\",\n
   \"parameters\": {\n        \"version\": {\n            \"default\": \"\",\n
         \"description\": \"(Optional) A specific version of the Amazon SSM Agent
 to install. If not specified, the agent will be updated to the latest version.\",
\n            \"type\": \"String\"\n        },\n        \"allowDowngrade\": {\n
            \"default\": \"false\",\n            \"description\": \"(Optional)
 Allow the Amazon SSM Agent service to be downgraded to an earlier version. If
 set to false, the service can be upgraded to newer versions only (default). If
 set to true, specify the earlier version.\",\n            \"type\": \"String\",
\n            \"allowedValues\": [\n                \"true\",\n
 \"false\"\n            ]\n        }\n    },\n    \"runtimeConfig\": {\n
 \"aws:updateSsmAgent\": {\n            \"properties\": [\n                {\n
              \"agentName\": \"amazon-ssm-agent\",\n                \"source\":
 \"https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/ssm-agent-manifest.json\",
\n                \"allowDowngrade\": \"{{ allowDowngrade }}\",\n
 \"targetVersion\": \"{{ version }}\"\n                }\n            ]\n        }\n
    }\n}\n",
            "AssociationVersion": "1"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeEffectiveInstanceAssociations。

describe-effective-patches-for-patch-baseline

以下代码示例演示了如何使用 describe-effective-patches-for-patch-baseline。

Amazon CLI

示例 1：获取自定义补丁基准定义的所有补丁

以下 describe-effective-patches-for-patch-baseline 示例返回当前 Amazon 账户中由自定义补丁基准定义的补丁。请注意，对于自定义基准，--baseline-id 只需要 ID 。

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "pb-08b654cf9b9681f04"
```

输出：

```
{
    "EffectivePatches": [
        {
            "Patch": {
                "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",
                "ReleaseDate": 1544047205.0,
                "Title": "2018-11 Update for Windows Server 2019 for x64-based
 Systems (KB4470788)",
                "Description": "Install this update to resolve issues in Windows.
 For a complete listing of the issues that are included in this update, see the
 associated Microsoft Knowledge Base article for more information. After you install
 this item, you may have to restart your computer.",
                "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",
                "Vendor": "Microsoft",
                "ProductFamily": "Windows",
                "Product": "WindowsServer2019",
                "Classification": "SecurityUpdates",
                "MsrcSeverity": "Critical",
                "KbNumber": "KB4470788",
                "MsrcNumber": "",
                "Language": "All"
            },
            "PatchStatus": {
                "DeploymentStatus": "APPROVED",
                "ComplianceLevel": "CRITICAL",
                "ApprovalDate": 1544047205.0
            }
```

```
        },
        {
            "Patch": {
                "Id": "915a6b1a-f556-4d83-8f50-b2e75a9a7e58",
                "ReleaseDate": 1549994400.0,
                "Title": "2019-02 Cumulative Update for .NET Framework 3.5 and 4.7.2
 for Windows Server 2019 for x64 (KB4483452)",
                "Description": "A security issue has been identified in a Microsoft
 software product that could affect your system. You can help protect your system by
 installing this update from Microsoft. For a complete listing of the issues that
 are included in this update, see the associated Microsoft Knowledge Base article.
 After you install this update, you may have to restart your system.",
                "ContentUrl": "https://support.microsoft.com/en-us/kb/4483452",
                "Vendor": "Microsoft",
                "ProductFamily": "Windows",
                "Product": "WindowsServer2019",
                "Classification": "SecurityUpdates",
                "MsrcSeverity": "Important",
                "KbNumber": "KB4483452",
                "MsrcNumber": "",
                "Language": "All"
            },
            "PatchStatus": {
                "DeploymentStatus": "APPROVED",
                "ComplianceLevel": "CRITICAL",
                "ApprovalDate": 1549994400.0
            }
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

示例 2：获取由 Amazon 托管式补丁基准定义的所有补丁

以下 `describe-effective-patches-for-patch-baseline` 示例返回由 Amazon 托管式补丁基准定义的补丁。请注意，对于 Amazon 托管式基准，`--baseline-id` 需要完整的基准 ARN

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的如何选择安全补丁。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeEffectivePatchesForPatchBaseline。

describe-instance-associations-status

以下代码示例演示了如何使用 describe-instance-associations-status。

Amazon CLI

描述实例关联的状态

此示例显示实例关联的详细信息。

命令:

```
aws ssm describe-instance-associations-status --instance-id "i-1234567890abcdef0"
```

输出：

```
{
  "InstanceAssociationStatusInfos": [
      {
          "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
          "Name": "AWS-GatherSoftwareInventory",
          "DocumentVersion": "1",
          "AssociationVersion": "1",
          "InstanceId": "i-1234567890abcdef0",
          "ExecutionDate": 1550501886.0,
          "Status": "Success",
          "ExecutionSummary": "1 out of 1 plugin processed, 1 success, 0 failed, 0
 timedout, 0 skipped. ",
          "AssociationName": "Inventory-Association"
      },
      {
          "AssociationId": "5c5a31f6-6dae-46f9-944c-0123456789ab",
          "Name": "AWS-UpdateSSMAgent",
          "DocumentVersion": "1",
          "AssociationVersion": "1",
          "InstanceId": "i-1234567890abcdef0",
          "ExecutionDate": 1550505828.548,
          "Status": "Success",
```

```
            "DetailedStatus": "Success",
            "AssociationName": "UpdateSSMAgent"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstanceAssociationsStatus。

describe-instance-information

以下代码示例演示了如何使用 describe-instance-information。

Amazon CLI

示例 1：描述托管式实例信息

以下 describe-instance-information 示例检索每个托管式实例的详细信息。

```
aws ssm describe-instance-information
```

示例 2：描述有关特定托管式实例的信息

以下 describe-instance-information 示例显示托管式实例 i-028ea792daEXAMPLE 的详细信息。

```
aws ssm describe-instance-information \
    --filters "Key=InstanceIds,Values=i-028ea792daEXAMPLE"
```

示例 3：描述有关具有特定标签键的托管式实例的信息

以下 describe-instance-information 示例显示具有标签键 DEV 的托管式实例的详细信息。

```
aws ssm describe-instance-information \
    --filters "Key=tag-key,Values=DEV"
```

输出：

```
{
    "InstanceInformationList": [
        {
```

```
            "InstanceId": "i-028ea792daEXAMPLE",
            "PingStatus": "Online",
            "LastPingDateTime": 1582221233.421,
            "AgentVersion": "2.3.842.0",
            "IsLatestVersion": true,
            "PlatformType": "Linux",
            "PlatformName": "SLES",
            "PlatformVersion": "15.1",
            "ResourceType": "EC2Instance",
            "IPAddress": "192.0.2.0",
            "ComputerName": "ip-198.51.100.0.us-east-2.compute.internal",
            "AssociationStatus": "Success",
            "LastAssociationExecutionDate": 1582220806.0,
            "LastSuccessfulAssociationExecutionDate": 1582220806.0,
            "AssociationOverview": {
                "DetailedStatus": "Success",
                "InstanceAssociationStatusAggregatedCount": {
                    "Success": 2
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的托管式实例。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstanceInformation。

describe-instance-patch-states-for-patch-group

以下代码示例演示了如何使用 describe-instance-patch-states-for-patch-group。

Amazon CLI

示例 1：获取补丁组的实例状态

以下 describe-instance-patch-states-for-patch-group 示例检索有关指定补丁组每个实例的补丁摘要状态的详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --patch-group "Production"
```

输出：


```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 2671,
            "NotApplicableCount": 400,
            "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
            "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        },
        {
            "InstanceId": "i-0471e04240EXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-09ca3fb51fEXAMPLE",
            "SnapshotId": "05d8ffb0-1bbe-4812-ba2d-d9b7bEXAMPLE",
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 2671,
            "NotApplicableCount": 400,
            "OperationStartTime": "2021-08-04T22:06:20.340000-07:00",
            "OperationEndTime": "2021-08-04T22:07:11.220000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
```

```
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        }
    ]
}
```

示例 2：获取缺失五个补丁以上的补丁组的实例状态

以下 describe-instance-patch-states-for-patch-group 示例针对缺失五个补丁以上的实例的指定补丁组，检索补丁摘要状态详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=MissingCount,Type=GreaterThan,Values=5 \
    --patch-group "Production"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "OwnerInformation": "",
            "InstalledCount": 46,
            "InstalledOtherCount": 4,
            "InstalledPendingRebootCount": 1,
            "InstalledRejectedCount": 1,
            "MissingCount": 7,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 232,
            "NotApplicableCount": 654,
            "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
            "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 1
        }
```

```
    ]
}
```

示例 3：获取需要重启的实例少于 10 个的补丁组的实例状态

以下 describe-instance-patch-states-for-patch-group 示例针对需要重启的实例少于 10 个实例的指定补丁组，检索补丁摘要状态的详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=InstalledPendingRebootCount,Type=LessThan,Values=10 \
    --patch-group "Production"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "PatchGroup": "Production",
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 4,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 846,
            "NotApplicableCount": 212,
            "OperationStartTime": "2021-08-046T11:03:50.590000-07:00",
            "OperationEndTime": "2021-08-06T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的了解补丁合规性状态值。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstancePatchStatesForPatchGroup。

describe-instance-patch-states

以下代码示例演示了如何使用 describe-instance-patch-states。

Amazon CLI

获取实例的补丁摘要状态

此 describe-instance-patch-states 示例获取实例的补丁摘要状态。

```
aws ssm describe-instance-patch-states \
    --instance-ids "i-1234567890abcdef0"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PatchGroup": "my-patch-group",
            "BaselineId": "pb-0713accee01234567",
            "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",
            "CriticalNonCompliantCount": 2,
            "SecurityNonCompliantCount": 2,
            "OtherNonCompliantCount": 1,
            "InstalledCount": 123,
            "InstalledOtherCount": 334,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 1,
            "FailedCount": 2,
            "UnreportedNotApplicableCount": 11,
            "NotApplicableCount": 2063,
            "OperationStartTime": "2021-05-03T11:00:56-07:00",
            "OperationEndTime": "2021-05-03T11:01:09-07:00",
            "Operation": "Scan",
            "LastNoRebootInstallOperationTime": "2020-06-14T12:17:41-07:00",
            "RebootOption": "RebootIfNeeded"
        }
```

```
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于补丁合规性。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstancePatchStates。

describe-instance-patches

以下代码示例演示了如何使用 describe-instance-patches。

Amazon CLI

示例 1：获取实例的补丁状态详细信息

以下 describe-instance-patches 示例将检索有关指定实例补丁的详细信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0"
```

输出：

```
{
    "Patches": [
        {
            "Title": "2019-01 Security Update for Adobe Flash Player for Windows
 Server 2016 for x64-based Systems (KB4480979)",
            "KBId": "KB4480979",
            "Classification": "SecurityUpdates",
            "Severity": "Critical",
            "State": "Installed",
            "InstalledTime": "2019-01-09T00:00:00+00:00"
        },
        {
            "Title": "",
            "KBId": "KB4481031",
            "Classification": "",
            "Severity": "",
            "State": "InstalledOther",
            "InstalledTime": "2019-02-08T00:00:00+00:00"
        },
```

```
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

示例 2：获取实例的处于“缺失”状态的补丁列表

以下 describe-instance-patches 示例检索有关指定实例处于“缺失”状态的补丁的信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Missing
```

输出：

```
{
    "Patches": [
        {
            "Title": "Windows Malicious Software Removal Tool x64 - February 2019
 (KB890830)",
            "KBId": "KB890830",
            "Classification": "UpdateRollups",
            "Severity": "Unspecified",
            "State": "Missing",
            "InstalledTime": "1970-01-01T00:00:00+00:00"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于补丁合规性状态。

示例 3：获取实例自指定 InstalledTime 以来所安装补丁的列表

以下 describe-instance-patches 示例通过组合使用 --filters 和 --query，检索指定实例自指定时间以来所安装补丁的信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Installed \
```

```
    --query "Patches[?InstalledTime >= `2023-01-01T16:00:00`]"
```

输出：

```
{
    "Patches": [
        {
            "Title": "2023-03 Cumulative Update for Windows Server 2019 (1809) for
 x64-based Systems (KB5023702)",
            "KBId": "KB5023702",
            "Classification": "SecurityUpdates",
            "Severity": "Critical",
            "State": "Installed",
            "InstalledTime": "2023-03-16T11:00:00+00:00"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeInstancePatches。

describe-inventory-deletions

以下代码示例演示了如何使用 describe-inventory-deletions。

Amazon CLI

获取清单删除信息

此示例检索清单删除操作的详细信息。

命令:

```
aws ssm describe-inventory-deletions
```

输出：

```
{
  "InventoryDeletions": [
```

```
      {
          "DeletionId": "6961492a-8163-44ec-aa1e-01234567850",
          "TypeName": "Custom:RackInformation",
          "DeletionStartTime": 1550254911.0,
          "LastStatus": "InProgress",
          "LastStatusMessage": "The Delete is in progress",
          "DeletionSummary": {
              "TotalCount": 0,
              "RemainingCount": 0,
              "SummaryItems": []
          },
          "LastStatusUpdateTime": 1550254911.0
      },
      {
          "DeletionId": "d72ac9e8-1f60-4d40-b1c6-987654321c4d",
          "TypeName": "Custom:RackInfo",
          "DeletionStartTime": 1550254859.0,
          "LastStatus": "InProgress",
          "LastStatusMessage": "The Delete is in progress",
          "DeletionSummary": {
              "TotalCount": 1,
              "RemainingCount": 1,
              "SummaryItems": [
                  {
                      "Version": "1.0",
                      "Count": 1,
                      "RemainingCount": 1
                  }
              ]
          },
          "LastStatusUpdateTime": 1550254859.0
      }
  ]
}
```

获取特定清单删除的详细信息

此示例检索特定清单删除操作的详细信息。

命令:

```
aws ssm describe-inventory-deletions --deletion-id "d72ac9e8-1f60-4d40-
b1c6-987654321c4d"
```

输出：

```
{
  "InventoryDeletions": [
      {
          "DeletionId": "d72ac9e8-1f60-4d40-b1c6-987654321c4d",
          "TypeName": "Custom:RackInfo",
          "DeletionStartTime": 1550254859.0,
          "LastStatus": "InProgress",
          "LastStatusMessage": "The Delete is in progress",
          "DeletionSummary": {
              "TotalCount": 1,
              "RemainingCount": 1,
              "SummaryItems": [
                  {
                      "Version": "1.0",
                      "Count": 1,
                      "RemainingCount": 1
                  }
              ]
          },
          "LastStatusUpdateTime": 1550254859.0
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInventoryDeletions。

describe-maintenance-window-execution-task-invocations

以下代码示例演示了如何使用 describe-maintenance-window-execution-task-invocations。

Amazon CLI

获取为执行维护时段任务而执行的特定任务调用

以下 describe-maintenance-window-execution-task-invocations 示例列出作为指定维护时段执行组成部分来执行的指定任务的调用。

```
aws ssm describe-maintenance-window-execution-task-invocations \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2a638355" \
```

```
    --task-id "ac0c6ae1-daa3-4a89-832e-d384503b6586"
```

输出：

```
{
    "WindowExecutionTaskInvocationIdentities": [
        {
            "Status": "SUCCESS",
            "Parameters": "{\"documentName\":\"AWS-RunShellScript\",\"instanceIds\":
[\"i-0000293ffd8c57862\"],\"parameters\":{\"commands\":[\"df\"]},\"maxConcurrency\":
\"1\",\"maxErrors\":\"1\"}",
            "InvocationId": "e274b6e1-fe56-4e32-bd2a-8073c6381d8b",
            "StartTime": 1487692834.723,
            "EndTime": 1487692834.871,
            "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2a638355",
            "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d384503b6586"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindowExecutionTaskInvocations。

describe-maintenance-window-execution-tasks

以下代码示例演示了如何使用 describe-maintenance-window-execution-tasks。

Amazon CLI

列出与维护时段执行相关的所有任务

以下 ssm describe-maintenance-window-execution-tasks 示例列出与指定维护时段执行相关的任务。

```
aws ssm describe-maintenance-window-execution-tasks \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

输出：

```
{
    "WindowExecutionTaskIdentities": [
        {
            "Status": "SUCCESS",
            "TaskArn": "AWS-RunShellScript",
            "StartTime": 1487692834.684,
            "TaskType": "RUN_COMMAND",
            "EndTime": 1487692835.005,
            "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
            "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindowExecutionTasks。

describe-maintenance-window-executions

以下代码示例演示了如何使用 describe-maintenance-window-executions。

Amazon CLI

示例 1：列出维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定维护时段的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE"
```

输出：

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
            "Status": "IN_PROGRESS",
```

```
            "StartTime": "2021-08-04T11:00:00.000000-07:00"

        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "ff75b750-4834-4377-8f61-b3cadEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": "2021-08-03T11:00:00.000000-07:00",
            "EndTime": "2021-08-03T11:37:21.450000-07:00"
        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",
            "Status": "FAILED",
            "StatusDetails": "One or more tasks in the orchestration failed.",
            "StartTime": "2021-08-02T11:00:00.000000-07:00",
            "EndTime": "2021-08-02T11:22:36.190000-07:00"
        }
    ]
}
```

示例 2：列出指定日期之前维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定日期之前指定维护时段内的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedBefore,Values=2021-08-03T00:00:00Z"
```

输出：

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",
        "Status": "FAILED",
        "StatusDetails": "One or more tasks in the orchestration failed.",
        "StartTime": "2021-08-02T11:00:00.000000-07:00",
        "EndTime": "2021-08-02T11:22:36.190000-07:00"
    }
    ]
```

```
}
```

示例 3：列出指定日期之后维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定日期之后指定维护时段内的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedAfter,Values=2021-08-04T00:00:00Z"
```

输出：

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
        "Status": "IN_PROGRESS",
        "StartTime": "2021-08-04T11:00:00.000000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindowExecutions。

describe-maintenance-window-schedule

以下代码示例演示了如何使用 describe-maintenance-window-schedule。

Amazon CLI

示例 1：列出维护时段内即将执行的任务

以下 describe-maintenance-window-schedule 示例列出指定维护时段内所有即将执行的任务。

```
aws ssm describe-maintenance-window-schedule \
```

```
    --window-id mw-ab12cd34eEXAMPLE
```

输出：

```
{
    "ScheduledWindowExecutions": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2020-02-19T16:00Z"
        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2020-02-26T16:00Z"
        },
        ...
    ]
}
```

示例 2：列出指定日期之前维护时段内的所有即将执行的任务

以下 describe-maintenance-window-schedule 示例列出指定日期之前指定维护时段内的所有即将执行的任务。

```
aws ssm describe-maintenance-window-schedule \
    --window-id mw-0ecb1226dd7b2e9a6 \
    --filters "Key=ScheduledBefore,Values=2020-02-15T06:00:00Z"
```

示例 3：列出指定日期之后维护时段内的所有即将执行的任务

以下 describe-maintenance-window-schedule 示例列出指定日期之后指定维护时段内的所有即将执行的任务。

```
aws ssm describe-maintenance-window-schedule \
    --window-id mw-0ecb1226dd7b2e9a6 \
    --filters "Key=ScheduledAfter,Values=2020-02-15T06:00:00Z"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMaintenanceWindowSchedule。

describe-maintenance-window-targets

以下代码示例演示了如何使用 describe-maintenance-window-targets。

Amazon CLI

示例 1：列出维护时段内的所有目标

以下 describe-maintenance-window-targets 示例列出维护时段内的所有目标。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-06cf17cbefEXAMPLE"
```

输出：

```
{
    "Targets": [
        {
            "ResourceType": "INSTANCE",
            "OwnerInformation": "Single instance",
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "i-0000293ffdEXAMPLE"
                    ],
                    "Key": "InstanceIds"
                }
            ],
            "WindowTargetId": "350d44e6-28cc-44e2-951f-4b2c9EXAMPLE"
        },
        {
            "ResourceType": "INSTANCE",
            "OwnerInformation": "Two instances in a list",
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "i-0000293ffdEXAMPLE",
```

```
                    "i-0cb2b964d3EXAMPLE"
                ],
                "Key": "InstanceIds"
            }
        ],
        "WindowTargetId": "e078a987-2866-47be-bedd-d9cf4EXAMPLE"
    }
]
}
```

示例 2：列出匹配特定所有者信息值的维护时段的所有目标

此 describe-maintenance-window-targets 示例列出具有特定值的维护时段的所有目标。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0ecb1226ddEXAMPLE" \
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

输出：

```
{
    "Targets": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "WindowTargetId": "da89dcc3-7f9c-481d-ba2b-edcb7d0057f9",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "tag:Environment",
                    "Values": [
                        "Prod"
                    ]
                }
            ],
            "OwnerInformation": "CostCenter1",
            "Name": "ProdTarget1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindowTargets。

describe-maintenance-window-tasks

以下代码示例演示了如何使用 describe-maintenance-window-tasks。

Amazon CLI

示例 1：列出维护时段内的所有任务

以下 describe-maintenance-window-tasks 示例列出指定维护时段内的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-06cf17cbefEXAMPLE"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "WindowTaskId": "018b31c3-2d77-4b9e-bd48-c91edEXAMPLE",
            "TaskArn": "AWS-RestartEC2Instance",
            "TaskParameters": {},
            "Type": "AUTOMATION",
            "Description": "Restarting EC2 Instance for maintenance",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "My-Automation-Example-Task",
            "Priority": 0,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ]
        },
```

```
        {
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "WindowTaskId": "1943dee0-0a17-4978-9bf4-3cc2fEXAMPLE",
            "TaskArn": "AWS-DisableS3BucketPublicReadWrite",
            "TaskParameters": {},
            "Type": "AUTOMATION",
            "Description": "Automation task to disable read/write access on public
 S3 buckets",
            "MaxConcurrency": "10",
            "MaxErrors": "5",
            "Name": "My-Disable-S3-Public-Read-Write-Access-Automation-Task",
            "Priority": 0,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ]
        }
    ]
}
```

示例 2：列出调用 Amazon-RunPowerShellScript 命令文档的维护时段内的所有任务

以下 `describe-maintenance-window-tasks` 示例列出在调用 `AWS-RunPowerShellScript` 命令文档的指定维护时段内的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
```

```
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyTask"
        }
    ]
}
```

示例 3：列出优先级为 3 的维护时段内的所有任务

以下 describe-maintenance-window-tasks 示例列出指定维护时段内 Priority 为 3 的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=3"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
```

```
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 3,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyRunCommandTask"
        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "ee45feff-ad65-4a6c-b478-5cab8EXAMPLE",
            "TaskArn": "AWS-RestartEC2Instance",
            "Type": "AUTOMATION",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 3,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "10",
            "MaxErrors": "5",
            "Name": "My-Automation-Task",
            "Description": "A description for my Automation task"
        }
    ]
}
```

示例 4：列出优先级为 1 并使用 Run Command 的维护时段内的所有任务

此 `describe-maintenance-window-tasks` 示例列出指定维护时段内 `Priority` 为 1 并使用 Run Command 的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
```

```
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyRunCommandTask"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindowTasks。

describe-maintenance-windows-for-target

以下代码示例演示了如何使用 describe-maintenance-windows-for-target。

Amazon CLI

列出与特定实例关联的所有维护时段

以下 describe-maintenance-windows-for-target 示例列出与指定实例关联的目标或任务的维护时段。

```
aws ssm describe-maintenance-windows-for-target \
    --targets Key=InstanceIds,Values=i-1234567890EXAMPLE \
    --resource-type INSTANCE
```

输出：

```
{
    "WindowIdentities": [
        {
            "WindowId": "mw-0c5ed765acEXAMPLE",
            "Name": "My-First-Maintenance-Window"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeMaintenanceWindowsForTarget。

describe-maintenance-windows

以下代码示例演示了如何使用 describe-maintenance-windows。

Amazon CLI

示例 1：列出所有维护时段

以下 describe-maintenance-windows 示例列出当前区域中您 Amazon 账户的所有维护时段。

```
aws ssm describe-maintenance-windows
```

输出：

```
{
```

```
    "WindowIdentities": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "Name": "MyMaintenanceWindow-1",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 1,
            "Schedule": "rate(180 minutes)",
            "NextExecutionTime": "2020-02-12T23:19:20.596Z"
        },
        {
            "WindowId": "mw-03eb9db428EXAMPLE",
            "Name": "MyMaintenanceWindow-2",
            "Enabled": true,
            "Duration": 3,
            "Cutoff": 1,
            "Schedule": "rate(7 days)",
            "NextExecutionTime": "2020-02-17T23:22:00.956Z"
        },
    ]
}
```

示例 2：列出所有已启用的维护时段

以下 `describe-maintenance-windows` 示例列出所有已启用的维护时段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Enabled,Values=true"
```

示例 3：列出与特定名称匹配的维护时段

此 `describe-maintenance-windows` 示例列出具有指定名称的所有维护时段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Name,Values=MyMaintenanceWindow"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribeMaintenanceWindows。

describe-ops-items

以下代码示例演示了如何使用 describe-ops-items。

Amazon CLI

列出一组 OpsItem

以下 describe-ops-items 示例显示您 Amazon 账户中所有打开的 OpsItem 列表。

```
aws ssm describe-ops-items \
    --ops-item-filters "Key=Status,Values=Open,Operator=Equal"
```

输出：

```
{
    "OpsItemSummaries": [
        {
            "CreatedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
            "CreatedTime": "2020-03-14T17:02:46.375000-07:00",
            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "LastModifiedTime": "2020-03-14T17:02:46.375000-07:00",
            "Source": "SSM",
            "Status": "Open",
            "OpsItemId": "oi-7cfc5EXAMPLE",
            "Title": "SSM Maintenance Window execution failed",
            "OperationalData": {
                "/aws/dedup": {
                    "Value": "{\"dedupString\":\"SSMOpsItems-SSM-maintenance-window-
execution-failed\"}",
                    "Type": "SearchableString"
                },
                "/aws/resources": {
                    "Value": "[{\"arn\":\"arn:aws:ssm:us-
east-2:111222333444:maintenancewindow/mw-034093d322EXAMPLE\"}]",
                    "Type": "SearchableString"
                }
            },
            "Category": "Availability",
            "Severity": "3"
        },
        {
```

```
            "CreatedBy": "arn:aws:sts::1112223233444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
            "CreatedTime": "2020-02-26T11:43:15.426000-08:00",
            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "LastModifiedTime": "2020-02-26T11:43:15.426000-08:00",
            "Source": "EC2",
            "Status": "Open",
            "OpsItemId": "oi-6f966EXAMPLE",
            "Title": "EC2 instance stopped",
            "OperationalData": {
                "/aws/automations": {
                    "Value": "[ { \"automationType\": \"AWS:SSM:Automation\",
 \"automationId\": \"AWS-RestartEC2Instance\" } ]",
                    "Type": "SearchableString"
                },
                "/aws/dedup": {
                    "Value": "{\"dedupString\":\"SSMOpsItems-EC2-instance-stopped
\"}",
                    "Type": "SearchableString"
                },
                "/aws/resources": {
                    "Value": "[{\"arn\":\"arn:aws:ec2:us-
east-2:111222333444:instance/i-0beccfbc02EXAMPLE\"}]",
                    "Type": "SearchableString"
                }
            },
            "Category": "Availability",
            "Severity": "3"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOpsItems。

describe-parameters

以下代码示例演示了如何使用 describe-parameters。

Amazon CLI

示例 1：列出所有参数

以下 describe-parameters 示例列出了当前 Amazon 账户和区域中的所有参数。

```
aws ssm describe-parameters
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MySecureStringParameter",
            "Type": "SecureString",
            "KeyId": "alias/aws/ssm",
            "LastModifiedDate": 1582155479.205,
            "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/Admin/
Richard-Roe-Managed",
            "Description": "This is a SecureString parameter",
            "Version": 2,
            "Tier": "Advanced",
            "Policies": [
                {
                    "PolicyText": "{\"Type\":\"Expiration\",\"Version\":\"1.0\",
\"Attributes\":{\"Timestamp\":\"2020-07-07T22:30:00Z\"}}",
                    "PolicyType": "Expiration",
                    "PolicyStatus": "Pending"
                },
                {
                    "PolicyText": "{\"Type\":\"ExpirationNotification\",\"Version\":
\"1.0\",\"Attributes\":{\"Before\":\"12\",\"Unit\":\"Hours\"}}",
                    "PolicyType": "ExpirationNotification",
                    "PolicyStatus": "Pending"
                }
            ]
        },
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "LastModifiedDate": 1582154764.222,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is a StringList parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
```

```
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582154711.976,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Alejandro-Rosalez",
            "Description": "This is a String parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "latestAmi",
            "Type": "String",
            "LastModifiedDate": 1580862415.521,
            "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/lambda-ssm-
role/Automation-UpdateSSM-Param",
            "Version": 3,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

示例 2：列出与特定元数据匹配的所有参数

以下 `describe-parameters` 示例列出了与筛选器匹配的所有参数。

aws ssm describe-parameters --filters "Key=Type,Values=StringList"

输出：

```
{
    "Parameters": [
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "LastModifiedDate": 1582154764.222,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is a StringList parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的搜索 Systems Manager 参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeParameters。

describe-patch-baselines

以下代码示例演示了如何使用 describe-patch-baselines。

Amazon CLI

示例 1：列出所有补丁基准

以下 describe-patch-baselines 示例检索您账户中当前区域所有补丁基准的详细信息。

```
aws ssm describe-patch-baselines
```

输出：

```
{
    "BaselineIdentities": [
        {
            "BaselineName": "AWS-SuseDefaultPatchBaseline",
            "DefaultBaseline": true,
            "BaselineDescription": "Default Patch Baseline for Suse Provided by
 AWS.",
            "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0123fdb36e334a3b2",
            "OperatingSystem": "SUSE"
        },
        {
            "BaselineName": "AWS-DefaultPatchBaseline",
            "DefaultBaseline": false,
            "BaselineDescription": "Default Patch Baseline Provided by AWS.",
            "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed",
            "OperatingSystem": "WINDOWS"
        },
        ...
        {
            "BaselineName": "MyWindowsPatchBaseline",
```

```
            "DefaultBaseline": true,
            "BaselineDescription": "My patch baseline for EC2 instances for Windows
 Server",
            "BaselineId": "pb-0ad00e0dd7EXAMPLE",
            "OperatingSystem": "WINDOWS"
        }
    ]
}
```

示例 2：列出 Amazon 提供的所有补丁基准

以下 `describe-patch-baselines` 示例列出 Amazon 提供的所有补丁基准。

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[AWS]"
```

示例 3：列出您拥有的所有补丁基准

以下 `describe-patch-baselines` 示例列出当前区域在您的账户中创建的所有自定义补丁基准。

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[Self]"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于预定义和自定义补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribePatchBaselines。

describe-patch-group-state

以下代码示例演示了如何使用 `describe-patch-group-state`。

Amazon CLI

获取补丁组的状态

以下 `describe-patch-group-state` 示例检索补丁组的高级补丁合规性摘要。

```
aws ssm describe-patch-group-state \
    --patch-group "Production"
```

输出：

```
{
    "Instances": 21,
    "InstancesWithCriticalNonCompliantPatches": 1,
    "InstancesWithFailedPatches": 2,
    "InstancesWithInstalledOtherPatches": 3,
    "InstancesWithInstalledPatches": 21,
    "InstancesWithInstalledPendingRebootPatches": 2,
    "InstancesWithInstalledRejectedPatches": 1,
    "InstancesWithMissingPatches": 3,
    "InstancesWithNotApplicablePatches": 4,
    "InstancesWithOtherNonCompliantPatches": 1,
    "InstancesWithSecurityNonCompliantPatches": 1,
    "InstancesWithUnreportedNotApplicablePatches": 2
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html>__ 和了解补丁合规性状态值。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribePatchGroupState。

describe-patch-groups

以下代码示例演示了如何使用 describe-patch-groups。

Amazon CLI

显示补丁组注册

以下 describe-patch-groups 示例列出补丁组注册。

```
aws ssm describe-patch-groups
```

输出：

```
{
    "Mappings": [
        {
            "PatchGroup": "Production",
```

```
            "BaselineIdentity": {
                "BaselineId": "pb-0123456789abcdef0",
                "BaselineName": "ProdPatching",
                "OperatingSystem": "WINDOWS",
                "BaselineDescription": "Patches for Production",
                "DefaultBaseline": false
            }
        },
        {
            "PatchGroup": "Development",
            "BaselineIdentity": {
                "BaselineId": "pb-0713accee01234567",
                "BaselineName": "DevPatching",
                "OperatingSystem": "WINDOWS",
                "BaselineDescription": "Patches for Development",
                "DefaultBaseline": true
            }
        },
        ...
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DescribePatchGroups。

`describe-patch-properties`

以下代码示例演示了如何使用 `describe-patch-properties`。

Amazon CLI

列出 Amazon Linux 补丁的可用性

以下 `describe-patch-properties` 示例显示您 Amazon 账户中已提供补丁的 Amazon Linux 产品列表。

```
aws ssm describe-patch-properties \
    --operating-system AMAZON_LINUX \
    --property PRODUCT
```

输出：

```
{
    "Properties": [
        {
            "Name": "AmazonLinux2012.03"
        },
        {
            "Name": "AmazonLinux2012.09"
        },
        {
            "Name": "AmazonLinux2013.03"
        },
        {
            "Name": "AmazonLinux2013.09"
        },
        {
            "Name": "AmazonLinux2014.03"
        },
        {
            "Name": "AmazonLinux2014.09"
        },
        {
            "Name": "AmazonLinux2015.03"
        },
        {
            "Name": "AmazonLinux2015.09"
        },
        {
            "Name": "AmazonLinux2016.03"
        },
        {
            "Name": "AmazonLinux2016.09"
        },
        {
            "Name": "AmazonLinux2017.03"
        },
        {
            "Name": "AmazonLinux2017.09"
        },
        {
            "Name": "AmazonLinux2018.03"
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribePatchProperties。

describe-sessions

以下代码示例演示了如何使用 describe-sessions。

Amazon CLI

示例 1：列出所有活动 Session Manager 会话

此 describe-sessions 示例检索过去 30 天内由指定用户在最近创建的活动会话（包括已连接和已断开连接的会话）的列表。此命令仅返回与使用 Session Manager 启动的目标所进行的连接的结果。它不会列出通过 Remote Desktop Connections 或 SSH 等其他方式建立的连接。

```
aws ssm describe-sessions \
    --state "Active" \
    --filters "key=Owner,value=arn:aws:sts::123456789012:assumed-role/Administrator/
Shirley-Rodriguez"
```

输出：

```
{
    "Sessions": [
        {
            "SessionId": "John-07a16060613c408b5",
            "Target": "i-1234567890abcdef0",
            "Status": "Connected",
            "StartDate": 1550676938.352,
            "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Shirley-
Rodriguez",
            "OutputUrl": {}
        },
        {
            "SessionId": "John-01edf534b8b56e8eb",
            "Target": "i-9876543210abcdef0",
            "Status": "Connected",
            "StartDate": 1550676842.194,
```

```
            "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Shirley-
Rodriguez",
            "OutputUrl": {}
        }
    ]
}
```

示例 2：列出所有已终止的 Session Manager 会话

此 `describe-sessions` 示例检索过去 30 天内所有用户最近终止的会话列表。

```
aws ssm describe-sessions \
    --state "History"
```

输出：

```
{
    "Sessions": [
        {
            "SessionId": "Mary-Major-0022b1eb2b0d9e3bd",
            "Target": "i-1234567890abcdef0",
            "Status": "Terminated",
            "StartDate": 1550520701.256,
            "EndDate": 1550521931.563,
            "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Mary-
Major"
        },
        {
            "SessionId": "Jane-Roe-0db53f487931ed9d4",
            "Target": "i-9876543210abcdef0",
            "Status": "Terminated",
            "StartDate": 1550161369.149,
            "EndDate": 1550162580.329,
            "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Jane-Roe"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看会话历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSessions。

disassociate-ops-item-related-item

以下代码示例演示了如何使用 disassociate-ops-item-related-item。

Amazon CLI

删除相关项目关联

以下 disassociate-ops-item-related-item 示例删除 OpsItem 与相关项目之间的关联。

```
aws ssm disassociate-ops-item-related-item \
    --ops-item-id "oi-f99f2EXAMPLE" \
    --association-id "e2036148-cccb-490e-ac2a-390e5EXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 OpsCenter 中处理 Incident Manager 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateOpsItemRelatedItem。

get-automation-execution

以下代码示例演示了如何使用 get-automation-execution。

Amazon CLI

显示有关自动化执行的详细信息

以下 get-automation-execution 示例显示有关自动化执行的详细信息。

```
aws ssm get-automation-execution \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：

```
{
    "AutomationExecution": {
        "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",
        "DocumentName": "AWS-StartEC2Instance",
        "DocumentVersion": "1",
        "ExecutionStartTime": 1583737233.748,
        "ExecutionEndTime": 1583737234.719,
```

```
        "AutomationExecutionStatus": "Success",
        "StepExecutions": [
            {
                "StepName": "startInstances",
                "Action": "aws:changeInstanceState",
                "ExecutionStartTime": 1583737234.134,
                "ExecutionEndTime": 1583737234.672,
                "StepStatus": "Success",
                "Inputs": {
                    "DesiredState": "\"running\"",
                    "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
                },
                "Outputs": {
                    "InstanceStates": [
                        "running"
                    ]
                },
                "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
                "OverriddenParameters": {}
            }
        ],
        "StepExecutionsTruncated": false,
        "Parameters": {
            "AutomationAssumeRole": [
                ""
            ],
            "InstanceId": [
                "i-0cb99161f6EXAMPLE"
            ]
        },
        "Outputs": {},
        "Mode": "Auto",
        "ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/mw_service_role/
OrchestrationService",
        "Targets": [],
        "ResolvedTargets": {
            "ParameterValues": [],
            "Truncated": false
        }
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的演练：修补 Linux AMI（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetAutomationExecution。

get-calendar-state

以下代码示例演示了如何使用 get-calendar-state。

Amazon CLI

示例 1：获取更改日历的当前状态

此 get-calendar-state 示例返回日历在当前时间的状态。由于该示例没有指定时间，因此会报告日历的当前状态。

```
aws ssm get-calendar-state \
    --calendar-names "MyCalendar"
```

输出：

```
{
    "State": "OPEN",
    "AtTime": "2020-02-19T22:28:51Z",
    "NextTransitionTime": "2020-02-24T21:15:19Z"
}
```

示例 2：获取指定时间内变更日历的状态

此 get-calendar-state 示例返回日历在指定时间的状态。

```
aws ssm get-calendar-state \
    --calendar-names "MyCalendar" \
    --at-time "2020-07-19T21:15:19Z"
```

输出：

```
{
    "State": "CLOSED",
    "AtTime": "2020-07-19T21:15:19Z"
```

```
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的获取变更日历的状态。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetCalendarState。

get-command-invocation

以下代码示例演示了如何使用 get-command-invocation。

Amazon CLI

显示命令调用的详细信息

以下 get-command-invocation 示例列出对指定实例上指定命令的所有调用。

```
aws ssm get-command-invocation \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --instance-id "i-1234567890abcdef0"
```

输出：

```
{
    "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
    "InstanceId": "i-1234567890abcdef0",
    "Comment": "b48291dd-ba76-43e0-b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-
d6ce8EXAMPLE",
    "DocumentName": "AWS-UpdateSSMAgent",
    "DocumentVersion": "",
    "PluginName": "aws:updateSsmAgent",
    "ResponseCode": 0,
    "ExecutionStartDateTime": "2020-02-19T18:18:03.419Z",
    "ExecutionElapsedTime": "PT0.091S",
    "ExecutionEndDateTime": "2020-02-19T18:18:03.419Z",
    "Status": "Success",
    "StatusDetails": "Success",
    "StandardOutputContent": "Updating amazon-ssm-agent from 2.3.842.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been installed,
 update skipped\n",
    "StandardOutputUrl": "",
    "StandardErrorContent": "",
    "StandardErrorUrl": "",
```

```
    "CloudWatchOutputConfig": {
        "CloudWatchLogGroupName": "",
        "CloudWatchOutputEnabled": false
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的了解命令状态。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetCommandInvocation。

get-connection-status

以下代码示例演示了如何使用 get-connection-status。

Amazon CLI

显示托管式实例的连接状态

此 get-connection-status 示例返回指定托管式实例的连接状态。

```
aws ssm get-connection-status \
    --target i-1234567890abcdef0
```

输出：

```
{
    "Target": "i-1234567890abcdef0",
    "Status": "connected"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetConnectionStatus。

get-default-patch-baseline

以下代码示例演示了如何使用 get-default-patch-baseline。

Amazon CLI

示例 1：显示默认 Windows 补丁基准

以下 get-default-patch-baseline 示例检索 Windows Server 默认补丁基准的详细信息。

```
aws ssm get-default-patch-baseline
```

输出：

```
{
  "BaselineId": "pb-0713accee01612345",
  "OperatingSystem": "WINDOWS"
}
```

示例 2：显示 Amazon Linux 的默认补丁基准

以下 get-default-patch-baseline 示例检索 Amazon Linux 默认补丁基准的详细信息。

```
aws ssm get-default-patch-baseline \
    --operating-system AMAZON_LINUX
```

输出：

```
{
    "BaselineId": "pb-047c6eb9c8fc12345",
    "OperatingSystem": "AMAZON_LINUX"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于预定义和自定义补丁基准 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html>__ 和将现有补丁基准设置为默认项。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDefaultPatchBaseline。

get-deployable-patch-snapshot-for-instance

以下代码示例演示了如何使用 get-deployable-patch-snapshot-for-instance。

Amazon CLI

检索实例使用的补丁基准的当前快照

以下 get-deployable-patch-snapshot-for-instance 示例检索实例使用的指定补丁基准当前快照的详细信息。此命令必须使用实例凭证从实例运行。为确保其使用实例凭证，请运行 aws configure 并仅指定您的实例的区域。将 Access Key 和 Secret Key 字段留空。

提示：使用 uuidgen 生成 snapshot-id。

```
aws ssm get-deployable-patch-snapshot-for-instance \
    --instance-id "i-1234567890abcdef0" \
    --snapshot-id "521c3536-930c-4aa9-950e-01234567abcd"
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",
    "Product": "AmazonLinux2018.03",
    "SnapshotDownloadUrl": "https://patch-baseline-snapshot-us-
east-1.s3.amazonaws.com/
ed85194ef27214f5984f28b4d664d14f7313568fea7d4b6ac6c10ad1f729d7e7-773304212436/
AMAZON_LINUX-521c3536-930c-4aa9-950e-01234567abcd?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Date=20190215T164031Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-
Amz-Credential=AKIAJ5C56P35AEBRX2QQ%2F20190215%2Fus-east-1%2Fs3%2Faws4_request&X-
Amz-Signature=efaaaf6e3878e77f48a6697e015efdbda9c426b09c5822055075c062f6ad2149"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的参数名称：快照 ID。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetDeployablePatchSnapshotForInstance。

get-document

以下代码示例演示了如何使用 get-document。

Amazon CLI

获取文档内容

以下 get-document 示例显示 Systems Manager 文档的内容。

```
aws ssm get-document \
    --name "AWS-RunShellScript"
```

输出：

```
{
    "Name": "AWS-RunShellScript",
    "DocumentVersion": "1",
    "Status": "Active",
    "Content": "{\n    \"schemaVersion\":\"1.2\",\n    \"description\":\"Run a
 shell script or specify the commands to run.\",\n    \"parameters\":{\n
 \"commands\":{\n            \"type\":\"StringList\",\n            \"description
\":\"(Required) Specify a shell script or a command to run.\",\n
 \"minItems\":1,\n            \"displayType\":\"textarea\"\n        },\n
 \"workingDirectory\":{\n            \"type\":\"String\",\n            \"default
\":\"\",\n            \"description\":\"(Optional) The path to the working
 directory on your instance.\",\n            \"maxChars\":4096\n        },\n
   \"executionTimeout\":{\n            \"type\":\"String\",\n            \"default
\":\"3600\",\n            \"description\":\"(Optional) The time in seconds for a
 command to complete before it is considered to have failed. Default is 3600 (1
 hour). Maximum is 172800 (48 hours).\",\n            \"allowedPattern\":\"([1-9]
[0-9]{0,4})|(1[0-6][0-9]{4})|(17[0-1][0-9]{3})|(172[0-7][0-9]{2})|(172800)\"\n
        }\n    },\n    \"runtimeConfig\":{\n        \"aws:runShellScript\":{\n
            \"properties\":[\n                {\n                    \"id\":
\"0.aws:runShellScript\",\n                    \"runCommand\":\"{{ commands }}\",\n
                    \"workingDirectory\":\"{{ workingDirectory }}\",\n
       \"timeoutSeconds\":\"{{ executionTimeout }}\"\n                }\n
  ]\n        }\n    }\n}\n",
    "DocumentType": "Command",
    "DocumentFormat": "JSON"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Amazon Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocument。

`get-inventory-schema`

以下代码示例演示了如何使用 `get-inventory-schema`。

Amazon CLI

查看您的清单架构

此示例返回账户清单类型名称列表。

命令:

```
aws ssm get-inventory-schema
```

输出：

```
{
  "Schemas": [
      {
          "TypeName": "AWS:AWSComponent",
          "Version": "1.0",
          "Attributes": [
              {
                  "Name": "Name",
                  "DataType": "STRING"
              },
              {
                  "Name": "ApplicationType",
                  "DataType": "STRING"
              },
              {
                  "Name": "Publisher",
                  "DataType": "STRING"
              },
              {
                  "Name": "Version",
                  "DataType": "STRING"
              },
              {
                  "Name": "InstalledTime",
                  "DataType": "STRING"
              },
              {
                  "Name": "Architecture",
                  "DataType": "STRING"
              },
              {
                  "Name": "URL",
                  "DataType": "STRING"
              }
          ]
      },
      ...
  ],
  "NextToken": "--token string truncated--"
```

```
}
```

查看特定清单类型的清单架构

此示例返回 Amazon:AmazonComponent 清单类型的清单架构。

命令:

```
aws ssm get-inventory-schema --type-name "AWS:AWSComponent"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetInventorySchema。

get-inventory

以下代码示例演示了如何使用 get-inventory。

Amazon CLI

查看您的清单

此示例获取清单的自定义元数据。

命令:

```
aws ssm get-inventory
```

输出：

```
{
  "Entities": [
      {
          "Data": {
              "AWS:InstanceInformation": {
                  "Content": [
                      {
                          "ComputerName": "ip-172-31-44-222.us-
west-2.compute.internal",
                          "InstanceId": "i-0cb2b964d3e14fd9f",
                          "IpAddress": "172.31.44.222",
                          "AgentType": "amazon-ssm-agent",
                          "ResourceType": "EC2Instance",
                          "AgentVersion": "2.0.672.0",
                          "PlatformVersion": "2016.09",
```

```
                        "PlatformName": "Amazon Linux AMI",
                        "PlatformType": "Linux"
                    }
                ],
                "TypeName": "AWS:InstanceInformation",
                "SchemaVersion": "1.0",
                "CaptureTime": "2017-02-20T18:03:58Z"
            }
        },
        "Id": "i-0cb2b964d3e14fd9f"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetInventory。

get-maintenance-window-execution-task-invocation

以下代码示例演示了如何使用 get-maintenance-window-execution-task-invocation。

Amazon CLI

查看有关维护时段任务调用的信息

以下 get-maintenance-window-execution-task-invocation 示例列出有关作为指定维护时段执行组成部分的指定任务调用的信息。

```
aws ssm get-maintenance-window-execution-task-invocation \
    --window-execution-id "bc494bfa-e63b-49f6-8ad1-aa9f2EXAMPLE" \
    --task-id "96f2ad59-97e3-461d-a63d-40c8aEXAMPLE" \
    --invocation-id "a5273e2c-d2c6-4880-b3e1-5e550EXAMPLE"
```

输出：

```
{
    "Status": "SUCCESS",
    "Parameters": "{\"comment\":\"\",\"documentName\":\"AWS-RunPowerShellScript\",
\"instanceIds\":[\"i-1234567890EXAMPLE\"],\"maxConcurrency\":\"1\",\"maxErrors\":
\"1\",\"parameters\":{\"executionTimeout\":[\"3600\"],\"workingDirectory\":[\"\"],
\"commands\":[\"echo Hello\"]},\"timeoutSeconds\":600}",
    "ExecutionId": "03b6baa0-5460-4e15-83f2-ea685EXAMPLE",
    "InvocationId": "a5273e2c-d2c6-4880-b3e1-5e550EXAMPLE",
```

```
    "StartTime": 1549998326.421,
    "TaskType": "RUN_COMMAND",
    "EndTime": 1550001931.784,
    "WindowExecutionId": "bc494bfa-e63b-49f6-8ad1-aa9f2EXAMPLE",
    "StatusDetails": "Failed",
    "TaskExecutionId": "96f2ad59-97e3-461d-a63d-40c8aEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMaintenanceWindowExecutionTaskInvocation。

get-maintenance-window-execution-task

以下代码示例演示了如何使用 get-maintenance-window-execution-task。

Amazon CLI

获取有关维护时段任务执行的信息

以下 get-maintenance-window-execution-task 示例列出有关作为指定维护时段执行组成部分的任务的信息。

```
aws ssm get-maintenance-window-execution-task \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
```

输出：

```
{
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
    "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE",
    "TaskArn": "AWS-RunPatchBaseline",
    "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "Type": "RUN_COMMAND",
    "TaskParameters": [
        {
            "BaselineOverride": {
                "Values": [
                    ""
```

```
                ]
            },
            "InstallOverrideList": {
                "Values": [
                    ""
                ]
            },
            "Operation": {
                "Values": [
                    "Scan"
                ]
            },
            "RebootOption": {
                "Values": [
                    "RebootIfNeeded"
                ]
            },
            "SnapshotId": {
                "Values": [
                    "{{ aws:ORCHESTRATION_ID }}"
                ]
            },
            "aws:InstanceId": {
                "Values": [
                    "i-02573cafcfEXAMPLE",
                    "i-0471e04240EXAMPLE",
                    "i-07782c72faEXAMPLE"
                ]
            }
        }
    ],
    "Priority": 1,
    "MaxConcurrency": "1",
    "MaxErrors": "3",
    "Status": "SUCCESS",
    "StartTime": "2021-08-04T11:45:35.088000-07:00",
    "EndTime": "2021-08-04T11:53:09.079000-07:00"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMaintenanceWindowExecutionTask。

get-maintenance-window-execution

以下代码示例演示了如何使用 get-maintenance-window-execution。

Amazon CLI

获取有关维护时段任务执行的信息

以下 get-maintenance-window-execution 示例列出有关指定维护时段执行组成部分来执行的任务的信息。

```
aws ssm get-maintenance-window-execution \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

输出：

```
{
    "Status": "SUCCESS",
    "TaskIds": [
        "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
    ],
    "StartTime": 1487692834.595,
    "EndTime": 1487692835.051,
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关任务和任务执行的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMaintenanceWindowExecution。

get-maintenance-window-task

以下代码示例演示了如何使用 get-maintenance-window-task。

Amazon CLI

查看有关维护时段任务的信息

以下 get-maintenance-window-task 示例检索指定维护时段任务的详细信息。

```
aws ssm get-maintenance-window-task \
```

```
    --window-id mw-0c5ed765acEXAMPLE \
    --window-task-id 0e842a8d-2d44-4886-bb62-af8dcEXAMPLE
```

输出：

```
{
    "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "MaxErrors": "1",
    "TaskArn": "AWS-RunPowerShellScript",
    "MaxConcurrency": "1",
    "WindowTaskId": "0e842a8d-2d44-4886-bb62-af8dcEXAMPLE",
    "TaskParameters": {},
    "Priority": 1,
    "TaskInvocationParameters": {
        "RunCommand": {
            "Comment": "",
            "TimeoutSeconds": 600,
            "Parameters": {
                "commands": [
                    "echo Hello"
                ],
                "executionTimeout": [
                    "3600"
                ],
                "workingDirectory": [
                    ""
                ]
            }
        }
    },
    "WindowId": "mw-0c5ed765acEXAMPLE",
    "TaskType": "RUN_COMMAND",
    "Targets": [
        {
            "Values": [
                "84c818da-b619-4d3d-9651-946f3EXAMPLE"
            ],
            "Key": "WindowTargetIds"
        }
    ],
    "Name": "ExampleTask"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMaintenanceWindowTask。

get-maintenance-window

以下代码示例演示了如何使用 get-maintenance-window。

Amazon CLI

查看有关维护时段的信息

以下 get-maintenance-window 示例将检索指定维护时段的详细信息。

```
aws ssm get-maintenance-window \
    --window-id "mw-03eb9db428EXAMPLE"
```

输出：

```
{
    "AllowUnassociatedTargets": true,
    "CreatedDate": 1515006912.957,
    "Cutoff": 1,
    "Duration": 6,
    "Enabled": true,
    "ModifiedDate": 2020-01-01T10:04:04.099Z,
    "Name": "My-Maintenance-Window",
    "Schedule": "rate(3 days)",
    "WindowId": "mw-03eb9db428EXAMPLE",
    "NextExecutionTime": "2020-02-25T00:08:15.099Z"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看有关维护时段的信息（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetMaintenanceWindow。

get-ops-item

以下代码示例演示了如何使用 get-ops-item。

Amazon CLI

查看有关 OpsItem 的信息

以下 `get-ops-item` 示例显示指定 OpsItem 的详细信息。

```
aws ssm get-ops-item \
    --ops-item-id oi-0b725EXAMPLE
```

输出：

```
{
    "OpsItem": {
        "CreatedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
        "CreatedTime": "2019-12-04T15:52:16.793000-08:00",
        "Description": "CloudWatch Event Rule SSMOpsItems-EC2-instance-terminated
 was triggered. Your EC2 instance has terminated. See below for more details.",
        "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
        "LastModifiedTime": "2019-12-04T15:52:16.793000-08:00",
        "Notifications": [],
        "RelatedOpsItems": [],
        "Status": "Open",
        "OpsItemId": "oi-0b725EXAMPLE",
        "Title": "EC2 instance terminated",
        "Source": "EC2",
        "OperationalData": {
            "/aws/automations": {
                "Value": "[ { \"automationType\": \"AWS:SSM:Automation\",
 \"automationId\": \"AWS-CreateManagedWindowsInstance\" }, { \"automationType\":
 \"AWS:SSM:Automation\", \"automationId\": \"AWS-CreateManagedLinuxInstance\" } ]",
                "Type": "SearchableString"
            },
            "/aws/dedup": {
                "Value": "{\"dedupString\":\"SSMOpsItems-EC2-instance-terminated
\"}",
                "Type": "SearchableString"
            },
            "/aws/resources": {
                "Value": "[{\"arn\":\"arn:aws:ec2:us-east-2:111222333444:instance/
i-05adec7e97EXAMPLE\"}]",
                "Type": "SearchableString"
```

```
            },
            "event-time": {
                "Value": "2019-12-04T23:52:16Z",
                "Type": "String"
            },
            "instance-state": {
                "Value": "terminated",
                "Type": "String"
            }
        },
        "Category": "Availability",
        "Severity": "4"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOpsItem。

get-ops-summary

以下代码示例演示了如何使用 get-ops-summary。

Amazon CLI

查看所有 OpsItem 的摘要

以下 get-ops-summary 示例显示您 Amazon 账户中所有 OpsItem 的摘要。

```
aws ssm get-ops-summary
```

输出：

```
{
    "Entities": [
        {
            "Id": "oi-4309fEXAMPLE",
            "Data": {
                "AWS:OpsItem": {
                    "CaptureTime": "2020-02-26T18:58:32.918Z",
                    "Content": [
                        {
                            "AccountId": "111222333444",
```

```
                            "Category": "Availability",
                            "CreatedBy": "arn:aws:sts::111222333444:assumed-role/
OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                            "CreatedTime": "2020-02-26T19:10:44.149Z",
                            "Description": "CloudWatch Event Rule SSMOpsItems-EC2-
instance-terminated was triggered. Your EC2 instance has terminated. See below for
 more details.",
                            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-
role/OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                            "LastModifiedTime": "2020-02-26T19:10:44.149Z",
                            "Notifications": "",
                            "OperationalData": "{\"/aws/automations\":
{\"type\":\"SearchableString\",\"value\":\"[ { \\\"automationType\\\": \\
\"AWS:SSM:Automation\\\", \\\"automationId\\\": \\\"AWS-CreateManagedWindowsInstance
\\\" }, { \\\"automationType\\\": \\\"AWS:SSM:Automation\\\", \\\"automationId
\\\": \\\"AWS-CreateManagedLinuxInstance\\\" } ]\"},\"/aws/resources\":
{\"type\":\"SearchableString\",\"value\":\"[{\\\"arn\\\":\\\"arn:aws:ec2:us-
east-2:111222333444:instance/i-0acbd0800fEXAMPLE\\\"}]\"},\"/aws/dedup\":{\"type\":
\"SearchableString\",\"value\":\"{\\\"dedupString\\\":\\\"SSMOpsItems-EC2-instance-
terminated\\\"}\"}}",
                            "OpsItemId": "oi-4309fEXAMPLE",
                            "RelatedItems": "",
                            "Severity": "3",
                            "Source": "EC2",
                            "Status": "Open",
                            "Title": "EC2 instance terminated"
                        }
                    ]
                }
            }
        },
        {
            "Id": "oi-bb2a0e6a4541",
            "Data": {
                "AWS:OpsItem": {
                    "CaptureTime": "2019-11-26T19:20:06.161Z",
                    "Content": [
                        {
                            "AccountId": "111222333444",
                            "Category": "Availability",
                            "CreatedBy": "arn:aws:sts::111222333444:assumed-role/
OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                            "CreatedTime": "2019-11-26T20:00:07.237Z",
```

```
                            "Description": "CloudWatch Event Rule SSMOpsItems-SSM-
maintenance-window-execution-failed was triggered. Your SSM Maintenance Window
 execution has failed. See below for more details.",
                            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-
role/OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                            "LastModifiedTime": "2019-11-26T20:00:07.237Z",
                            "Notifications": "",
                            "OperationalData": "{\"/aws/resources\":{\"type
\":\"SearchableString\",\"value\":\"[{\\\"arn\\\":\\\"arn:aws:ssm:us-
east-2:111222333444:maintenancewindow/mw-0e83ba440dEXAMPLE\\\"}]\"},\"/aws/dedup\":
{\"type\":\"SearchableString\",\"value\":\"{\\\"dedupString\\\":\\\"SSMOpsItems-SSM-
maintenance-window-execution-failed\\\"}\"}}",
                            "OpsItemId": "oi-bb2a0EXAMPLE",
                            "RelatedItems": "",
                            "Severity": "3",
                            "Source": "SSM",
                            "Status": "Open",
                            "Title": "SSM Maintenance Window execution failed"
                        }
                    ]
                }
            }
        ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOpsSummary。

get-parameter-history

以下代码示例演示了如何使用 get-parameter-history。

Amazon CLI

获取参数的值历史记录

以下 get-parameter-history 示例列出指定参数的更改历史记录，包括其值。

```
aws ssm get-parameter-history \
    --name "MyStringParameter"
```

输出：


```
{
    "Parameters": [
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582154711.976,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the first version of my String parameter",
            "Value": "Veni",
            "Version": 1,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582156093.471,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the second version of my String parameter",
            "Value": "Vidi",
            "Version": 2,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582156117.545,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the third version of my String parameter",
            "Value": "Vici",
            "Version": 3,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数版本。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetParameterHistory。

get-parameter

以下代码示例演示了如何使用 get-parameter。

Amazon CLI

示例 1：显示参数的值

以下 get-parameter 示例列出指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyStringParameter"
```

输出：

```
{
    "Parameter": {
        "Name": "MyStringParameter",
        "Type": "String",
        "Value": "Veni",
        "Version": 1,
        "LastModifiedDate": 1530018761.888,
        "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

示例 2：解密 SecureString 参数的值

以下 get-parameter 示例解密指定 SecureString 参数的值。

```
aws ssm get-parameter \
    --name "MySecureStringParameter" \
    --with-decryption
```

输出：

```
{
    "Parameter": {
        "Name": "MySecureStringParameter",
        "Type": "SecureString",
        "Value": "16679b88-310b-4895-a943-e0764EXAMPLE",
        "Version": 2,
        "LastModifiedDate": 1582155479.205,
        "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/
MySecureStringParameter"
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

示例 3：使用标签显示参数的值

以下 `get-parameter` 示例列出具有指定标签的指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

输出：

```
{
    "Parameter": {
        "Name": "MyParameter",
        "Type": "String",
        "Value": "parameter version 2",
        "Version": 2,
        "Selector": ":label",
        "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
        "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数标签。

示例 4：使用版本显示参数的值

以下 get-parameter 示例列出指定单个参数版本的值。

```
aws ssm get-parameter \
    --name "MyParameter:2"
```

输出：

```
{
    "Parameter": {
        "Name": "MyParameter",
        "Type": "String",
        "Value": "parameter version 2",
        "Version": 2,
        "Selector": ":2",
        "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
        "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数标签。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考中的 GetParameter。

get-parameters-by-path

以下代码示例演示了如何使用 get-parameters-by-path。

Amazon CLI

列出指定路径中的参数

以下 get-parameters-by-path 示例列出指定层次结构中的参数。

```
aws ssm get-parameters-by-path \
    --path "/site/newyork/department/"
```

输出：

```
{
    "Parameters": [
```

```
        {
            "Name": "/site/newyork/department/marketing",
            "Type": "String",
            "Value": "Floor 2",
            "Version": 1,
            "LastModifiedDate": 1530018761.888,
            "ARN": "arn:aws:ssm:us-east-1:111222333444:parameter/site/newyork/
department/marketing"
        },
        {
            "Name": "/site/newyork/department/infotech",
            "Type": "String",
            "Value": "Floor 3",
            "Version": 1,
            "LastModifiedDate": 1530018823.429,
            "ARN": "arn:aws:ssm:us-east-1:111222333444:parameter/site/newyork/
department/infotech"
        },
        ...
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数层次结构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetParametersByPath。

get-parameters

以下代码示例演示了如何使用 get-parameters。

Amazon CLI

示例 1：列出参数的值

以下 get-parameters 示例列出三个指定参数的值。

```
aws ssm get-parameters \
    --names "MyStringParameter" "MyStringListParameter" "MyInvalidParameterName"
```

输出：

```
{
```

```
    "Parameters": [
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "Value": "alpha,beta,gamma",
            "Version": 1,
            "LastModifiedDate": 1582154764.222,
            "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/
MyStringListParameter"
            "DataType": "text"
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "Value": "Vici",
            "Version": 3,
            "LastModifiedDate": 1582156117.545,
            "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"
            "DataType": "text"
        }
    ],
    "InvalidParameters": [
        "MyInvalidParameterName"
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

示例 2：使用 ``--query`` 选项列出多个参数的名称和值

以下 `get-parameters` 示例列出指定参数的名称和值。

```
aws ssm get-parameters \
    --names MyStringParameter MyStringListParameter \
    --query "Parameters[*].{Name:Name,Value:Value}"
```

输出：

```
[
    {
        "Name": "MyStringListParameter",
        "Value": "alpha,beta,gamma"
    },
```

```
    {
        "Name": "MyStringParameter",
        "Value": "Vidi"
    }
]
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Parameter Store。

示例 3：使用标签显示参数的值

以下 get-parameter 示例列出具有指定标签的指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MyLabelParameter",
            "Type": "String",
            "Value": "parameter by label",
            "Version": 1,
            "Selector": ":label",
            "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
            "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
            "DataType": "text"
        },
        {
            "Name": "MyVersionParameter",
            "Type": "String",
            "Value": "parameter by version",
            "Version": 2,
            "Selector": ":2",
            "LastModifiedDate": "2021-03-24T16:20:28.236000-07:00",
            "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/unlabel-param",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数标签。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetParameters。

get-patch-baseline-for-patch-group

以下代码示例演示了如何使用 get-patch-baseline-for-patch-group。

Amazon CLI

显示补丁组的补丁基准

以下 get-patch-baseline-for-patch-group 示例检索有关指定补丁组补丁基准的详细信息。

```
aws ssm get-patch-baseline-for-patch-group \
    --patch-group "DEV"
```

输出：

```
{
    "PatchGroup": "DEV",
    "BaselineId": "pb-0123456789abcdef0",
    "OperatingSystem": "WINDOWS"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetPatchBaselineForPatchGroup。

get-patch-baseline

以下代码示例演示了如何使用 get-patch-baseline。

Amazon CLI

显示补丁基准

以下 get-patch-baseline 示例检索指定补丁基准的详细信息。

```
aws ssm get-patch-baseline \
    --baseline-id "pb-0123456789abcdef0"
```

输出：

```
{
    "BaselineId": "pb-0123456789abcdef0",
    "Name": "WindowsPatching",
    "OperatingSystem": "WINDOWS",
    "GlobalFilters": {
        "PatchFilters": []
    },
    "ApprovalRules": {
        "PatchRules": [
            {
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "PRODUCT",
                            "Values": [
                                "WindowsServer2016"
                            ]
                        }
                    ]
                },
                "ComplianceLevel": "CRITICAL",
                "ApproveAfterDays": 0,
                "EnableNonSecurity": false
            }
        ]
    },
    "ApprovedPatches": [],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "RejectedPatches": [],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "PatchGroups": [
        "QA",
        "DEV"
    ],
    "CreatedDate": 1550244180.465,
```

```
    "ModifiedDate": 1550244180.465,
    "Description": "Patches for Windows Servers",
    "Sources": []
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetPatchBaseline。

get-service-setting

以下代码示例演示了如何使用 get-service-setting。

Amazon CLI

检索 Parameter Store 吞吐量的服务设置

以下 get-service-setting 示例检索指定区域中 Parameter Store 吞吐量的当前服务设置。

```
aws ssm get-service-setting \
    --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-store/high-throughput-enabled
```

输出：

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/parameter-store/high-throughput-enabled",
        "SettingValue": "false",
        "LastModifiedDate": 1555532818.578,
        "LastModifiedUser": "System",
        "ARN": "arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-store/high-throughput-enabled",
        "Status": "Default"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的提高 Parameter Store 吞吐量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceSetting。

label-parameter-version

以下代码示例演示了如何使用 label-parameter-version。

Amazon CLI

示例 1：为最新版本的参数添加标签

以下 label-parameter-version 示例为最新版本的指定参数添加标签。

```
aws ssm label-parameter-version \
    --name "MyStringParameter" \
    --labels "ProductionReady"
```

输出：

```
{
    "InvalidLabels": [],
    "ParameterVersion": 3
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数标签。

示例 2：为指定版本的参数添加标签

以下 label-parameter-version 示例为指定版本的参数添加标签。

```
aws ssm label-parameter-version \
    --name "MyStringParameter" \
    --labels "ProductionReady" \
    --parameter-version "2" --labels "DevelopmentReady"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用参数标签。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 LabelParameterVersion。

list-association-versions

以下代码示例演示了如何使用 list-association-versions。

Amazon CLI

列出特定关联 ID 的关联的所有版本

以下 list-association-versions 示例列出指定关联的所有版本。

```
aws ssm list-association-versions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
"AssociationVersions": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "CreatedDate": 1550505536.726,
            "Name": "AWS-UpdateSSMAgent",
            "Parameters": {
                "allowDowngrade": [
                    "false"
                ],
                "version": [
                    ""
                ]
            },
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-1234567890abcdef0"
                    ]
                }
            ],
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAssociationVersions。

list-associations

以下代码示例演示了如何使用 list-associations。

Amazon CLI

示例 1：列出特定实例的关联

以下 list-associations 示例列出具有 AssociationName、UpdateSSMAgent 的所有关联。

```
aws ssm list-associations /
    --association-filter-list "key=AssociationName,value=UpdateSSMAgent"
```

输出：

```
{
    "Associations": [
        {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-1234567890abcdef0",
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-016648b75dd622dab"
                    ]
                }
            ],
            "Overview": {
                "Status": "Pending",
                "DetailedStatus": "Associated",
                "AssociationStatusAggregatedCount": {
                    "Pending": 1
                }
            },
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        }
    ]
}
```

有关更多信息，请参阅《Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

示例 2：列出特定文档的关联

以下 list-associations 示例列出指定文档的所有关联。

```
aws ssm list-associations /
    --association-filter-list "key=Name,value=AWS-UpdateSSMAgent"
```

输出：

```
{
    "Associations": [
        {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-1234567890abcdef0",
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-1234567890abcdef0"
                    ]
                }
            ],
            "LastExecutionDate": 1550505828.548,
            "Overview": {
                "Status": "Success",
                "DetailedStatus": "Success",
                "AssociationStatusAggregatedCount": {
                    "Success": 1
                }
            },
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        },
    {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-9876543210abcdef0",
            "AssociationId": "fbc07ef7-b985-4684-b82b-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
```

```
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-9876543210abcdef0"
                    ]
                }
            ],
            "LastExecutionDate": 1550507531.0,
            "Overview": {
                "Status": "Success",
                "AssociationStatusAggregatedCount": {
                    "Success": 1
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListAssociations。

`list-command-invocations`

以下代码示例演示了如何使用 `list-command-invocations`。

Amazon CLI

列出特定命令的调用

以下 `list-command-invocations` 示例列出命令的所有调用。

```
aws ssm list-command-invocations \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --details
```

输出：

```
{
    "CommandInvocations": [
        {
            "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
            "InstanceId": "i-02573cafcfEXAMPLE",
```

```
            "InstanceName": "",
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
            "DocumentVersion": "",
            "RequestedDateTime": 1582136283.089,
            "Status": "Success",
            "StatusDetails": "Success",
            "StandardOutputUrl": "",
            "StandardErrorUrl": "",
            "CommandPlugins": [
                {
                    "Name": "aws:updateSsmAgent",
                    "Status": "Success",
                    "StatusDetails": "Success",
                    "ResponseCode": 0,
                    "ResponseStartDateTime": 1582136283.419,
                    "ResponseFinishDateTime": 1582136283.51,
                    "Output": "Updating amazon-ssm-agent from 2.3.842.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been installed,
 update skipped\n",
                    "StandardOutputUrl": "",
                    "StandardErrorUrl": "",
                    "OutputS3Region": "us-east-2",
                    "OutputS3BucketName": "",
                    "OutputS3KeyPrefix": ""
                }
            ],
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        },
        {
            "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
            "InstanceId": "i-0471e04240EXAMPLE",
            "InstanceName": "",
```

```
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
            "DocumentVersion": "",
            "RequestedDateTime": 1582136283.02,
            "Status": "Success",
            "StatusDetails": "Success",
            "StandardOutputUrl": "",
            "StandardErrorUrl": "",
            "CommandPlugins": [
                {
                    "Name": "aws:updateSsmAgent",
                    "Status": "Success",
                    "StatusDetails": "Success",
                    "ResponseCode": 0,
                    "ResponseStartDateTime": 1582136283.812,
                    "ResponseFinishDateTime": 1582136295.031,
                    "Output": "Updating amazon-ssm-agent from 2.3.672.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
ssm-agent-manifest.json\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/
amazon-ssm-us-east-2/amazon-ssm-agent-updater/2.3.842.0/amazon-ssm-agent-updater-
snap-amd64.tar.gz\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/
amazon-ssm-us-east-2/amazon-ssm-agent/2.3.672.0/amazon-ssm-agent-snap-amd64.tar.gz
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
amazon-ssm-agent/2.3.842.0/amazon-ssm-agent-snap-amd64.tar.gz\nInitiating amazon-
ssm-agent update to 2.3.842.0\namazon-ssm-agent updated successfully to 2.3.842.0",
                    "StandardOutputUrl": "",
                    "StandardErrorUrl": "",
                    "OutputS3Region": "us-east-2",
                    "OutputS3BucketName": "",
                    "OutputS3KeyPrefix": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE/
i-0471e04240EXAMPLE/awsupdateSsmAgent"
                }
            ],
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
```

```
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的了解命令状态。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListCommandInvocations。

list-commands

以下代码示例演示了如何使用 list-commands。

Amazon CLI

示例 1：获取特定命令的状态

以下 list-commands 示例检索并显示指定命令的状态。

```
aws ssm list-commands \
    --command-id "0831e1a8-a1ac-4257-a1fd-c831bEXAMPLE"
```

示例 2：获取特定日期之后请求的命令的状态

以下 list-commands 示例检索在指定日期之后请求的命令的详细信息。

```
aws ssm list-commands \
    --filter "key=InvokedAfter,value=2020-02-01T00:00:00Z"
```

示例 3：列出 Amazon 账户中请求的所有命令

以下 list-commands 示例列出了当前 Amazon 账户和区域中用户请求的所有命令。

```
aws ssm list-commands
```

输出：

```
{
    "Commands": [
        {
            "CommandId": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
```

```
            "DocumentVersion": "",
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "ExpiresAfter": "2020-02-19T11:28:02.500000-08:00",
            "Parameters": {},
            "InstanceIds": [
                "i-028ea792daEXAMPLE",
                "i-02feef8c46EXAMPLE",
                "i-038613f3f0EXAMPLE",
                "i-03a530a2d4EXAMPLE",
                "i-083b678d37EXAMPLE",
                "i-0dee81debaEXAMPLE"
            ],
            "Targets": [],
            "RequestedDateTime": "2020-02-19T10:18:02.500000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "",
            "OutputS3KeyPrefix": "",
            "MaxConcurrency": "50",
            "MaxErrors": "100%",
            "TargetCount": 6,
            "CompletedCount": 6,
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
        {
            "CommandId": "e9ade581-c03d-476b-9b07-26667EXAMPLE",
            "DocumentName": "AWS-FindWindowsUpdates",
            "DocumentVersion": "1",
            "Comment": "",
            "ExpiresAfter": "2020-01-24T12:37:31.874000-08:00",
            "Parameters": {
                "KbArticleIds": [
```

```
                    ""
                ],
                "UpdateLevel": [
                    "All"
                ]
            },
            "InstanceIds": [],
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-00ec29b21eEXAMPLE",
                        "i-09911ddd90EXAMPLE"
                    ]
                }
            ],
            "RequestedDateTime": "2020-01-24T11:27:31.874000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "my-us-east-2-bucket",
            "OutputS3KeyPrefix": "my-rc-output",
            "MaxConcurrency": "50",
            "MaxErrors": "0",
            "TargetCount": 2,
            "CompletedCount": 2,
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-east-2-
notification-arn",
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Invocation"
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
        {
            "CommandId": "d539b6c3-70e8-4853-80e5-0ce4fEXAMPLE",
```

```
            "DocumentName": "AWS-RunPatchBaseline",
            "DocumentVersion": "1",
            "Comment": "",
            "ExpiresAfter": "2020-01-24T12:21:04.350000-08:00",
            "Parameters": {
                "InstallOverrideList": [
                    ""
                ],
                "Operation": [
                    "Install"
                ],
                "RebootOption": [
                    "RebootIfNeeded"
                ],
                "SnapshotId": [
                    ""
                ]
            },
            "InstanceIds": [],
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-00ec29b21eEXAMPLE",
                        "i-09911ddd90EXAMPLE"
                    ]
                }
            ],
            "RequestedDateTime": "2020-01-24T11:11:04.350000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "my-us-east-2-bucket",
            "OutputS3KeyPrefix": "my-rc-output",
            "MaxConcurrency": "50",
            "MaxErrors": "0",
            "TargetCount": 2,
            "CompletedCount": 2,
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-east-2-
notification-arn",
```

```
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Invocation"
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListCommands。

`list-compliance-items`

以下代码示例演示了如何使用 `list-compliance-items`。

Amazon CLI

列出特定实例的合规性项目

此示例列出指定实例的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types "ManagedInstance"
```

输出：

```
{
  "ComplianceItems": [
      {
          "ComplianceType": "Association",
          "ResourceType": "ManagedInstance",
          "ResourceId": "i-1234567890abcdef0",
          "Id": "8dfe3659-4309-493a-8755-0123456789ab",
          "Title": "",
```

```
            "Status": "COMPLIANT",
            "Severity": "UNSPECIFIED",
            "ExecutionSummary": {
                "ExecutionTime": 1550408470.0
            },
            "Details": {
                "DocumentName": "AWS-GatherSoftwareInventory",
                "DocumentVersion": "1"
            }
        },
        {
            "ComplianceType": "Association",
            "ResourceType": "ManagedInstance",
            "ResourceId": "i-1234567890abcdef0",
            "Id": "e4c2ed6d-516f-41aa-aa2a-0123456789ab",
            "Title": "",
            "Status": "COMPLIANT",
            "Severity": "UNSPECIFIED",
            "ExecutionSummary": {
                "ExecutionTime": 1550508475.0
            },
            "Details": {
                "DocumentName": "AWS-UpdateSSMAgent",
                "DocumentVersion": "1"
            }
        },
                ...
    ],
    "NextToken": "--token string truncated--"
}
```

列出特定实例和关联 ID 的合规性项目

此示例列出指定实例和关联 ID 的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --
filters "Key=ComplianceType,Values=Association,Type=EQUAL" "Key=Id,Values=e4c2ed6d-516f-41aa
aa2a-0123456789ab,Type=EQUAL"
```

列出特定日期和时间之后实例的合规性项目

此示例列出指定日期和时间之后实例的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --
filters "Key=ExecutionTime,Values=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListComplianceItems。

list-compliance-summaries

以下代码示例演示了如何使用 list-compliance-summaries。

Amazon CLI

列出所有合规性类型的合规性摘要

此示例列出您账户中所有合规性类型的合规性摘要。

命令:

```
aws ssm list-compliance-summaries
```

输出：

```
{
  "ComplianceSummaryItems": [
      {
          "ComplianceType": "Association",
          "CompliantSummary": {
              "CompliantCount": 2,
              "SeveritySummary": {
                  "CriticalCount": 0,
                  "HighCount": 0,
                  "MediumCount": 0,
                  "LowCount": 0,
                  "InformationalCount": 0,
                  "UnspecifiedCount": 2
              }
          },
          "NonCompliantSummary": {
```

```
                "NonCompliantCount": 0,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 0
                }
            }
        },
        {
            "ComplianceType": "Patch",
            "CompliantSummary": {
                "CompliantCount": 1,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 1
                }
            },
            "NonCompliantSummary": {
                "NonCompliantCount": 1,
                "SeveritySummary": {
                    "CriticalCount": 1,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 0
                }
            }
        },
                ...
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

列出特定合规性类型的合规性摘要

此示例列出补丁合规性类型的合规性摘要。

命令:

```
aws ssm list-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListComplianceSummaries。

list-document-metadata-history

以下代码示例演示了如何使用 list-document-metadata-history。

Amazon CLI

示例：查看更改模板的批准历史记录和状态

以下 list-document-metadata-history 示例返回指定 Change Manager 变更模板的批准历史记录。

```
aws ssm list-document-metadata-history \
    --name MyChangeManageTemplate \
    --metadata DocumentReviews
```

输出：

```
{
    "Name": "MyChangeManagerTemplate",
    "DocumentVersion": "1",
    "Author": "arn:aws:iam::111222333444;:user/JohnDoe",
    "Metadata": {
        "ReviewerResponse": [
            {
                "CreateTime": "2021-07-30T11:58:28.025000-07:00",
                "UpdatedTime": "2021-07-30T12:01:19.274000-07:00",
                "ReviewStatus": "APPROVED",
                "Comment": [
                    {
                        "Type": "COMMENT",
                        "Content": "I approve this template version"
                    }
                ],
```

```
                "Reviewer": "arn:aws:iam::111222333444;:user/ShirleyRodriguez"
            },
            {
                "CreateTime": "2021-07-30T11:58:28.025000-07:00",
                "UpdatedTime": "2021-07-30T11:58:28.025000-07:00",
                "ReviewStatus": "PENDING"
            }
        ]
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的审查、批准或拒绝变更模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDocumentMetadataHistory。

list-document-versions

以下代码示例演示了如何使用 list-document-versions。

Amazon CLI

列出文档版本

以下 list-document-versions 示例列出 Systems Manager 文档的所有版本。

```
aws ssm list-document-versions \
    --name "Example"
```

输出：

```
{
    "DocumentVersions": [
        {
            "Name": "Example",
            "DocumentVersion": "1",
            "CreatedDate": 1583257938.266,
            "IsDefaultVersion": true,
            "DocumentFormat": "YAML",
            "Status": "Active"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的发送使用文档版本参数的命令。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDocumentVersions。

`list-documents`

以下代码示例演示了如何使用 `list-documents`。

Amazon CLI

示例 1：列出文档

以下 `list-documents` 示例列出标有自定义标签的请求账户拥有的文档。

```
aws ssm list-documents \
    --filters Key=Owner,Values=Self Key=tag:DocUse,Values=Testing
```

输出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "29884EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Automation",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": [
                {
                    "Key": "DocUse",
                    "Value": "Testing"
                }
            ]
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Amazon Systems Manager 文档。

示例 2：列出共享文档

以下 `list-documents` 示例列出共享文档，包括不属于 Amazon 的私有共享文档。

```
aws ssm list-documents \
    --filters Key=Name,Values=sharedDocNamePrefix  Key=Owner,Values=Private
```

输出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "12345EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Command",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的 Amazon Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListDocuments。

list-inventory-entries

以下代码示例演示了如何使用 `list-inventory-entries`。

Amazon CLI

示例 1：查看实例的特定清单类型条目

以下 `list-inventory-entries` 示例列出特定实例上 Amazon:Application 清单类型的清单条目。

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "AWS:Application"
```

输出：

```
{
  "TypeName": "AWS:Application",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.1",
  "CaptureTime": "2019-02-15T12:17:55Z",
  "Entries": [
    {
      "Architecture": "i386",
      "Name": "Amazon SSM Agent",
      "PackageId": "{88a60be2-89a1-4df8-812a-80863c2a2b68}",
      "Publisher": "Amazon Web Services",
      "Version": "2.3.274.0"
    },
    {
      "Architecture": "x86_64",
      "InstalledTime": "2018-05-03T13:42:34Z",
      "Name": "AmazonCloudWatchAgent",
      "Publisher": "",
      "Version": "1.200442.0"
    }
  ]
}
```

示例 2：查看分配给实例的自定义清单条目

以下 `list-inventory-entries` 示例列出分配给实例的自定义清单条目。

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
```

```
    --type-name "Custom:RackInfo"
```

输出：

```
{
  "TypeName": "Custom:RackInfo",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.0",
  "CaptureTime": "2021-05-22T10:01:01Z",
  "Entries": [
    {
      "RackLocation": "Bay B/Row C/Rack D/Shelf E"
    }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListInventoryEntries。

list-ops-item-related-items

以下代码示例演示了如何使用 list-ops-item-related-items。

Amazon CLI

列出 OpsItem 的相关项目资源

以下 list-ops-item-related-items 示例列出 OpsItem 的相关项目资源。

```
aws ssm list-ops-item-related-items \
    --ops-item-id "oi-f99f2EXAMPLE"
```

输出：

```
{
    "Summaries": [
        {
            "OpsItemId": "oi-f99f2EXAMPLE",
            "AssociationId": "e2036148-cccb-490e-ac2a-390e5EXAMPLE",
            "ResourceType": "AWS::SSMIncidents::IncidentRecord",
            "AssociationType": "IsParentOf",
            "ResourceUri": "arn:aws:ssm-incidents::111122223333:incident-record/
example-response/64bd9b45-1d0e-2622-840d-03a87a1451fa",
```

```
            "CreatedBy": {
                "Arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForIncidentManager/IncidentResponse"
            },
            "CreatedTime": "2021-08-11T18:47:14.994000+00:00",
            "LastModifiedBy": {
                "Arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForIncidentManager/IncidentResponse"
            },
            "LastModifiedTime": "2021-08-11T18:47:14.994000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 OpsCenter 中处理 Incident Manager 事件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOpsItemRelatedItems。

list-resource-compliance-summaries

以下代码示例演示了如何使用 `list-resource-compliance-summaries`。

Amazon CLI

列出资源级合规性摘要计数

此示例列出资源级合规性摘要计数。

命令:

```
aws ssm list-resource-compliance-summaries
```

输出：

```
{
  "ResourceComplianceSummaryItems": [
      {
          "ComplianceType": "Association",
          "ResourceType": "ManagedInstance",
          "ResourceId": "i-1234567890abcdef0",
          "Status": "COMPLIANT",
          "OverallSeverity": "UNSPECIFIED",
```

```
            "ExecutionSummary": {
                "ExecutionTime": 1550509273.0
            },
            "CompliantSummary": {
                "CompliantCount": 2,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 2
                }
            },
            "NonCompliantSummary": {
                "NonCompliantCount": 0,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 0
                }
            }
        },
        {
            "ComplianceType": "Patch",
            "ResourceType": "ManagedInstance",
            "ResourceId": "i-9876543210abcdef0",
            "Status": "COMPLIANT",
            "OverallSeverity": "UNSPECIFIED",
            "ExecutionSummary": {
                "ExecutionTime": 1550248550.0,
                "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
                "ExecutionType": "Command"
            },
            "CompliantSummary": {
                "CompliantCount": 397,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
```

```
                "InformationalCount": 0,
                "UnspecifiedCount": 397
            }
        },
        "NonCompliantSummary": {
            "NonCompliantCount": 0,
            "SeveritySummary": {
                "CriticalCount": 0,
                "HighCount": 0,
                "MediumCount": 0,
                "LowCount": 0,
                "InformationalCount": 0,
                "UnspecifiedCount": 0
            }
        }
    }
  ],
  "NextToken": "--token string truncated--"
}
```

列出特定合性规类型的资源级合规性摘要

此示例列出补丁合规性类型的资源级合规性摘要。

命令:

```
aws ssm list-resource-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceComplianceSummaries。

list-resource-data-sync

以下代码示例演示了如何使用 list-resource-data-sync。

Amazon CLI

列出资源数据同步配置

此示例检索有关您资源数据同步配置的信息。

```
aws ssm list-resource-data-sync
```

输出：

```
{
    "ResourceDataSyncItems": [
        {
            "SyncName": "MyResourceDataSync",
            "S3Destination": {
                "BucketName": "ssm-resource-data-sync",
                "SyncFormat": "JsonSerDe",
                "Region": "us-east-1"
            },
            "LastSyncTime": 1550261472.003,
            "LastSuccessfulSyncTime": 1550261472.003,
            "LastStatus": "Successful",
            "SyncCreatedTime": 1543235736.72,
            "LastSyncStatusMessage": "The sync was successfully completed"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceDataSync。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出应用于补丁基准的标签

以下 list-tags-for-resource 示例列出了补丁基准的标签。

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0"
```

输出：

```
{
```

```
    "TagList": [
        {
            "Key": "Environment",
            "Value": "Production"
        },
        {
            "Key": "Region",
            "Value": "EMEA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon 一般参考》中的标记 Amazon 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`modify-document-permission`

以下代码示例演示了如何使用 `modify-document-permission`。

Amazon CLI

修改文档权限

以下 `modify-document-permission` 示例公开共享一个 Systems Manager 文档。

```
aws ssm modify-document-permission \
    --name "Example" \
    --permission-type "Share" \
    --account-ids-to-add "All"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的共享 Systems Manager 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ModifyDocumentPermission。

`put-compliance-items`

以下代码示例演示了如何使用 `put-compliance-items`。

Amazon CLI

向指定实例注册合规性类型和合规性详细信息

此示例将合规性类型 Custom:AVCheck 注册到指定的托管式实例。如果此命令成功，则无任何输出。

命令:

```
aws ssm put-compliance-items --resource-id "i-1234567890abcdef0" --
resource-type "ManagedInstance" --compliance-type "Custom:AVCheck"
 --execution-summary "ExecutionTime=2019-02-18T16:00:00Z" --
items "Id=Version2.0,Title=ScanHost,Severity=CRITICAL,Status=COMPLIANT"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutComplianceItems。

put-inventory

以下代码示例演示了如何使用 put-inventory。

Amazon CLI

将客户元数据分配给实例

此示例将机架位置信息分配给某个实例。如果此命令成功，则无任何输出。

命令（Linux）：

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --items
 '[{"TypeName": "Custom:RackInfo","SchemaVersion": "1.0","CaptureTime":
 "2019-01-22T10:01:01Z","Content":[{"RackLocation": "Bay B/Row C/Rack D/Shelf
 E"}]}]'
```

命令（Windows）：

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --
items "TypeName=Custom:RackInfo,SchemaVersion=1.0,CaptureTime=2019-01-22T10:01:01Z,Content=[
 B/Row C/Rack D/Shelf F'}]"
```

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 PutInventory。

put-parameter

以下代码示例演示了如何使用 put-parameter。

Amazon CLI

示例 1：更改参数值

以下 put-parameter 示例更改了指定参数的值。

```
aws ssm put-parameter \
    --name "MyStringParameter" \
    --type "String" \
    --value "Vici" \
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Standard"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 参数（Amazon CLI）、管理参数层和处理参数策略。

示例 2：创建高级参数

以下 put-parameter 示例将创建高级参数。

```
aws ssm put-parameter \
    --name "MyAdvancedParameter" \
    --description "This is an advanced parameter" \
    --value "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
 tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
 quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat
 [truncated]" \
    --type "String" \
    --tier Advanced
```

输出：

```
{
```

```
    "Version": 1,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 参数（Amazon CLI）、管理参数层和处理参数策略。

示例 3：将标准参数转换为高级参数

以下 put-parameter 示例将现有标准参数转换为高级参数。

```
aws ssm put-parameter \
    --name "MyConvertedParameter" \
    --value "abc123" \
    --type "String" \
    --tier Advanced \
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 参数（Amazon CLI）、管理参数层和处理参数策略。

示例 4：创建附加有策略的参数

以下 put-parameter 示例创建了一个附加参数策略的高级参数。

```
aws ssm put-parameter \
    --name "/Finance/Payroll/q2accesskey" \
    --value "P@sSwW)rd" \
    --type "SecureString" \
    --tier Advanced \
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification\",
\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},{\"Type\":
\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":\"60\",\"Unit
\":\"Days\"}}]"
```

输出：

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 参数（Amazon CLI）、管理参数层和处理参数策略。

示例 5：向现有参数添加策略

以下 put-parameter 示例将策略附加到现有高级参数。

```
aws ssm put-parameter \
    --name "/Finance/Payroll/q2accesskey" \
    --value "N3wP@sSwW)rd" \
    --type "SecureString" \
    --tier Advanced \
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification\",
\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},{\"Type\":
\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":\"60\",\"Unit
\":\"Days\"}}]"
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建 Systems Manager 参数（Amazon CLI）、管理参数层和处理参数策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutParameter。

register-default-patch-baseline

以下代码示例演示了如何使用 register-default-patch-baseline。

Amazon CLI

设置默认补丁基准

以下 register-default-patch-baseline 示例将指定的自定义补丁基准注册为其支持的操作系统类型的默认补丁基准。

```
aws ssm register-default-patch-baseline \
    --baseline-id "pb-abc123cf9bEXAMPLE"
```

输出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

以下 register-default-patch-baseline 示例将 Amazon 为 CentOS 提供的默认补丁基准注册为默认补丁基准。

```
aws ssm register-default-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0574b43a65ea646ed"
```

输出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的关于预定义和自定义补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RegisterDefaultPatchBaseline。

register-patch-baseline-for-patch-group

以下代码示例演示了如何使用 register-patch-baseline-for-patch-group。

Amazon CLI

为补丁组注册补丁基准

以下 register-patch-baseline-for-patch-group 示例为补丁组注册补丁基准。

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id "pb-045f10b4f382baeda" \
    --patch-group "Production"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f382baeda",
    "PatchGroup": "Production"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RegisterPatchBaselineForPatchGroup。

register-target-with-maintenance-window

以下代码示例演示了如何使用 register-target-with-maintenance-window。

Amazon CLI

示例 1：向维护时段注册单个目标

以下 register-target-with-maintenance-window 示例向维护时段注册实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862" \
    --owner-information "Single instance" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 2：使用实例 ID 向维护时段注册多个目标

以下 register-target-with-maintenance-window 示例通过指定其实例 ID 向维护时段注册两个实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862,i-0cb2b964d3e14fd9f" \
    --owner-information "Two instances in a list" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 3：使用资源标签向维护时段注册目标

以下 register-target-with-maintenance-window 示例通过指定已应用于实例的资源标签，向维护时段注册实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-06cf17cbefcb4bf4f" \
    --targets "Key=tag:Environment,Values=Prod" "Key=Role,Values=Web" \
    --owner-information "Production Web Servers" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 4：使用一组标签键注册目标

以下 register-target-with-maintenance-window 示例注册所有被分配了一个或多个标签键的实例（不考虑其键值）。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 5：使用资源组名称注册目标

以下 register-target-with-maintenance-window 示例注册指定的资源组，无论其包含的资源类型如何。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册目标实例（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RegisterTargetWithMaintenanceWindow。

register-task-with-maintenance-window

以下代码示例演示了如何使用 register-task-with-maintenance-window。

Amazon CLI

示例 1：向维护时段注册 Automation 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Automation 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649EXAMPLE" \
    --targets Key=InstanceIds,Values=i-1234520122EXAMPLE \
    --task-arn AWS-RestartEC2Instance \
    --service-role-arn arn:aws:iam::111222333444:role/SSM --task-type AUTOMATION \
    --task-invocation-parameters "{\"Automation\":{\"DocumentVersion\":\"\$LATEST\",
\"Parameters\":{\"InstanceId\":[\"{{RESOURCE_ID}}\"]}}}" \
    --priority 0 \
    --max-concurrency 1 \
    --max-errors 1 \
    --name "AutomationExample" \
    --description "Restarting EC2 Instance for maintenance"
```

输出：

```
{
    "WindowTaskId":"11144444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册任务（Amazon CLI）。

示例 2：向维护时段注册 Lambda 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Lambda 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649dee04e4" \
    --targets Key=InstanceIds,Values=i-12344d305eEXAMPLE \
    --task-arn arn:aws:lambda:us-east-1:111222333444:function:SSMTestLAMBDA \
    --service-role-arn arn:aws:iam::111222333444:role/SSM \
    --task-type LAMBDA \
    --task-invocation-parameters '{"Lambda":{"Payload":"{\"InstanceId\":
\"{{RESOURCE_ID}}\",\"targetType\":\"{{TARGET_TYPE}}\"}","Qualifier":"$LATEST"}}' \
```

```
    --priority 0 \
    --max-concurrency 10 \
    --max-errors 5 \
    --name "Lambda_Example" \
    --description "My Lambda Example"
```

输出：

```
{
    "WindowTaskId":"22244444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册任务（Amazon CLI）。

示例 3：向维护时段注册 Run Command 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Run Command 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649dee04e4" \
    --targets "Key=InstanceIds,Values=i-12344d305eEXAMPLE" \
    --service-role-arn "arn:aws:iam::111222333444:role/SSM" \
    --task-type "RUN_COMMAND" \
    --name "SSMInstallPowerShellModule" \
    --task-arn "AWS-InstallPowerShellModule" \
    --task-invocation-parameters "{\"RunCommand\":{\"Comment\":\"\",
\"OutputS3BucketName\":\"runcommandlogs\",\"Parameters\":{\"commands\":[\"Get-
Module -ListAvailable\"],\"executionTimeout\":[\"3600\"],\"source\":[\"https:\/
\/gallery.technet.microsoft.com\/EZOut-33ae0fb7\/file\/110351\/1\/EZOut.zip\"],
\"workingDirectory\":[\"\\\\\"]},\"TimeoutSeconds\":600}}" \
    --max-concurrency 1 \
    --max-errors 1 \
    --priority 10
```

输出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册任务（Amazon CLI）。

示例 4：向维护时段注册 Step Functions 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Step Functions 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-1234d787d6EXAMPLE" \
    --targets Key=WindowTargetIds,Values=12347414-69c3-49f8-95b8-ed2dcEXAMPLE \
    --task-arn arn:aws:states:us-
east-1:111222333444:stateMachine:SSMTestStateMachine \
    --service-role-arn arn:aws:iam::111222333444:role/MaintenanceWindows \
    --task-type STEP_FUNCTIONS \
    --task-invocation-parameters '{"StepFunctions":{"Input":"{\"InstanceId\":
\"{{RESOURCE_ID}}\"}"}}' \
    --priority 0 \
    --max-concurrency 10 \
    --max-errors 5 \
    --name "Step_Functions_Example" \
    --description "My Step Functions Example"
```

输出：

```
{
    "WindowTaskId":"44444444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册任务（Amazon CLI）。

示例 5：使用维护时段目标 ID 注册任务

以下 `register-task-with-maintenance-window` 示例使用维护时段目标 ID 注册任务。维护时段目标 ID 位于 `aws ssm register-target-with-maintenance-window` 命令的输出中。您也可以从 `aws ssm describe-maintenance-window-targets` 命令输出中进行检索。

```
aws ssm register-task-with-maintenance-window \
    --targets "Key=WindowTargetIds,Values=350d44e6-28cc-44e2-951f-4b2c9EXAMPLE" \
    --task-arn "AWS-RunShellScript" \
```

```
    --service-role-arn "arn:aws:iam::111222333444:role/MaintenanceWindowsRole" \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --task-type "RUN_COMMAND" \
    --task-parameters  "{\"commands\":{\"Values\":[\"df\"]}}" \
    --max-concurrency 1 \
    --max-errors 1 \
    --priority 10
```

输出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的向维护时段注册任务（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 RegisterTaskWithMaintenanceWindow。

remove-tags-from-resource

以下代码示例演示了如何使用 remove-tags-from-resource。

Amazon CLI

从补丁基准删除标签

以下 remove-tags-from-resource 示例将从补丁基准中删除标签。

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0" \
    --tag-keys "Region"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon 一般参考》中的标记 Amazon 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveTagsFromResource。

reset-service-setting

以下代码示例演示了如何使用 reset-service-setting。

Amazon CLI

重置 Parameter Store 吞吐量的服务设置

以下 reset-service-setting 示例重置指定区域中 Parameter Store 吞吐量的服务设置，不再使用提高的吞吐量。

```
aws ssm reset-service-setting \
    --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-store/high-throughput-enabled
```

输出：

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/parameter-store/high-throughput-enabled",
        "SettingValue": "false",
        "LastModifiedDate": 1555532818.578,
        "LastModifiedUser": "System",
        "ARN": "arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-store/high-throughput-enabled",
        "Status": "Default"
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的提高 Parameter Store 吞吐量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetServiceSetting。

resume-session

以下代码示例演示了如何使用 resume-session。

Amazon CLI

恢复 Session Manager 会话

此 resume-session 示例在实例断开连接后恢复与该实例的 Session Manager 会话。请注意，此交互式命令要求在进行调用的客户端计算机上安装会话管理器插件。

```
aws ssm resume-session \
    --session-id Mary-Major-07a16060613c408b5
```

输出：

```
{
    "SessionId": "Mary-Major-07a16060613c408b5",
    "TokenValue":
 "AAEAAVbTGsaOnyvcUoNGqifbv5r/8lgxuQljCuY8qVcvOnoBAAAAAFxtd3jIXAFUUXGTJ7zF/
AWJPwDviOlF5p3dlAgrqVIVO6IEXhkHLz0/1gXKRKEME71E6TLOplLDJAMZ
+kREejkZu4c5AxMkrQjMF+gtHP1bYJKTwtHQd1wjulPLexO8SHl7g5R/
wekrj6WsDUpnEegFBfGftpAIz2GXQVfTJXKfkc5qepQ11C11DOIT2dozOqXgHwfQHfAKLErM5dWDZqKwyT1Z3iw7unQd
+ihfGa6MEJJ97Jmat/a2TspEnOjNn9Mvu5iwXIW2yCvWZrGUj+/
QI5Xr7s1XJBEnSKR54o4fN0GV9RWl0RZsZm1m1ki0JJtiwwgZ",
    "StreamUrl": "wss://ssmmessages.us-east-2.amazonaws.com/v1/data-channel/Mary-
Major-07a16060613c408b5?role=publish_subscribe"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的为 Amazon CLI 安装 Session Manager 插件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResumeSession。

send-automation-signal

以下代码示例演示了如何使用 send-automation-signal。

Amazon CLI

向自动化执行系统发送信号

以下 send-automation-signal 示例向自动化执行系统发送批准信号。

```
aws ssm send-automation-signal \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
    --signal-type "Approve"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的由审批人运行自动化工作流。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 SendAutomationSignal。

send-command

以下代码示例演示了如何使用 send-command。

Amazon CLI

示例 1：在一个或多个远程实例上运行命令

以下 send-command 示例在目标实例上运行 echo 命令。

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --parameters 'commands=["echo HelloWorld"]' \
    --targets "Key=instanceids,Values=i-1234567890abcdef0" \
    --comment "echo HelloWorld"
```

输出：

```
{
    "Command": {
        "CommandId": "92853adf-ba41-4cd6-9a88-142d1EXAMPLE",
        "DocumentName": "AWS-RunShellScript",
        "DocumentVersion": "",
        "Comment": "echo HelloWorld",
        "ExpiresAfter": 1550181014.717,
        "Parameters": {
            "commands": [
                "echo HelloWorld"
            ]
        },
        "InstanceIds": [
            "i-0f00f008a2dcbefe2"
        ],
        "Targets": [],
        "RequestedDateTime": 1550173814.717,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "OutputS3BucketName": "",
        "OutputS3KeyPrefix": "",
```

```
        "MaxConcurrency": "50",
        "MaxErrors": "0",
        "TargetCount": 1,
        "CompletedCount": 0,
        "ErrorCount": 0,
        "DeliveryTimedOutCount": 0,
        "ServiceRole": "",
        "NotificationConfig": {
            "NotificationArn": "",
            "NotificationEvents": [],
            "NotificationType": ""
        },
        "CloudWatchOutputConfig": {
            "CloudWatchLogGroupName": "",
            "CloudWatchOutputEnabled": false
        }
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 2：获取有关实例的 IP 信息

以下 send-command 示例检索关于实例的 IP 信息。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 3：在具有特定标签的实例上运行命令

以下 send-command 示例在标签键为“ENV”且值为“Dev”的实例上运行命令。

```
aws ssm send-command \
    --targets "Key=tag:ENV,Values=Dev" \
```

```
    --document-name "AWS-RunShellScript" \
    --parameters "commands=ifconfig"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 4：运行发送 SNS 通知的命令

以下 send-command 示例运行一条命令，发送所有通知事件和 Command 通知类型的 SNS 通知。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig" \
    --service-role-arn "arn:aws:iam::123456789012:role/SNS_Role" \
    --notification-config "NotificationArn=arn:aws:sns:us-
east-1:123456789012:SNSTopicName,NotificationEvents=All,NotificationType=Command"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 5：运行输出到 S3 和 CloudWatch 的命令

以下 send-command 示例运行一条命令，将命令详细信息输出到 S3 存储桶和 CloudWatch Logs 日志组。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig" \
    --output-s3-bucket-name "s3-bucket-name" \
    --output-s3-key-prefix "runcommand" \
    --cloud-watch-output-
config "CloudWatchOutputEnabled=true,CloudWatchLogGroupName=CWLGroupName"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 6：在具有不同标签的多个实例上运行命令

以下 send-command 示例对具有两个不同标签键和值的实例运行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev Key=tag:Role,Values=WebServers
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 7：将具有相同标签键的多个实例设为目标

以下 send-command 示例在具有相同标签键但不同值的实例上运行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev,Test
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 8：运行使用共享文档的命令

以下 send-command 示例在目标实例上运行共享文档。

```
aws ssm send-command \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument" \
    --targets "Key=instanceids,Values=i-1234567890abcdef0"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用共享 SSM 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 SendCommand。

start-associations-once

以下代码示例演示了如何使用 start-associations-once。

Amazon CLI

立即运行关联，且只运行一次

以下 start-associations-once 示例立即运行指定的关联，且只运行一次。如果此命令成功，则无任何输出。

```
aws ssm start-associations-once \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的查看关联历史记录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartAssociationsOnce。

start-automation-execution

以下代码示例演示了如何使用 start-automation-execution。

Amazon CLI

示例 1：执行自动化文档

以下 start-automation-execution 示例运行自动化文档。

```
aws ssm start-automation-execution \
    --document-name "AWS-UpdateLinuxAmi" \
    --parameters "AutomationAssumeRole=arn:aws:iam::123456789012:role/
SSMAutomationRole,SourceAmiId=ami-EXAMPLE,IamInstanceProfileName=EC2InstanceRole"
```

输出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的手动运行自动化工作流。

示例 2：运行共享自动化文档

以下 start-automation-execution 示例运行一个共享的自动化文档。

```
aws ssm start-automation-execution \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
```

输出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用共享 SSM 文档。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StartAutomationExecution。

start-change-request-execution

以下代码示例演示了如何使用 start-change-request-execution。

Amazon CLI

示例 1：启动变更请求

以下 start-change-request-execution 示例使用最少的指定选项启动变更请求。

```
aws ssm start-change-request-execution \
    --change-request-name MyChangeRequest \
    --document-name AWS-HelloWorldChangeTemplate \
    --runbooks '[{"DocumentName": "AWS-HelloWorld","Parameters":
 {"AutomationAssumeRole": ["arn:aws:iam:us-east-2:1112223233444:role/
MyChangeManagerAssumeRole"]}}]' \
    --parameters
 Approver="JohnDoe",ApproverType="IamUser",ApproverSnsTopicArn="arn:aws:sns:us-
east-2:1112223233444:MyNotificationTopic"
```

输出：

```
{
  "AutomationExecutionId": "9d32a4fc-f944-11e6-4105-0a1b2EXAMPLE"
```

```
}
```

示例 2：使用外部 JSON 文件启动变更请求

以下 `start-automation-execution` 示例使用 JSON 文件中指定的多个选项启动变更请求。

```
aws ssm start-change-request-execution \
    --cli-input-json file://MyChangeRequest.json
```

的内容MyChangeRequest.json：

```
{
    "ChangeRequestName": "MyChangeRequest",
    "DocumentName": "AWS-HelloWorldChangeTemplate",
    "DocumentVersion": "$DEFAULT",
    "ScheduledTime": "2021-12-30T03:00:00",
    "ScheduledEndTime": "2021-12-30T03:05:00",
    "Tags": [
        {
            "Key": "Purpose",
            "Value": "Testing"
        }
    ],
    "Parameters": {
        "Approver": [
            "JohnDoe"
        ],
        "ApproverType": [
            "IamUser"
        ],
        "ApproverSnsTopicArn": [
            "arn:aws:sns:us-east-2:111222333444;:MyNotificationTopic
        ]
    },
    "Runbooks": [
        {
            "DocumentName": "AWS-HelloWorld",
            "DocumentVersion": "1",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Parameters": {
                "AutomationAssumeRole": [
                    "arn:aws:iam::111222333444:role/MyChangeManagerAssumeRole"
```

```
                ]
            }
        }
    ],
    "ChangeDetails": "### Document Name: HelloWorldChangeTemplate\n\n## What does
 this document do?\nThis change template demonstrates the feature set available
 for creating change templates for Change Manager. This template starts a Runbook
 workflow for the Automation document called AWS-HelloWorld.\n\n## Input Parameters
\n* ApproverSnsTopicArn: (Required) Amazon Simple Notification Service ARN for
 approvers.\n* Approver: (Required) The name of the approver to send this request
 to.\n* ApproverType: (Required) The type of reviewer.\n  * Allowed Values: IamUser,
 IamGroup, IamRole, SSOGroup, SSOUser\n\n## Output Parameters\nThis document has no
 outputs \n"
}
```

输出：

```
{
  "AutomationExecutionId": "9d32a4fc-f944-11e6-4105-0a1b2EXAMPLE"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建变更请求。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartChangeRequestExecution。

start-session

以下代码示例演示了如何使用 start-session。

Amazon CLI

示例 1：启动会话管理器会话

此 start-session 示例为会话管理器会话建立与实例的连接。请注意，此交互式命令要求在进行调用的客户端计算机上安装会话管理器插件。

```
aws ssm start-session \
    --target "i-1234567890abcdef0"
```

输出：

```
Starting session with SessionId: Jane-Roe-07a16060613c408b5
```

示例 2：使用 SSH 启动会话管理器会话

此 start-session 示例为使用 SSH 的会话管理器会话建立与实例的连接。请注意，此交互式命令要求在进行调用的客户端计算机上安装会话管理器插件，并且该命令在实例上使用默认用户，例如为 Linux 的 EC2 实例使用 ec2-user。

```
ssh -i /path/my-key-pair.pem ec2-user@i-02573cafcfEXAMPLE
```

输出：

```
Starting session with SessionId: ec2-user-07a16060613c408b5
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的启动会话和为 Amazon CLI 安装会话管理器插件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartSession。

stop-automation-execution

以下代码示例演示了如何使用 stop-automation-execution。

Amazon CLI

停止自动化执行

以下 stop-automation-execution 示例停止自动化文档。

```
aws ssm stop-automation-execution
    --automation-execution-id "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的手动运行自动化工作流。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 StopAutomationExecution。

terminate-session

以下代码示例演示了如何使用 terminate-session。

Amazon CLI

结束 Session Manager 会话

此 terminate-session 示例永久结束用户“Shirley-Rodriguez”创建的会话，并关闭 Session Manager 客户端与实例上的 SSM Agent 之间的数据连接。

```
aws ssm terminate-session \
    --session-id "Shirley-Rodriguez-07a16060613c408b5"
```

输出：

```
{
    "SessionId": "Shirley-Rodriguez-07a16060613c408b5"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的终止会话。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateSession。

unlabel-parameter-version

以下代码示例演示了如何使用 unlabel-parameter-version。

Amazon CLI

删除参数标签

以下 unlabel-parameter-version 示例从给定版本的参数中删除指定的标签。

```
aws ssm unlabel-parameter-version \
    --name "parameterName" \
    --parameter-version "version" \
    --labels "label_1" "label_2" "label_3"
```

输出：

```
{
    "RemovedLabels": [
        "label_1"
        "label_2"
        "label_3"
```

```
    ],
    "InvalidLabels": []
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的删除参数标签（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UnlabelParameterVersion。

update-association-status

以下代码示例演示了如何使用 update-association-status。

Amazon CLI

更新关联状态

以下 update-association-status 示例更新了实例和文档之间关联的关联状态。

```
aws ssm update-association-status \
    --name "AWS-UpdateSSMAgent" \
    --instance-id "i-1234567890abcdef0" \
    --association-
status "Date=1424421071.939,Name=Pending,Message=temp_status_change,AdditionalInfo=Additiona
Config-Needed"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-1234567890abcdef0",
        "AssociationVersion": "1",
        "Date": 1550507529.604,
        "LastUpdateAssociationDate": 1550507806.974,
        "Status": {
            "Date": 1424421071.0,
            "Name": "Pending",
            "Message": "temp_status_change",
            "AdditionalInfo": "Additional-Config-Needed"
        },
        "Overview": {
            "Status": "Success",
```

```
            "AssociationStatusAggregatedCount": {
                "Success": 1
            }
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-1234567890abcdef0"
                ]
            }
        ],
        "LastExecutionDate": 1550507808.0,
        "LastSuccessfulExecutionDate": 1550507808.0
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateAssociationStatus。

update-association

以下代码示例演示了如何使用 update-association。

Amazon CLI

示例 1：更新文档关联

以下 update-association 示例使用新文档版本更新关联。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --document-version "\$LATEST"
```

输出：

```
{
    "AssociationDescription": {
```

```
        "Name": "AWS-UpdateSSMAgent",
        "AssociationVersion": "2",
        "Date": 1550508093.293,
        "LastUpdateAssociationDate": 1550508106.596,
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$LATEST",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "tag:Name",
                "Values": [
                    "Linux"
                ]
            }
        ],
        "LastExecutionDate": 1550508094.879,
        "LastSuccessfulExecutionDate": 1550508094.879
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：更新关联的计划表达式

以下 update-association 示例更新了指定关联的计划表达式。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --schedule-expression "cron(0 0 0/4 1/1 * ? *)"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-HelloWorld",
        "AssociationVersion": "2",
        "Date": "2021-02-08T13:54:19.203000-08:00",
        "LastUpdateAssociationDate": "2021-06-29T11:51:07.933000-07:00",
        "Overview": {
            "Status": "Pending",
```

```
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "aws:NoOpAutomationTag",
                "Values": [
                    "AWS-NoOpAutomationTarget-Value"
                ]
            }
        ],
        "ScheduleExpression": "cron(0 0 0/4 1/1 * ? *)",
        "LastExecutionDate": "2021-06-26T19:00:48.110000-07:00",
        "ApplyOnlyAtCronInterval": false
    }
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编辑和创建关联的新版本。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateAssociation。

update-document-default-version

以下代码示例演示了如何使用 update-document-default-version。

Amazon CLI

更新文档的默认版本

以下 update-document-default-version 示例更新了 Systems Manager 文档的默认版本。

```
aws ssm update-document-default-version \
    --name "Example" \
    --document-version "2"
```

输出：

```
{
    "Description": {
        "Name": "Example",
        "DefaultVersion": "2"
    }
```

```
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的编写 SSM 文档内容。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateDocumentDefaultVersion。

update-document-metadata

以下代码示例演示了如何使用 update-document-metadata。

Amazon CLI

示例：批准最新版本的变更模板

以下 update-document-metadata 将提供对已提交审核的最新版本变更模板的批准情况。

```
aws ssm update-document-metadata \
    --name MyChangeManagerTemplate \
    --document-reviews 'Action=Approve,Comment=[{Type=Comment,Content=Approved!}]'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的审查、批准或拒绝变更模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDocumentMetadata。

update-document

以下代码示例演示了如何使用 update-document。

Amazon CLI

创建文档的新版本

以下 update-document 示例在 Windows 计算机上运行时创建文档的新版本。--document 指定的文档必须采用 JSON 格式。请注意，必须先引用 file://，后跟内容文件的路径。由于 --document-version 参数的开头有 $，因此在 Windows 上，必须用双引号将该值括起来。在 Linux、MacOS 或 PowerShell 提示符下，必须用单引号将该值括起来。

Windows 版本：

```
aws ssm update-document \
```

```
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version "$LATEST"
```

Linux/Mac 版本：

```
aws ssm update-document \
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version '$LATEST'
```

输出：

```
{
  "DocumentDescription": {
      "Status": "Updating",
      "Hash": "f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b",
      "Name": "RunShellScript",
      "Parameters": [
          {
              "Type": "StringList",
              "Name": "commands",
              "Description": "(Required) Specify a shell script or a command to
 run."
          }
      ],
      "DocumentType": "Command",
      "PlatformTypes": [
          "Linux"
      ],
      "DocumentVersion": "2",
      "HashType": "Sha256",
      "CreatedDate": 1487899655.152,
      "Owner": "809632081692",
      "SchemaVersion": "2.0",
      "DefaultVersion": "1",
      "LatestVersion": "2",
      "Description": "Run an updated script"
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDocument。

update-maintenance-window-target

以下代码示例演示了如何使用 update-maintenance-window-target。

Amazon CLI

更新维护时段目标

以下 update-maintenance-window-target 示例仅更新维护时段目标的名称。

```
aws ssm update-maintenance-window-target \
    --window-id "mw-0c5ed765acEXAMPLE" \
    --window-target-id "57e8344e-fe64-4023-8191-6bf05EXAMPLE" \
    --name "NewName" \
    --no-replace
```

输出：

```
{
    "Description": "",
    "OwnerInformation": "",
    "WindowTargetId": "57e8344e-fe64-4023-8191-6bf05EXAMPLE",
    "WindowId": "mw-0c5ed765acEXAMPLE",
    "Targets": [
        {
            "Values": [
                "i-1234567890EXAMPLE"
            ],
            "Key": "InstanceIds"
        }
    ],
    "Name": "NewName"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新维护时段（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMaintenanceWindowTarget。

update-maintenance-window-task

以下代码示例演示了如何使用 update-maintenance-window-task。

Amazon CLI

更新维护时段任务

以下 update-maintenance-window-task 示例更新维护时段任务的服务角色。

```
aws ssm update-maintenance-window-task \
    --window-id "mw-0c5ed765acEXAMPLE" \
    --window-task-id "23d3809e-9fbe-4ddf-b41a-b49d7EXAMPLE" \
    --service-role-arn "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM"
```

输出：

```
{
    "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "MaxErrors": "1",
    "TaskArn": "AWS-UpdateEC2Config",
    "MaxConcurrency": "1",
    "WindowTaskId": "23d3809e-9fbe-4ddf-b41a-b49d7EXAMPLE",
    "TaskParameters": {},
    "Priority": 1,
    "TaskInvocationParameters": {
        "RunCommand": {
            "TimeoutSeconds": 600,
            "Parameters": {
                "allowDowngrade": [
                    "false"
                ]
            }
        }
    },
    "WindowId": "mw-0c5ed765acEXAMPLE",
    "Description": "UpdateEC2Config",
    "Targets": [
        {
            "Values": [
                "57e8344e-fe64-4023-8191-6bf05EXAMPLE"
            ],
            "Key": "WindowTargetIds"
        }
    ],
```

```
    "Name": "UpdateEC2Config"
}
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新维护时段（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMaintenanceWindowTask。

update-maintenance-window

以下代码示例演示了如何使用 update-maintenance-window。

Amazon CLI

示例 1：更新维护时段

以下 update-maintenance-window 示例更新了维护时段的名称。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --name "My-Renamed-MW"
```

输出：

```
{
    "Cutoff": 1,
    "Name": "My-Renamed-MW",
    "Schedule": "cron(0 16 ? * TUE *)",
    "Enabled": true,
    "AllowUnassociatedTargets": true,
    "WindowId": "mw-1a2b3c4d5e6f7g8h9",
    "Duration": 4
}
```

示例 2：禁用维护时段

以下 update-maintenance-window 示例禁用维护时段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
```

```
    --no-enabled
```

示例 3：启用维护时段

以下 update-maintenance-window 示例启用维护时段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --enabled
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新维护时段（Amazon CLI）。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateMaintenanceWindow。

update-managed-instance-role

以下代码示例演示了如何使用 update-managed-instance-role。

Amazon CLI

更新托管式实例的 IAM 角色

以下 update-managed-instance-role 示例更新了托管式实例的 IAM 实例配置文件。

```
aws ssm update-managed-instance-role \
    --instance-id "mi-08ab247cdfEXAMPLE" \
    --iam-role "ExampleRole"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的步骤 4：为 Systems Manager 创建 IAM 实例配置文件。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdateManagedInstanceRole。

update-ops-item

以下代码示例演示了如何使用 update-ops-item。

Amazon CLI

更新 OpsItem

以下 update-ops-item 示例更新 OpsItem 的描述、优先级和类别。此外，该命令还指定一个 SNS 主题，即，当编辑或更改此 OpsItem 时，将发送通知。

```
aws ssm update-ops-item \
    --ops-item-id "oi-287b5EXAMPLE" \
    --description "Primary OpsItem for failover event 2020-01-01-fh398yf" \
    --priority 2 \
    --category "Security" \
    --notifications "Arn=arn:aws:sns:us-east-2:111222333444:my-us-east-2-topic"
```

输出：

```
This command produces no output.
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOpsItem。

update-patch-baseline

以下代码示例演示了如何使用 update-patch-baseline。

Amazon CLI

示例 1：更新补丁基准

以下 update-patch-baseline 示例将指定的两个补丁（作为已拒绝的补丁）和一个补丁（作为已批准的补丁）添加到指定的补丁基准。

```
aws ssm update-patch-baseline \
        --baseline-id "pb-0123456789abcdef0" \
        --rejected-patches "KB2032276" "MS10-048" \
        --approved-patches "KB2124261"
```

输出：

```
{
```

```
    "BaselineId": "pb-0123456789abcdef0",
    "Name": "WindowsPatching",
    "OperatingSystem": "WINDOWS",
    "GlobalFilters": {
        "PatchFilters": []
    },
    "ApprovalRules": {
        "PatchRules": [
            {
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "PRODUCT",
                            "Values": [
                                "WindowsServer2016"
                            ]
                        }
                    ]
                },
                "ComplianceLevel": "CRITICAL",
                "ApproveAfterDays": 0,
                "EnableNonSecurity": false
            }
        ]
    },
    "ApprovedPatches": [
        "KB2124261"
    ],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "RejectedPatches": [
        "KB2032276",
        "MS10-048"
    ],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "CreatedDate": 1550244180.465,
    "ModifiedDate": 1550244180.465,
    "Description": "Patches for Windows Servers",
    "Sources": []
}
```

示例 2：重命名补丁基准

以下 update-patch-baseline 示例重命名指定的补丁基准。

```
aws ssm update-patch-baseline \
    --baseline-id "pb-0713accee01234567" \
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的更新或删除补丁基准 <https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html>`__。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 UpdatePatchBaseline。

update-resource-data-sync

以下代码示例演示了如何使用 update-resource-data-sync。

Amazon CLI

更新资源数据同步

以下 update-resource-data-sync 示例更新 SyncFromSource 资源数据同步。

```
aws ssm update-resource-data-sync \
    --sync-name exampleSync \
    --sync-type SyncFromSource \
    --sync-source '{"SourceType":"SingleAccountMultiRegions", "SourceRegions":["us-east-1", "us-west-2"]}'
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的设置 Systems Manager Explorer 以显示来自多个账户和区域的数据。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResourceDataSync。

update-service-setting

以下代码示例演示了如何使用 update-service-setting。

Amazon CLI

更新 Parameter Store 吞吐量的服务设置

以下 update-service-setting 示例更新指定区域中 Parameter Store 吞吐量的当前服务设置，以使用提高的吞吐量。

```
aws ssm update-service-setting \
    --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-store/high-throughput-enabled \
    --setting-value true
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的提高 Parameter Store 吞吐量。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateServiceSetting。

使用 Amazon CLI 的 Amazon Textract 示例

以下代码示例演示如何通过将Amazon Command Line Interface 与 Amazon Textract 结合使用，以执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

analyze-document

以下代码示例演示了如何使用 analyze-document。

Amazon CLI

分析文档中的文本

以下 analyze-document 示例演示如何分析文档中的文本。

Linux/macOS：

```
aws textract analyze-document \
    --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
    --feature-types '["TABLES","FORMS"]'
```

Windows:

```
aws textract analyze-document \
    --document "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document\"}}" \
    --feature-types "[\"TABLES\",\"FORMS\"]" \
    --region region-name
```

输出：

```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
```

```
                ]
            },
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "87586964-d50d-43e2-ace5-8a890657b9a0",
                        "a1e72126-21d9-44f4-a8d6-5c385f9002ba",
                        "e889d012-8a6b-4d2e-b7cd-7a8b327d876a"
                    ]
                }
            ],
            "BlockType": "PAGE",
            "Id": "c2227f12-b25d-4e1f-baea-1ee180d926b2"
        }
    ],
    "DocumentMetadata": {
        "Pages": 1
    }
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“使用 Amazon Textract 分析文档文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AnalyzeDocument。

`detect-document-text`

以下代码示例演示了如何使用 `detect-document-text`。

Amazon CLI

检测文档中的文本

以下 `detect-document-text` 示例演示了如何检测文档中的文本。

Linux/macOS：

```
aws textract detect-document-text \
    --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}'
```

Windows:

```
aws textract detect-document-text \
    --document "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document\"}}" \
    --region region-name
```

输出：

```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
                ]
            },
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "896a9f10-9e70-4412-81ce-49ead73ed881",
                        "0da18623-dc4c-463d-a3d1-9ac050e9e720",
                        "167338d7-d38c-4760-91f1-79a8ec457bb2"
                    ]
```

```
                }
            ],
            "BlockType": "PAGE",
            "Id": "21f0535e-60d5-4bc7-adf2-c05dd851fa25"
        },
        {
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "62490c26-37ea-49fa-8034-7a9ff9369c9c",
                        "1e4f3f21-05bd-4da9-ba10-15d01e66604c"
                    ]
                }
            ],
            "Confidence": 89.11581420898438,
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33642634749412537,
                    "Top": 0.17169663310050964,
                    "Left": 0.13885067403316498,
                    "Height": 0.49159330129623413
                },
                "Polygon": [
                    {
                        "Y": 0.17169663310050964,
                        "X": 0.13885067403316498
                    },
                    {
                        "Y": 0.17169663310050964,
                        "X": 0.47527703642845154
                    },
                    {
                        "Y": 0.6632899641990662,
                        "X": 0.47527703642845154
                    },
                    {
                        "Y": 0.6632899641990662,
                        "X": 0.13885067403316498
                    }
                ]
            },
            "Text": "He llo,",
            "BlockType": "LINE",
```

```
            "Id": "896a9f10-9e70-4412-81ce-49ead73ed881"
        },
        {
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "19b28058-9516-4352-b929-64d7cef29daf"
                    ]
                }
            ],
            "Confidence": 85.5694351196289,
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33182239532470703,
                    "Top": 0.23131252825260162,
                    "Left": 0.5091826915740967,
                    "Height": 0.3766750991344452
                },
                "Polygon": [
                    {
                        "Y": 0.23131252825260162,
                        "X": 0.5091826915740967
                    },
                    {
                        "Y": 0.23131252825260162,
                        "X": 0.8410050868988037
                    },
                    {
                        "Y": 0.607987642288208,
                        "X": 0.8410050868988037
                    },
                    {
                        "Y": 0.607987642288208,
                        "X": 0.5091826915740967
                    }
                ]
            },
            "Text": "worlc",
            "BlockType": "LINE",
            "Id": "0da18623-dc4c-463d-a3d1-9ac050e9e720"
        }
    ],
    "DocumentMetadata": {
```

```
        "Pages": 1
    }
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“使用 Amazon Textract 检测文档文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetectDocumentText。

get-document-analysis

以下代码示例演示了如何使用 get-document-analysis。

Amazon CLI

获取对多页文档进行异步文本分析的结果

以下 get-document-analysis 示例演示了如何获取对多页文档进行异步文本分析的结果。

```
aws textract get-document-analysis \
    --job-id df7cf32ebbd2a5de113535fcf4d921926a701b09b4e7d089f3aebadb41e0712b \
    --max-results 1000
```

输出：

```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
```

```
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
                ]
            },
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "75966e64-81c2-4540-9649-d66ec341cd8f",
                        "bb099c24-8282-464c-a179-8a9fa0a057f0",
                        "5ebf522d-f9e4-4dc7-bfae-a288dc094595"
                    ]
                }
            ],
            "BlockType": "PAGE",
            "Id": "247c28ee-b63d-4aeb-9af0-5f7ea8ba109e",
            "Page": 1
        }
    ],
    "NextToken": "cY1W3eTFvoB0cH7YrKVudI4Gb0H8J0xAYLo8xI/JunCIPWCthaKQ+07n/
ElyutsSy0+1VOImoTRmP1zw4P0RFtaeV9Bzhnfedpx1YqwB4xaGDA==",
    "DocumentMetadata": {
        "Pages": 1
    },
    "JobStatus": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“检测和分析多页文档中的文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocumentAnalysis。

get-document-text-detection

以下代码示例演示了如何使用 get-document-text-detection。

Amazon CLI

获取对多页文档进行异步文本检测的结果

以下 get-document-text-detection 示例演示了如何获取对多页文档进行异步文本检测的结果。

```
aws textract get-document-text-detection \
    --job-id 57849a3dc627d4df74123dca269d69f7b89329c870c65bb16c9fd63409d200b9 \
    --max-results 1000
```

Output

```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
                ]
            },
            "Relationships": [
```

```
                {
                    "Type": "CHILD",
                    "Ids": [
                        "1b926a34-0357-407b-ac8f-ec473160c6a9",
                        "0c35dc17-3605-4c9d-af1a-d9451059df51",
                        "dea3db8a-52c2-41c0-b50c-81f66f4aa758"
                    ]
                }
            ],
            "BlockType": "PAGE",
            "Id": "84671a5e-8c99-43be-a9d1-6838965da33e",
            "Page": 1
        }
    ],
    "NextToken": "GcqyoAJuZwujOT35EN4LCI3EUzMtiLq3nKyFFHvU5q1SaIdEBcSty+njNgoWwuMP/
muqc96S4o5NzDqehhXvhkodMyVO5OJGyms5lsrCxibWJw==",
    "DocumentMetadata": {
        "Pages": 1
    },
    "JobStatus": "SUCCEEDED"
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“检测和分析多页文档中的文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocumentTextDetection。

start-document-analysis

以下代码示例演示了如何使用 start-document-analysis。

Amazon CLI

开始分析多页文档中的文本

以下 start-document-analysis 示例演示如何开始异步分析多页文档中的文本。

Linux/macOS：

```
aws textract start-document-analysis \
    --document-location '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
    --feature-types '["TABLES","FORMS"]' \
    --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"
```

Windows:

```
aws textract start-document-analysis \
    --document-location "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document
\"}}" \
    --feature-types "[\"TABLES\", \"FORMS\"]" \
    --region region-name \
    --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"
```

输出：

```
{
    "JobId": "df7cf32ebbd2a5de113535fcf4d921926a701b09b4e7d089f3aebadb41e0712b"
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“检测和分析多页文档中的文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDocumentAnalysis。

start-document-text-detection

以下代码示例演示了如何使用 start-document-text-detection。

Amazon CLI

开始检测多页文档中的文本

以下 start-document-text-detection 示例演示如何开始异步检测多页文档中的文本。

Linux/macOS：

```
aws textract start-document-text-detection \
        --document-location '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
        --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleARN"
```

Windows:

```
aws textract start-document-text-detection \
    --document-location "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document
\"}}" \
    --region region-name \
    --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"
```

输出：

```
{
    "JobId": "57849a3dc627d4df74123dca269d69f7b89329c870c65bb16c9fd63409d200b9"
}
```

有关更多信息，请参阅《Amazon Textract 开发人员指南》中的“检测和分析多页文档中的文本”

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartDocumentTextDetection。

使用 Amazon CLI 的 Amazon Transcribe 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Transcribe 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-language-model`

以下代码示例演示了如何使用 `create-language-model`。

Amazon CLI

示例 1：使用训练和调整数据创建自定义语言模型。

以下 `create-language-model` 示例创建一个自定义语言模型。您可以使用自定义语言模型来提高法律、酒店、金融和保险等领域的转录性能。对于语言代码，请输入有效的语言代码。在 base-model-name 中，指定最适合您要使用自定义语言模型转录的音频采样率的基本模型。在 model-name 中，指定要调用自定义语言模型的名称。

```
aws transcribe create-language-model \
```

```
    --language-code language-code \
    --base-model-name base-model-name \
    --model-name cli-clm-example \
    --input-data-config S3Uri="s3://amzn-s3-demo-bucket/Amazon-S3-Prefix-for-
training-data",TuningDataS3Uri="s3://amzn-s3-demo-bucket/Amazon-S3-Prefix-for-
tuning-data",DataAccessRoleArn="arn:aws:iam::AWS-account-number:role/IAM-role-with-
permissions-to-create-a-custom-language-model"
```

输出：

```
{
    "LanguageCode": "language-code",
    "BaseModelName": "base-model-name",
    "ModelName": "cli-clm-example",
    "InputDataConfig": {
        "S3Uri": "s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/",
        "TuningDataS3Uri": "s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/",
        "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-with-
permissions-create-a-custom-language-model"
    },
    "ModelStatus": "IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

示例 2：仅使用训练数据创建自定义语言模型。

以下 `create-language-model` 示例转录音频文件。您可以使用自定义语言模型来提高法律、酒店、金融和保险等领域的转录性能。对于语言代码，请输入有效的语言代码。在 base-model-name 中，指定最适合您要使用自定义语言模型转录的音频采样率的基本模型。在 model-name 中，指定要调用自定义语言模型的名称。

```
aws transcribe create-language-model \
    --language-code en-US \
    --base-model-name base-model-name \
    --model-name cli-clm-example \
    --input-data-config S3Uri="s3://amzn-s3-demo-bucket/Amazon-S3-Prefix-For-
Training-Data",DataAccessRoleArn="arn:aws:iam::AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
```

输出：

```
{
    "LanguageCode": "en-US",
    "BaseModelName": "base-model-name",
    "ModelName": "cli-clm-example",
    "InputDataConfig": {
        "S3Uri": "s3://amzn-s3-demo-bucket/Amazon-S3-Prefix-For-Training-Data/",
        "DataAccessRoleArn": "arn:aws:iam::your-AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
    },
    "ModelStatus": "IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLanguageModel。

`create-medical-vocabulary`

以下代码示例演示了如何使用 `create-medical-vocabulary`。

Amazon CLI

创建医疗领域自定义词汇表

以下 `create-medical-vocabulary` 示例创建一个自定义词汇表。要创建自定义词汇表，您必须创建一个文本文件，其中包含要更准确地进行转录的所有术语。对于 vocabulary-file-uri，指定该文本文件的 Amazon Simple Storage Service (Amazon S3) URI。对于 language-code，指定与自定义词汇表的语言对应的语言代码。对于 vocabulary-name，指定所需的自定义词汇表名称。

```
aws transcribe create-medical-vocabulary \
    --vocabulary-name cli-medical-vocab-example \
    --language-code language-code \
    --vocabulary-file-uri https://amzn-s3-demo-bucket.AWS-Region.amazonaws.com/the-
text-file-for-the-medical-custom-vocabulary.txt
```

输出：

```
{
    "VocabularyName": "cli-medical-vocab-example",
    "LanguageCode": "language-code",
    "VocabularyState": "PENDING"
```

```
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateMedicalVocabulary。

create-vocabulary-filter

以下代码示例演示了如何使用 create-vocabulary-filter。

Amazon CLI

创建词汇表过滤器

以下 create-vocabulary-filter 示例创建一个词汇表过滤器，该过滤器使用一个其中包含您不想在转录中出现的单词列表的文本文件。对于 language-code，请指定与词汇表过滤器语言对应的语言代码。对于 vocabulary-file-uri，请指定该文本文件的 Amazon Simple Storage Service（Amazon S3）URI。对于 vocabulary-filter-name，请指定词汇表过滤器的名称。

```
aws transcribe create-vocabulary-filter \
    --language-code language-code \
    --vocabulary-filter-file-uri s3://amzn-s3-demo-bucket/vocabulary-filter.txt \
    --vocabulary-filter-name cli-vocabulary-filter-example
```

输出：

```
{
    "VocabularyFilterName": "cli-vocabulary-filter-example",
    "LanguageCode": "language-code"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤不需要的字词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVocabularyFilter。

create-vocabulary

以下代码示例演示了如何使用 create-vocabulary。

Amazon CLI

创建自定义词汇表

以下 create-vocabulary 示例创建一个自定义词汇表。要创建自定义词汇表，您必须创建一个文本文件，其中包含要更准确地进行转录的所有术语。对于 vocabulary-file-uri，指定该文本文件的 Amazon Simple Storage Service (Amazon S3) URI。对于 language-code，指定与自定义词汇表的语言对应的语言代码。对于 vocabulary-name，指定所需的自定义词汇表名称。

```
aws transcribe create-vocabulary \
    --language-code language-code \
    --vocabulary-name cli-vocab-example \
    --vocabulary-file-uri s3://amzn-s3-demo-bucket/Amazon-S3-prefix/the-text-file-for-the-custom-vocabulary.txt
```

输出：

```
{
    "VocabularyName": "cli-vocab-example",
    "LanguageCode": "language-code",
    "VocabularyState": "PENDING"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateVocabulary。

delete-language-model

以下代码示例演示了如何使用 delete-language-model。

Amazon CLI

删除自定义语言模型

以下 delete-language-model 示例删除自定义语言模型。

```
aws transcribe delete-language-model \
    --model-name model-name
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLanguageModel。

delete-medical-transcription-job

以下代码示例演示了如何使用 delete-medical-transcription-job。

Amazon CLI

删除医疗领域转录作业

以下 delete-medical-transcription-job 示例删除一个医疗领域转录作业。

```
aws transcribe delete-medical-transcription-job \
    --medical-transcription-job-name medical-transcription-job-name
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的 DeleteMedicalTranscriptionJob。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMedicalTranscriptionJob。

delete-medical-vocabulary

以下代码示例演示了如何使用 delete-medical-vocabulary。

Amazon CLI

删除医疗领域自定义词汇表

以下 delete-medical-vocabulary 示例删除一个医疗领域自定义词汇表。对于 vocabulary-name，请指定医疗领域自定义词汇表的名称。

```
aws transcribe delete-vocabulary \
    --vocabulary-name medical-custom-vocabulary-name
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMedicalVocabulary。

delete-transcription-job

以下代码示例演示了如何使用 delete-transcription-job。

Amazon CLI

删除转录作业

以下 delete-transcription-job 示例删除一个转录作业。

```
aws transcribe delete-transcription-job \
    --transcription-job-name your-transcription-job
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的 DeleteTranscriptionJob。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTranscriptionJob。

delete-vocabulary-filter

以下代码示例演示了如何使用 delete-vocabulary-filter。

Amazon CLI

删除词汇表过滤器

以下 delete-vocabulary-filter 示例删除一个词汇表过滤器。

```
aws transcribe delete-vocabulary-filter \
    --vocabulary-filter-name vocabulary-filter-name
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤不需要的字词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVocabularyFilter。

delete-vocabulary

以下代码示例演示了如何使用 delete-vocabulary。

Amazon CLI

删除自定义词汇表

以下 delete-vocabulary 示例删除一个自定义词汇表。

```
aws transcribe delete-vocabulary \
    --vocabulary-name vocabulary-name
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteVocabulary。

describe-language-model

以下代码示例演示了如何使用 describe-language-model。

Amazon CLI

获取有关特定自定义语言模型的信息

以下 describe-language-model 示例获取有关特定自定义语言模型的信息。例如，在 BaseModelName 下，您可以看到您的模型是使用 NarrowBand 还是 WideBand 模型进行训练的。具有 NarrowBand 基础模型的自定义语言模型可以转录采样率低于 16 kHz 的音频。使用 WideBand 基础模型的语言模型可以转录采样率大于 16 kHz 的音频。S3Uri 参数表示您用于访问训练数据以创建自定义语言模型的 Amazon S3 前缀。

```
aws transcribe describe-language-model \
    --model-name cli-clm-example
```

输出：

```
{
    "LanguageModel": {
        "ModelName": "cli-clm-example",
        "CreateTime": "2020-09-25T17:57:38.504000+00:00",
        "LastModifiedTime": "2020-09-25T17:57:48.585000+00:00",
        "LanguageCode": "language-code",
        "BaseModelName": "base-model-name",
        "ModelStatus": "IN_PROGRESS",
```

```
        "UpgradeAvailability": false,
        "InputDataConfig": {
            "S3Uri": "s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/",
            "TuningDataS3Uri": "s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/",
            "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeLanguageModel。

get-medical-transcription-job

以下代码示例演示了如何使用 get-medical-transcription-job。

Amazon CLI

获取有关特定医疗领域转录作业的信息

以下 get-medical-transcription-job 示例获取有关医疗领域特定转录作业的信息。要访问转录结果，请使用 TranscriptFileUri 参数。如果您在转录作业中启用了其他功能，则可以在 Settings 对象中查看它们。Specialty 参数显示提供者的医学专科。Type 参数表示转录作业中的语音是医疗对话还是医疗笔录。

```
aws transcribe get-medical-transcription-job \
    --medical-transcription-job-name vocabulary-dictation-medical-transcription-job
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-
job",
        "TranscriptionJobStatus": "COMPLETED",
        "LanguageCode": "en-US",
        "MediaSampleRateHertz": 48000,
        "MediaFormat": "mp4",
        "Media": {
```

```
            "MediaFileUri": "s3://Amazon-S3-Prefix/your-audio-file.file-extension"
        },
        "Transcript": {
            "TranscriptFileUri": "https://s3.Region.amazonaws.com/Amazon-S3-Prefix/
vocabulary-dictation-medical-transcription-job.json"
        },
        "StartTime": "2020-09-21T21:17:27.045000+00:00",
        "CreationTime": "2020-09-21T21:17:27.016000+00:00",
        "CompletionTime": "2020-09-21T21:17:59.561000+00:00",
        "Settings": {
            "ChannelIdentification": false,
            "ShowAlternatives": false,
            "VocabularyName": "cli-medical-vocab-example"
        },
        "Specialty": "PRIMARYCARE",
        "Type": "DICTATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的批量转录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMedicalTranscriptionJob。

get-medical-vocabulary

以下代码示例演示了如何使用 get-medical-vocabulary。

Amazon CLI

获取有关医疗领域自定义词汇表的信息

以下 get-medical-vocabulary 示例获取有关医疗领域自定义词汇表的信息。您可以使用 VocabularyState 参数来查看词汇表的处理状态。如果状态为 READY，您就可以在 StartMedicalTranscriptionJob 操作中使用它。

```
aws transcribe get-medical-vocabulary \
    --vocabulary-name medical-vocab-example
```

输出：

```
{
    "VocabularyName": "medical-vocab-example",
```

```
    "LanguageCode": "en-US",
    "VocabularyState": "READY",
    "LastModifiedTime": "2020-09-19T23:59:04.349000+00:00",
    "DownloadUri": "https://link-to-download-the-text-file-used-to-create-your-
medical-custom-vocabulary"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMedicalVocabulary。

get-transcription-job

以下代码示例演示了如何使用 get-transcription-job。

Amazon CLI

获取有关特定转录作业的信息

以下 get-transcription-job 示例获取有关特定转录作业的信息。要访问转录结果，请使用 TranscriptFileUri 参数。可以使用 MediaFileUri 参数查看您使用该作业转录的音频文件。您可以使用 Settings 对象查看在转录作业中启用的可选功能。

```
aws transcribe get-transcription-job \
    --transcription-job-name your-transcription-job
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "your-transcription-job",
        "TranscriptionJobStatus": "COMPLETED",
        "LanguageCode": "language-code",
        "MediaSampleRateHertz": 48000,
        "MediaFormat": "mp4",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.file-
extension"
        },
        "Transcript": {
            "TranscriptFileUri": "https://Amazon-S3-file-location-of-transcription-
output"
```

```
        },
        "StartTime": "2020-09-18T22:27:23.970000+00:00",
        "CreationTime": "2020-09-18T22:27:23.948000+00:00",
        "CompletionTime": "2020-09-18T22:28:21.197000+00:00",
        "Settings": {
            "ChannelIdentification": false,
            "ShowAlternatives": false
        },
        "IdentifyLanguage": true,
        "IdentifiedLanguageScore": 0.8672199249267578
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的入门（Amazon 命令行界面）。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 GetTranscriptionJob。

get-vocabulary-filter

以下代码示例演示了如何使用 get-vocabulary-filter。

Amazon CLI

获取有关词汇表过滤器的信息

以下 get-vocabulary-filter 示例获取有关词汇表过滤器的信息。您可以使用 DownloadUri 参数来获取用于创建词汇过滤器的单词列表。

```
aws transcribe get-vocabulary-filter \
    --vocabulary-filter-name testFilter
```

输出：

```
{
    "VocabularyFilterName": "testFilter",
    "LanguageCode": "language-code",
    "LastModifiedTime": "2020-05-07T22:39:32.147000+00:00",
    "DownloadUri": "https://Amazon-S3-location-to-download-your-vocabulary-filter"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤不需要的字词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetVocabularyFilter。

get-vocabulary

以下代码示例演示了如何使用 get-vocabulary。

Amazon CLI

获取有关自定义词汇表的信息

以下 get-vocabulary 示例获取有关以前创建的自定义词汇表的信息。

```
aws transcribe get-vocabulary \
    --vocabulary-name cli-vocab-1
```

输出：

```
{
    "VocabularyName": "cli-vocab-1",
    "LanguageCode": "language-code",
    "VocabularyState": "READY",
    "LastModifiedTime": "2020-09-19T23:22:32.836000+00:00",
    "DownloadUri": "https://link-to-download-the-text-file-used-to-create-your-
custom-vocabulary"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自定义词汇表。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 GetVocabulary。

list-language-models

以下代码示例演示了如何使用 list-language-models。

Amazon CLI

列出自定义语言模型

以下 list-language-models 示例列出与您的 Amazon 账户和区域关联的自定义语言模型。您可以使用 S3Uri 和 TuningDataS3Uri 参数来查找用作训练数据或调整数据的 Amazon S3 前缀。BaseModelName 会告诉您是使用 NarrowBand 还是 WideBand 模型来创建自定义语言模型。您可以通过使用 NarrowBand 基础模型的自定义语言模型，以低于 16 kHz 的采样速率转录音频。您可以通过使用 WideBand 基础模型的自定义语言模型，转录 16 kHz 或更高采样速率

的音频。ModelStatus 参数显示您是否可以在转录作业中使用自定义语言模型。如果该值为 COMPLETED，表明您可以在转录作业中使用该模型。

```
aws transcribe list-language-models
```

输出：

```
{
    "Models": [
        {
            "ModelName": "cli-clm-2",
            "CreateTime": "2020-09-25T17:57:38.504000+00:00",
            "LastModifiedTime": "2020-09-25T17:57:48.585000+00:00",
            "LanguageCode": "language-code",
            "BaseModelName": "WideBand",
            "ModelStatus": "IN_PROGRESS",
            "UpgradeAvailability": false,
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/clm-training-data/",
                "TuningDataS3Uri": "s3://amzn-s3-demo-bucket/clm-tuning-data/",
                "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-
used-to-create-the-custom-language-model"
            }
        },
        {
            "ModelName": "cli-clm-1",
            "CreateTime": "2020-09-25T17:16:01.835000+00:00",
            "LastModifiedTime": "2020-09-25T17:16:15.555000+00:00",
            "LanguageCode": "language-code",
            "BaseModelName": "WideBand",
            "ModelStatus": "IN_PROGRESS",
            "UpgradeAvailability": false,
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/clm-training-data/",
                "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-
used-to-create-the-custom-language-model"
            }
        },
        {
            "ModelName": "clm-console-1",
            "CreateTime": "2020-09-24T19:26:28.076000+00:00",
            "LastModifiedTime": "2020-09-25T04:25:22.271000+00:00",
            "LanguageCode": "language-code",
```

```
            "BaseModelName": "NarrowBand",
            "ModelStatus": "COMPLETED",
            "UpgradeAvailability": false,
            "InputDataConfig": {
                "S3Uri": "s3://amzn-s3-demo-bucket/clm-training-data/",
                "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-
used-to-create-the-custom-language-model"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLanguageModels。

list-medical-transcription-jobs

以下代码示例演示了如何使用 list-medical-transcription-jobs。

Amazon CLI

列出医疗领域转录作业

以下 list-medical-transcription-jobs 示例列出与您的 Amazon 账户和区域关联的医疗领域转录作业。要获取有关特定转录作业的更多信息，请复制转录输出中 MedicalTranscriptionJobName 参数的值，并为 get-medical-transcription-job 命令的 MedicalTranscriptionJobName 选项指定该值。要查看更多转录作业，请复制 NextToken 参数值，再次运行 list-medical-transcription-jobs 命令，然后在 --next-token 选项中指定该值。

```
aws transcribe list-medical-transcription-jobs
```

输出：

```
{
    "NextToken": "3/PblzkiGhzjER3KHuQt2fmbPLF7cDYafjFMEoGn44ON/
gsuUSTIkGyanvRE6WMXFd/ZTEc2EZj+P9eii/
z1O2FDYli6RLI0WoRX4RwMisVrh9G0Kie0Y8ikBCdtqlZB10Wa9McC+ebOl
+LaDtZPC4u6ttoHLRlEfzqstHXSgapXg3tEBtm9piIaPB6MOM5BB6t86+qtmocTR/
qrteHZBBudhTfbCwhsxaqujHiiUvFdm3BQbKKWIW06yV9b+4f38oD2lVIan
```

```
+vfUs3gBYAl5VTDmXXzQPBQOHPjtwmFI+IWX15nSUjWuN3TUylHgPWzDaYT8qBtu0Z+3UG4V6b
+K2CC0XszXg5rBq9hYgNzy4XoFh/6s5DoSnzq49Q9xHgHdT2yBADFmvFK7myZBsj75+2vQZOSVpWUPy3WT/32zFAcoEL
+mFYfUjtTZ8n/jq7aQEjQ42A
+X/7K6JgOcdVPtEg8PlDr5kgYYG3q3OmYXX37U3FZuJmnTI63VtIXsNnOU5eGoYObtpk00Nq9UkzgSJxqj84ZD5n
+S0EGy9ZUYBJRRcGeYUM3Q4DbSJfUwSAqcFdLIWZdp8qIREMQIBWy7BLwSdyqsQo2vRrd53hm5aWM7SVf6pPq6X/
IXR5+1eUOOD8/coaTT4ES2DerbV6RkV4o0VT1d0SdVX/
MmtkNG8nYj8PqU07w7988quh1ZP6D80veJS1q73tUUR9MjnGernW2tAnvnLNhdefBcD
+sZVfYq3iBMFY7wTy1P1G6NqW9GrYDYoX3tTPWlD7phpbVSyKrh/
PdYrps5UxnsGoA1b7L/FfAXDfUoGrGUB4N3JsPYXX9D++g+6gV1qBBs/
WfF934aKqfD6UTggm/zV3GAOWiBpfvAZRvEb924i6yGHyMC7y54O1ZAwSBupmI
+FFd13CaPO4kN1vJlth6aM5vUPXg4BpyUhtbRhwD/KxCvf9K0tLJGyL1A==",
    "MedicalTranscriptionJobSummaries": [
        {
            "MedicalTranscriptionJobName": "vocabulary-dictation-medical-
transcription-job",
            "CreationTime": "2020-09-21T21:17:27.016000+00:00",
            "StartTime": "2020-09-21T21:17:27.045000+00:00",
            "CompletionTime": "2020-09-21T21:17:59.561000+00:00",
            "LanguageCode": "en-US",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "CUSTOMER_BUCKET",
            "Specialty": "PRIMARYCARE",
            "Type": "DICTATION"
        },
        {
            "MedicalTranscriptionJobName": "alternatives-dictation-medical-
transcription-job",
            "CreationTime": "2020-09-21T21:01:14.569000+00:00",
            "StartTime": "2020-09-21T21:01:14.592000+00:00",
            "CompletionTime": "2020-09-21T21:01:43.606000+00:00",
            "LanguageCode": "en-US",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "CUSTOMER_BUCKET",
            "Specialty": "PRIMARYCARE",
            "Type": "DICTATION"
        },
        {
            "MedicalTranscriptionJobName": "alternatives-conversation-medical-
transcription-job",
            "CreationTime": "2020-09-21T19:09:18.171000+00:00",
            "StartTime": "2020-09-21T19:09:18.199000+00:00",
            "CompletionTime": "2020-09-21T19:10:22.516000+00:00",
            "LanguageCode": "en-US",
            "TranscriptionJobStatus": "COMPLETED",
```

```
            "OutputLocationType": "CUSTOMER_BUCKET",
            "Specialty": "PRIMARYCARE",
            "Type": "CONVERSATION"
        },
        {
            "MedicalTranscriptionJobName": "speaker-id-conversation-medical-
transcription-job",
            "CreationTime": "2020-09-21T18:43:37.157000+00:00",
            "StartTime": "2020-09-21T18:43:37.265000+00:00",
            "CompletionTime": "2020-09-21T18:44:21.192000+00:00",
            "LanguageCode": "en-US",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "CUSTOMER_BUCKET",
            "Specialty": "PRIMARYCARE",
            "Type": "CONVERSATION"
        },
        {
            "MedicalTranscriptionJobName": "multichannel-conversation-medical-
transcription-job",
            "CreationTime": "2020-09-20T23:46:44.053000+00:00",
            "StartTime": "2020-09-20T23:46:44.081000+00:00",
            "CompletionTime": "2020-09-20T23:47:35.851000+00:00",
            "LanguageCode": "en-US",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "CUSTOMER_BUCKET",
            "Specialty": "PRIMARYCARE",
            "Type": "CONVERSATION"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的 https://docs.aws.amazon.com/transcribe/latest/dg/batch-med-transcription.html>。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 ListMedicalTranscriptionJobs。

`list-medical-vocabularies`

以下代码示例演示了如何使用 `list-medical-vocabularies`。

Amazon CLI

列出医疗领域自定义词汇表

以下 list-medical-vocabularies 示例列出与您的 Amazon 账户和区域关联的医疗领域自定义词汇表。要获取有关特定转录作业的更多信息，请复制转录输出中 MedicalTranscriptionJobName 参数的值，并为 get-medical-transcription-job 命令的 MedicalTranscriptionJobName 选项指定该值。要查看更多转录作业，请复制 NextToken 参数值，再次运行 list-medical-transcription-jobs 命令，然后在 --next-token 选项中指定该值。

```
aws transcribe list-medical-vocabularies
```

输出：

```
{
    "Vocabularies": [
        {
            "VocabularyName": "cli-medical-vocab-2",
            "LanguageCode": "en-US",
            "LastModifiedTime": "2020-09-21T21:44:59.521000+00:00",
            "VocabularyState": "READY"
        },
        {
            "VocabularyName": "cli-medical-vocab-1",
            "LanguageCode": "en-US",
            "LastModifiedTime": "2020-09-19T23:59:04.349000+00:00",
            "VocabularyState": "READY"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMedicalVocabularies。

list-transcription-jobs

以下代码示例演示了如何使用 list-transcription-jobs。

Amazon CLI

列出转录作业

以下 list-transcription-jobs 示例列出与您的 Amazon 账户和区域关联的转录作业。

```
aws transcribe list-transcription-jobs
```

输出：

```
{
    "NextToken": "NextToken",
    "TranscriptionJobSummaries": [
        {
            "TranscriptionJobName": "speak-id-job-1",
            "CreationTime": "2020-08-17T21:06:15.391000+00:00",
            "StartTime": "2020-08-17T21:06:15.416000+00:00",
            "CompletionTime": "2020-08-17T21:07:05.098000+00:00",
            "LanguageCode": "language-code",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "SERVICE_BUCKET"
        },
        {
            "TranscriptionJobName": "job-1",
            "CreationTime": "2020-08-17T20:50:24.207000+00:00",
            "StartTime": "2020-08-17T20:50:24.230000+00:00",
            "CompletionTime": "2020-08-17T20:52:18.737000+00:00",
            "LanguageCode": "language-code",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "SERVICE_BUCKET"
        },
        {
            "TranscriptionJobName": "sdk-test-job-4",
            "CreationTime": "2020-08-17T20:32:27.917000+00:00",
            "StartTime": "2020-08-17T20:32:27.956000+00:00",
            "CompletionTime": "2020-08-17T20:33:15.126000+00:00",
            "LanguageCode": "language-code",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "SERVICE_BUCKET"
        },
        {
            "TranscriptionJobName": "Diarization-speak-id",
            "CreationTime": "2020-08-10T22:10:09.066000+00:00",
            "StartTime": "2020-08-10T22:10:09.116000+00:00",
            "CompletionTime": "2020-08-10T22:26:48.172000+00:00",
            "LanguageCode": "language-code",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "SERVICE_BUCKET"
        },
```

```
        {
            "TranscriptionJobName": "your-transcription-job-name",
            "CreationTime": "2020-07-29T17:45:09.791000+00:00",
            "StartTime": "2020-07-29T17:45:09.826000+00:00",
            "CompletionTime": "2020-07-29T17:46:20.831000+00:00",
            "LanguageCode": "language-code",
            "TranscriptionJobStatus": "COMPLETED",
            "OutputLocationType": "SERVICE_BUCKET"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的入门（Amazon 命令行界面）。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 ListTranscriptionJobs。

`list-vocabularies`

以下代码示例演示了如何使用 `list-vocabularies`。

Amazon CLI

列出自定义词汇表

以下 `list-vocabularies` 示例列出与您的 Amazon 账户和区域关联的自定义词汇表。

```
aws transcribe list-vocabularies
```

输出：

```
{
    "NextToken": "NextToken",
    "Vocabularies": [
        {
            "VocabularyName": "ards-test-1",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-04-27T22:00:27.330000+00:00",
            "VocabularyState": "READY"
        },
        {
            "VocabularyName": "sample-test",
            "LanguageCode": "language-code",
```

```
            "LastModifiedTime": "2020-04-24T23:04:11.044000+00:00",
            "VocabularyState": "READY"
        },
        {
            "VocabularyName": "CRLF-to-LF-test-3-1",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-04-24T22:12:22.277000+00:00",
            "VocabularyState": "READY"
        },
        {
            "VocabularyName": "CRLF-to-LF-test-2",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-04-24T21:53:50.455000+00:00",
            "VocabularyState": "READY"
        },
        {
            "VocabularyName": "CRLF-to-LF-1-1",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-04-24T21:39:33.356000+00:00",
            "VocabularyState": "READY"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自定义词汇表。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 ListVocabularies。

`list-vocabulary-filters`

以下代码示例演示了如何使用 `list-vocabulary-filters`。

Amazon CLI

列出词汇表过滤器

以下 `list-vocabulary-filters` 示例列出与您的 Amazon 账户和区域关联的词汇表过滤器。

```
aws transcribe list-vocabulary-filters
```

输出：

```
{
```

```
    "NextToken": "NextToken": [
        {
            "VocabularyFilterName": "testFilter",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-05-07T22:39:32.147000+00:00"
        },
        {
            "VocabularyFilterName": "testFilter2",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-05-21T23:29:35.174000+00:00"
        },
        {
            "VocabularyFilterName": "filter2",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-05-08T20:18:26.426000+00:00"
        },
        {
            "VocabularyFilterName": "filter-review",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-06-03T18:52:30.448000+00:00"
        },
        {
            "VocabularyFilterName": "crlf-filt",
            "LanguageCode": "language-code",
            "LastModifiedTime": "2020-05-22T19:42:42.737000+00:00"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤不需要的字词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListVocabularyFilters。

start-medical-transcription-job

以下代码示例演示了如何使用 start-medical-transcription-job。

Amazon CLI

示例 1：转录存储为音频文件的医疗口述

以下 start-medical-transcription-job 示例转录一个音频文件。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
    --cli-input-json file://myfile.json
```

的内容myfile.json：

```
{
    "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
    }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-20T00:35:22.256000+00:00",
        "CreationTime": "2020-09-20T00:35:22.218000+00:00",
        "Specialty": "PRIMARYCARE",
        "Type": "DICTATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的批量转录概述。

示例 2：转录存储为音频文件的临床医生与患者之间的对话

以下 start-medical-transcription-job 示例转录包含有临床医生与患者之间对话的音频文件。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
```

```
    --cli-input-json file://mysecondfile.json
```

的内容mysecondfile.json：

```
{
    "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
    }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "simple-conversation-medical-transcription-
job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-20T23:19:49.965000+00:00",
        "CreationTime": "2020-09-20T23:19:49.941000+00:00",
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的批量转录概述。

示例 3：转录临床医生与患者之间对话的多声道音频文件

以下 start-medical-transcription-job 示例转录音频文件中每个声道的音频，并将每个声道的单独转录合并为一个转录输出。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
```

```
    --cli-input-json file://mythirdfile.json
```

的内容mythirdfile.json：

```
{
    "MedicalTranscriptionJobName": "multichannel-conversation-medical-transcription-
job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
        "Media": {
          "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "Settings":{
          "ChannelIdentification": true
        }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "multichannel-conversation-medical-
transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-20T23:46:44.081000+00:00",
        "CreationTime": "2020-09-20T23:46:44.053000+00:00",
        "Settings": {
            "ChannelIdentification": true
        },
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的声道识别。

示例 4：转录临床医生与患者之间对话的音频文件并在转录输出中识别发言者

以下 start-medical-transcription-job 示例转录一个音频文件，并在转录输出中标记每个发言者的语音。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
    --cli-input-json file://myfourthfile.json
```

的内容myfourthfile.json：

```
{
    "MedicalTranscriptionJobName": "speaker-id-conversation-medical-transcription-
job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
    "Settings":{
        "ShowSpeakerLabels": true,
        "MaxSpeakerLabels": 2
        }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "speaker-id-conversation-medical-
transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-21T18:43:37.265000+00:00",
        "CreationTime": "2020-09-21T18:43:37.157000+00:00",
        "Settings": {
            "ShowSpeakerLabels": true,
            "MaxSpeakerLabels": 2
        },
        "Specialty": "PRIMARYCARE",
```

```
        "Type": "CONVERSATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的识别发言者。

示例 5：转录存储为音频文件的医疗对话，最多两个备选转录

以下 start-medical-transcription-job 示例通过单个音频文件创建最多两个备选转录。每个转录具有关联的置信度。默认情况下，Amazon Transcribe 返回置信度最高的转录。您可以指定 Amazon Transcribe 返回置信度较低的其他转录。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
    --cli-input-json file://myfifthfile.json
```

的内容myfifthfile.json：

```
{
    "MedicalTranscriptionJobName": "alternatives-conversation-medical-transcription-
job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
    },
    "Settings":{
        "ShowAlternatives": true,
        "MaxAlternatives": 2
    }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "alternatives-conversation-medical-
transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
```

```
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-21T19:09:18.199000+00:00",
        "CreationTime": "2020-09-21T19:09:18.171000+00:00",
        "Settings": {
            "ShowAlternatives": true,
            "MaxAlternatives": 2
        },
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的备选转录。

示例 6：转录医疗口述音频文件，最多具有两个备选转录

以下 `start-medical-transcription-job` 示例转录一个音频文件，并使用词汇表过滤器屏蔽任何不需要的单词。您可以在 OutputBucketName 参数中指定转录输出位置。

```
aws transcribe start-medical-transcription-job \
    --cli-input-json file://mysixthfile.json
```

的内容`mysixthfile.json`：

```
{
    "MedicalTranscriptionJobName": "alternatives-conversation-medical-transcription-
job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
    },
    "Settings":{
          "ShowAlternatives": true,
          "MaxAlternatives": 2
    }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "alternatives-dictation-medical-
transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-21T21:01:14.592000+00:00",
        "CreationTime": "2020-09-21T21:01:14.569000+00:00",
        "Settings": {
            "ShowAlternatives": true,
            "MaxAlternatives": 2
        },
        "Specialty": "PRIMARYCARE",
        "Type": "DICTATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的备选转录。

示例 7：使用自定义词汇表更准确地转录医疗口述音频文件

以下 `start-medical-transcription-job` 示例转录一个音频文件，并使用您以前创建的医学自定义词汇表提高转录准确性。您可以在 `OutputBucketName` 参数中指定转录输出位置。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myseventhfile.json
```

的内容`mysixthfile.json`：

```
{
    "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-job",
    "LanguageCode": "language-code",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION",
    "OutputBucketName":"amzn-s3-demo-bucket",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
    },
    "Settings":{
```

```
        "VocabularyName": "cli-medical-vocab-1"
    }
}
```

输出：

```
{
    "MedicalTranscriptionJob": {
        "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-
job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.extension"
        },
        "StartTime": "2020-09-21T21:17:27.045000+00:00",
        "CreationTime": "2020-09-21T21:17:27.016000+00:00",
        "Settings": {
            "VocabularyName": "cli-medical-vocab-1"
        },
        "Specialty": "PRIMARYCARE",
        "Type": "DICTATION"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartMedicalTranscriptionJob。

start-transcription-job

以下代码示例演示了如何使用 start-transcription-job。

Amazon CLI

示例 1：转录音频文件

以下 start-transcription-job 示例转录音频文件。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myfile.json
```

的内容myfile.json：

```
{
    "TranscriptionJobName": "cli-simple-transcription-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的入门（Amazon 命令行界面）。

示例 2：转录多声道音频文件

以下 start-transcription-job 示例转录多声道音频文件。

```
aws transcribe start-transcription-job \
    --cli-input-json file://mysecondfile.json
```

的内容mysecondfile.json：

```
{
    "TranscriptionJobName": "cli-channelid-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    },
    "Settings":{
        "ChannelIdentification":true
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-channelid-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "the-language-of-your-transcription-job",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
```

```
        },
        "StartTime": "2020-09-17T16:07:56.817000+00:00",
        "CreationTime": "2020-09-17T16:07:56.784000+00:00",
        "Settings": {
            "ChannelIdentification": true
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的转录多声道音频。

示例 3：转录音频文件并识别不同的发言者

以下 start-transcription-job 示例转录音频文件，并在转录输出中识别发言者。

```
aws transcribe start-transcription-job \
    --cli-input-json file://mythirdfile.json
```

的内容mythirdfile.json：

```
{
    "TranscriptionJobName": "cli-speakerid-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    },
    "Settings":{
    "ShowSpeakerLabels": true,
    "MaxSpeakerLabels": 2
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-speakerid-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "the-language-of-your-transcription-job",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
```

```
        },
        "StartTime": "2020-09-17T16:22:59.696000+00:00",
        "CreationTime": "2020-09-17T16:22:59.676000+00:00",
        "Settings": {
            "ShowSpeakerLabels": true,
            "MaxSpeakerLabels": 2
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的识别发言者。

示例 4：转录音频文件并在转录输出中屏蔽任何不需要的单词

以下 start-transcription-job 示例转录音频文件，并使用您以前创建的词汇表过滤器屏蔽任何不需要的单词。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myfourthfile.json
```

的内容myfourthfile.json：

```
{
    "TranscriptionJobName": "cli-filter-mask-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
          "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
    },
    "Settings":{
        "VocabularyFilterName": "your-vocabulary-filter",
        "VocabularyFilterMethod": "mask"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-filter-mask-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "the-language-of-your-transcription-job",
        "Media": {
```

```
            "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
        },
        "StartTime": "2020-09-18T16:36:18.568000+00:00",
        "CreationTime": "2020-09-18T16:36:18.547000+00:00",
        "Settings": {
            "VocabularyFilterName": "your-vocabulary-filter",
            "VocabularyFilterMethod": "mask"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤转录。

示例 5：转录音频文件并在转录输出中删除任何不需要的单词

以下 start-transcription-job 示例转录音频文件，并使用您以前创建的词汇表过滤器屏蔽任何不需要的单词。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myfifthfile.json
```

的内容myfifthfile.json：

```
{
    "TranscriptionJobName": "cli-filter-remove-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    },
    "Settings":{
        "VocabularyFilterName": "your-vocabulary-filter",
        "VocabularyFilterMethod": "remove"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-filter-remove-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "the-language-of-your-transcription-job",
```

```
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
        },
        "StartTime": "2020-09-18T16:36:18.568000+00:00",
        "CreationTime": "2020-09-18T16:36:18.547000+00:00",
        "Settings": {
            "VocabularyFilterName": "your-vocabulary-filter",
            "VocabularyFilterMethod": "remove"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤转录。

示例 6：使用自定义词汇表更准确地转录音频文件

以下 start-transcription-job 示例转录音频文件，并使用您以前创建的词汇表过滤器屏蔽任何不需要的单词。

```
aws transcribe start-transcription-job \
    --cli-input-json file://mysixthfile.json
```

的内容mysixthfile.json：

```
{
    "TranscriptionJobName": "cli-vocab-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    },
    "Settings":{
        "VocabularyName": "your-vocabulary"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-vocab-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
```

```
        "LanguageCode": "the-language-of-your-transcription-job",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
        },
        "StartTime": "2020-09-18T16:36:18.568000+00:00",
        "CreationTime": "2020-09-18T16:36:18.547000+00:00",
        "Settings": {
            "VocabularyName": "your-vocabulary"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤转录。

示例 7：识别音频文件语言并转录该文件

以下 start-transcription-job 示例转录音频文件，并使用您以前创建的词汇表过滤器屏蔽任何不需要的单词。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myseventhfile.json
```

的内容myseventhfile.json：

```
{
    "TranscriptionJobName": "cli-identify-language-transcription-job",
    "IdentifyLanguage": true,
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-file-
name.file-extension"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-identify-language-transcription-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/Amazon-S3-prefix/your-media-
file-name.file-extension"
```

```
        },
        "StartTime": "2020-09-18T22:27:23.970000+00:00",
        "CreationTime": "2020-09-18T22:27:23.948000+00:00",
        "IdentifyLanguage": true
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的识别语言。

示例 8：转录音频文件并编辑个人身份信息

以下 start-transcription-job 示例转录音频文件，并在转录输出中编辑任何个人身份信息。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myeighthfile.json
```

的内容myeigthfile.json：

```
{
    "TranscriptionJobName": "cli-redaction-job",
    "LanguageCode": "language-code",
    "Media": {
        "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
    },
    "ContentRedaction": {
        "RedactionOutput":"redacted",
        "RedactionType":"PII"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-redaction-job",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
        },
        "StartTime": "2020-09-25T23:49:13.195000+00:00",
        "CreationTime": "2020-09-25T23:49:13.176000+00:00",
        "ContentRedaction": {
```

```
            "RedactionType": "PII",
            "RedactionOutput": "redacted"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自动内容编辑。

示例 9：生成个人身份信息 (PII) 经过编辑和未经过编辑的转录

以下 start-transcription-job 示例生成两个音频文件转录，一个转录中的个人身份信息经过编辑，另一个转录没有进行任何编辑。

```
aws transcribe start-transcription-job \
    --cli-input-json file://myninthfile.json
```

的内容myninthfile.json：

```
{
    "TranscriptionJobName": "cli-redaction-job-with-unredacted-transcript",
    "LanguageCode": "language-code",
    "Media": {
          "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
        },
    "ContentRedaction": {
        "RedactionOutput":"redacted_and_unredacted",
        "RedactionType":"PII"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-redaction-job-with-unredacted-transcript",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
        },
        "StartTime": "2020-09-25T23:59:47.677000+00:00",
        "CreationTime": "2020-09-25T23:59:47.653000+00:00",
```

```
        "ContentRedaction": {
            "RedactionType": "PII",
            "RedactionOutput": "redacted_and_unredacted"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自动内容编辑。

示例 10：使用您以前创建的自定义语言模型转录音频文件

以下 start-transcription-job 示例使用您以前创建的自定义语言模型转录音频文件。

```
aws transcribe start-transcription-job \
    --cli-input-json file://mytenthfile.json
```

的内容mytenthfile.json：

```
{
    "TranscriptionJobName": "cli-clm-2-job-1",
    "LanguageCode": "language-code",
    "Media": {
        "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.file-extension"
    },
    "ModelSettings": {
        "LanguageModelName":"cli-clm-2"
    }
}
```

输出：

```
{
    "TranscriptionJob": {
        "TranscriptionJobName": "cli-clm-2-job-1",
        "TranscriptionJobStatus": "IN_PROGRESS",
        "LanguageCode": "language-code",
        "Media": {
            "MediaFileUri": "s3://amzn-s3-demo-bucket/your-audio-file.file-
extension"
        },
        "StartTime": "2020-09-28T17:56:01.835000+00:00",
        "CreationTime": "2020-09-28T17:56:01.801000+00:00",
        "ModelSettings": {
```

```
            "LanguageModelName": "cli-clm-2"
        }
    }
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的使用自定义语言模型提高特定领域的转录准确性。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 StartTranscriptionJob。

update-medical-vocabulary

以下代码示例演示了如何使用 update-medical-vocabulary。

Amazon CLI

使用新术语更新医疗领域自定义词汇表。

以下 update-medical-vocabulary 示例用新术语替换了医疗领域自定义词汇表中使用的术语。先决条件：要替换医疗领域自定义词汇表中的术语，需要一个包含新术语的文件。

```
aws transcribe update-medical-vocabulary \
    --vocabulary-file-uri s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/medical-custom-vocabulary.txt \
    --vocabulary-name medical-custom-vocabulary \
    --language-code language
```

输出：

```
{
    "VocabularyName": "medical-custom-vocabulary",
    "LanguageCode": "en-US",
    "VocabularyState": "PENDING"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的医疗自定义词汇表。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMedicalVocabulary。

update-vocabulary-filter

以下代码示例演示了如何使用 update-vocabulary-filter。

Amazon CLI

替换词汇表过滤器中的字词

以下 update-vocabulary-filter 示例将词汇表过滤器中的字词替换为新字词。先决条件：要使用新字词更新词汇表过滤器，必须将这些字词另存为文本文件。

```
aws transcribe update-vocabulary-filter \
    --vocabulary-filter-file-uri s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/your-text-file-to-update-your-vocabulary-filter.txt \
    --vocabulary-filter-name vocabulary-filter-name
```

输出：

```
{
    "VocabularyFilterName": "vocabulary-filter-name",
    "LanguageCode": "language-code",
    "LastModifiedTime": "2020-09-23T18:40:35.139000+00:00"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的过滤不需要的字词。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateVocabularyFilter。

update-vocabulary

以下代码示例演示了如何使用 update-vocabulary。

Amazon CLI

使用新术语更新自定义词汇表。

以下 update-vocabulary 示例使用您提供的新术语来覆盖用于创建自定义词汇表的术语。先决条件：要替换自定义词汇表中的术语，您需要使用一个包含新术语的文件。

```
aws transcribe update-vocabulary \
    --vocabulary-file-uri s3://amzn-s3-demo-bucket/Amazon-S3-Prefix/custom-vocabulary.txt \
    --vocabulary-name custom-vocabulary \
    --language-code language-code
```

输出：

```
{
    "VocabularyName": "custom-vocabulary",
    "LanguageCode": "language",
    "VocabularyState": "PENDING"
}
```

有关更多信息，请参阅《Amazon Transcribe 开发人员指南》中的自定义词汇表。

- 有关 API 详细信息，请参阅 Amazon CLI 命令参考 中的 UpdateVocabulary。

使用 Amazon CLI 的 Amazon Translate 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon Translate 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`import-terminology`

以下代码示例演示了如何使用 `import-terminology`。

Amazon CLI

从文件导入自定义术语

以下 `import-terminology` 示例从 `test-terminology.csv` 文件创建一个称为 `MyTestTerminology` 的术语：

```
aws translate import-terminology \
    --name MyTestTerminology \
    --description "Creating a test terminology in AWS Translate" \
    --merge-strategy OVERWRITE \
```

```
    --data-file fileb://test-terminology.csv \
    --terminology-data Format=CSV
```

的内容`test-terminology.csv`：

en,fr,es,zh Hello world!,Bonjour tout le monde!,Hola Mundo!,????
Amazon,Amazon,Amazon,Amazon

输出：

```
{
    "TerminologyProperties": {
        "SourceLanguageCode": "en",
        "Name": "MyTestTerminology",
        "TargetLanguageCodes": [
            "fr",
            "es",
            "zh"
        ],
        "SizeBytes": 97,
        "LastUpdatedAt": 1571089500.851,
        "CreatedAt": 1571089500.851,
        "TermCount": 6,
        "Arn": "arn:aws:translate:us-west-2:123456789012:terminology/
MyTestTerminology/LATEST",
        "Description": "Creating a test terminology in AWS Translate"
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ImportTerminology。

使用 Amazon CLI 的 Trusted Advisor 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Trusted Advisor 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

get-organization-recommendation

以下代码示例演示了如何使用 get-organization-recommendation。

Amazon CLI

获取组织建议

以下 get-organization-recommendation 示例按其标识符获取组织建议。

```
aws trustedadvisor get-organization-recommendation \
    --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-recommendation/9534ec9b-bf3a-44e8-8213-2ed68b39d9d5
```

输出：

```
{
    "organizationRecommendation": {
        "arn": "arn:aws:trustedadvisor:::organization-recommendation/9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
        "name": "Lambda Runtime Deprecation Warning",
        "description": "One or more lambdas are using a deprecated runtime",
        "awsServices": [
            "lambda"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/L4dfs2Q4C5",
        "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
        "lifecycleStage": "resolved",
        "pillars": [
            "security"
        ],
        "resourcesAggregates": {
            "errorCount": 0,
            "okCount": 0,
            "warningCount": 0
        },
        "source": "ta_check",
```

```
        "status": "warning",
        "type": "priority"
    }
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetOrganizationRecommendation。

`get-recommendation`

以下代码示例演示了如何使用 `get-recommendation`。

Amazon CLI

获取建议

以下 `get-recommendation` 示例按其标识符获取建议。

```
aws trustedadvisor get-recommendation \
    --recommendation-
identifier arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578
```

输出：

```
{
    "recommendation": {
        "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
        "name": "MFA Recommendation",
        "description": "Enable multi-factor authentication",
        "awsServices": [
            "iam"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/7DAFEmoDos",
        "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
        "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
        "pillarSpecificAggregates": {
            "costOptimizing": {
                "estimatedMonthlySavings": 0.0,
                "estimatedPercentMonthlySavings": 0.0
            }
```

```
        },
        "pillars": [
            "security"
        ],
        "resourcesAggregates": {
            "errorCount": 1,
            "okCount": 0,
            "warningCount": 0
        },
        "source": "ta_check",
        "status": "error",
        "type": "standard"
    }
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRecommendation。

`list-checks`

以下代码示例演示了如何使用 `list-checks`。

Amazon CLI

列出 Trusted Advisor 检查

以下 `list-checks` 示例列出了所有 Trusted Advisor 检查。

```
aws trustedadvisor list-checks
```

输出：

```
{
    "checkSummaries": [
        {
            "arn": "arn:aws:trustedadvisor:::check/1iG5NDGVre",
            "awsServices": [
                "EC2"
            ],
            "description": "Checks security groups for rules that allow unrestricted
 access to a resource. Unrestricted access increases opportunities for malicious
 activity (hacking, denial-of-service attacks, loss of data)",
```

```
            "id": "1iG5NDGVre",
            "metadata": {
                "0": "Region",
                "1": "Security Group Name",
                "2": "Security Group ID",
                "3": "Protocol",
                "4": "Port",
                "5": "Status",
                "6": "IP Range"
            },
            "name": "Security Groups - Unrestricted Access",
            "pillars": [
                "security"
            ],
            "source": "ta_check"
        },
        {
            "arn": "arn:aws:trustedadvisor:::check/1qazXsw23e",
            "awsServices": [
                "RDS"
            ],
            "description": "Checks your usage of RDS and provides recommendations
on purchase of Reserved Instances to help reduce costs incurred from using RDS
On-Demand. AWS generates these recommendations by analyzing your On-Demand usage
for the past 30 days. We then simulate every combination of reservations in the
generated category of usage in order to identify the best number of each type
of Reserved Instance to purchase to maximize your savings. This check covers
recommendations based on partial upfront payment option with 1-year or 3-year
commitment. This check is not available to accounts linked in Consolidated Billing.
Recommendations are only available for the Paying Account.",
            "id": "1qazXsw23e",
            "metadata": {
                "0": "Region",
                "1": "Family",
                "2": "Instance Type",
                "3": "License Model",
                "4": "Database Edition",
                "5": "Database Engine",
                "6": "Deployment Option",
                "7": "Recommended number of Reserved Instances to purchase",
                "8": "Expected Average Reserved Instance Utilization",
                "9": "Estimated Savings with Recommendation (monthly)"
                "10": "Upfront Cost of Reserved Instances",
                "11": "Estimated cost of Reserved Instances (monthly)",
```

```
                "12": "Estimated On-Demand Cost Post Recommended Reserved Instance
Purchase (monthly)",
                "13": "Estimated Break Even (months)",
                "14": "Lookback Period (days)",
                "15": "Term (years)"
            },
            "name": "Amazon Relational Database Service (RDS) Reserved Instance
Optimization",
            "pillars": [
                "cost_optimizing"
            ],
            "source": "ta_check"
        },
        {
            "arn": "arn:aws:trustedadvisor:::check/1qw23er45t",
            "awsServices": [
                "Redshift"
            ],
            "description": "Checks your usage of Redshift and provides
recommendations on purchase of Reserved Nodes to help reduce costs incurred from
using Redshift On-Demand. AWS generates these recommendations by analyzing your
On-Demand usage for the past 30 days. We then simulate every combination of
reservations in the generated category of usage in order to identify the best
number of each type of Reserved Nodes to purchase to maximize your savings. This
check covers recommendations based on partial upfront payment option with 1-year or
3-year commitment. This check is not available to accounts linked in Consolidated
Billing. Recommendations are only available for the Paying Account.",
            "id": "1qw23er45t",
            "metadata": {
                "0": "Region",
                "1": "Family",
                "2": "Node Type",
                "3": "Recommended number of Reserved Nodes to purchase",
                "4": "Expected Average Reserved Node Utilization",
                "5": "Estimated Savings with Recommendation (monthly)",
                "6": "Upfront Cost of Reserved Nodes",
                "7": "Estimated cost of Reserved Nodes (monthly)",
                "8": "Estimated On-Demand Cost Post Recommended Reserved Nodes
Purchase (monthly)",
                "9": "Estimated Break Even (months)",
                "10": "Lookback Period (days)",
                "11": "Term (years)",
            },
            "name": "Amazon Redshift Reserved Node Optimization",
```

```
            "pillars": [
                "cost_optimizing"
            ],
            "source": "ta_check"
        },
    ],
    "nextToken": "REDACTED"
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListChecks。

`list-organization-recommendation-accounts`

以下代码示例演示了如何使用 `list-organization-recommendation-accounts`。

Amazon CLI

列出组织建议账户

以下 `list-organization-recommendation-accounts` 示例按其标识符列出组织建议的所有账户建议摘要。

```
aws trustedadvisor list-organization-recommendation-accounts \
    --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-recommendation/9534ec9b-bf3a-44e8-8213-2ed68b39d9d5
```

输出：

```
{
    "accountRecommendationLifecycleSummaries": [{
        "accountId": "000000000000",
        "accountRecommendationArn":
 "arn:aws:trustedadvisor::000000000000:recommendation/9534ec9b-
bf3a-44e8-8213-2ed68b39d9d5",
        "lifecycleStage": "resolved",
        "updateReason": "Resolved issue",
        "updateReasonCode": "valid_business_case",
        "lastUpdatedAt": "2023-01-17T18:25:44.552Z"
    }],
    "nextToken": "REDACTED"
```

```
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizationRecommendationAccounts。

list-organization-recommendation-resources

以下代码示例演示了如何使用 list-organization-recommendation-resources。

Amazon CLI

列出组织建议资源

以下 list-organization-recommendation-resources 示例按其标识符列出组织建议的所有资源。

```
aws trustedadvisor list-organization-recommendation-resources \
    --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-recommendation/5a694939-2e54-45a2-ae72-730598fa89d0
```

输出：

```
{
    "organizationRecommendationResourceSummaries": [
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-resource/5a694939-2e54-45a2-ae72-730598fa89d0/bb38affc0ce0681d9a6cd13f30238ba03a8f63dfe7a379dc403c619119d86af",
            "awsResourceId": "database-1-instance-1",
            "id": "bb38affc0ce0681d9a6cd13f302383ba03a8f63dfe7a379dc403c619119d86af",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
            "metadata": {
                "0": "14",
                "1": "208.79999999999998",
                "2": "database-1-instance-1",
                "3": "db.r5.large",
                "4": "false",
                "5": "us-west-2",
                "6": "arn:aws:rds:us-west-2:000000000000:db:database-1-instance-1",
                "7": "1"
```

```
            },
            "recommendationArn": "arn:aws:trustedadvisor:::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
            "regionCode": "us-west-2",
            "status": "warning"
        },
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/5a694939-2e54-45a2-
ae72-730598fa89d0/51fded4d7a3278818df9cfe344ff5762cec46c095a6763d1ba1ba53bd0e1b0e6",
            "awsResourceId": "database-1",
            "id":
 "51fded4d7a3278818df9cfe344ff5762cec46c095a6763d1ba1ba53bd0e1b0e6",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
            "metadata": {
                "0": "14",
                "1": "31.679999999999996",
                "2": "database-1",
                "3": "db.t3.small",
                "4": "false",
                "5": "us-west-2",
                "6": "arn:aws:rds:us-west-2:000000000000:db:database-1",
                "7": "20"
            },
            "recommendationArn": "arn:aws:trustedadvisor:::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
            "regionCode": "us-west-2",
            "status": "warning"
        },
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/5a694939-2e54-45a2-ae72-730598fa89d0/
f4d01bd20f4cd5372062aafc8786c489e48f0ead7cdab121463bf9f89e40a36b",
            "awsResourceId": "database-2-instance-1-us-west-2a",
            "id":
 "f4d01bd20f4cd5372062aafc8786c489e48f0ead7cdab121463bf9f89e40a36b",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
            "metadata": {
                "0": "14",
                "1": "187.20000000000002",
                "2": "database-2-instance-1-us-west-2a",
                "3": "db.r6g.large",
                "4": "true",
                "5": "us-west-2",
```

```
                "6": "arn:aws:rds:us-west-2:000000000000:db:database-2-instance-1-
us-west-2a",
                "7": "1"
            },
            "recommendationArn": "arn:aws:trustedadvisor:::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
            "regionCode": "us-west-2",
            "status": "warning"
        },
    ],
    "nextToken": "REDACTED"
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizationRecommendationResources。

list-organization-recommendations

以下代码示例演示了如何使用 list-organization-recommendations。

Amazon CLI

示例 1：列出组织建议

以下 list-organization-recommendations 示例列出了所有组织建议，但不包括筛选条件。

```
aws trustedadvisor list-organization-recommendations
```

输出：

```
{
    "organizationRecommendationSummaries": [
        {
            "arn": "arn:aws:trustedadvisor:::organization-recommendation/9534ec9b-
bf3a-44e8-8213-2ed68b39d9d5",
            "name": "Lambda Runtime Deprecation Warning",
            "awsServices": [
                "lambda"
            ],
            "checkArn": "arn:aws:trustedadvisor:::check/L4dfs2Q4C5",
```

```
            "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
            "lifecycleStage": "resolved",
            "pillars": [
                "security"
            ],
            "resourcesAggregates": {
                "errorCount": 0,
                "okCount": 0,
                "warningCount": 0
            },
            "source": "ta_check",
            "status": "warning",
            "type": "priority"
        },
        {
            "arn": "arn:aws:trustedadvisor:::organization-
recommendation/4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
            "name": "Lambda Runtime Deprecation Warning",
            "awsServices": [
                "lambda"
            ],
            "checkArn": "arn:aws:trustedadvisor:::check/L4dfs2Q4C5",
            "id": "4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
            "lifecycleStage": "resolved",
            "pillars": [
                "security"
            ],
            "resourcesAggregates": {
                "errorCount": 0,
                "okCount": 0,
                "warningCount": 0
            },
            "source": "ta_check",
            "status": "warning",
            "type": "priority"
        },
    ],
    "nextToken": "REDACTED"
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

示例 2：使用筛选条件列出组织建议

以下 list-organization-recommendations 示例筛选并返回最多一项属于 “安全” 支柱的组织建议。

```
aws trustedadvisor list-organization-recommendations \
    --pillar security \
    --max-items 100
```

输出：

```
{
    "organizationRecommendationSummaries": [{
        "arn": "arn:aws:trustedadvisor:::organization-recommendation/9534ec9b-
bf3a-44e8-8213-2ed68b39d9d5",
        "name": "Lambda Runtime Deprecation Warning",
        "awsServices": [
            "lambda"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/L4dfs2Q4C5",
        "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
        "lifecycleStage": "resolved",
        "pillars": [
            "security"
        ],
        "resourcesAggregates": {
            "errorCount": 0,
            "okCount": 0,
            "warningCount": 0
        },
        "source": "ta_check",
        "status": "warning",
        "type": "priority"
    }],
    "nextToken": "REDACTED"
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

示例 3：使用分页令牌列出组织建议

以下 list-organization-recommendations 示例使用从上一个请求返回的 “nextToken” 来获取下一页的组织建议。

```
aws trustedadvisor list-organization-recommendations \
```

```
    --pillar security \
    --max-items 100 \
    --starting-token <next-token>
```

输出：

```
{
    "organizationRecommendationSummaries": [{
        "arn": "arn:aws:trustedadvisor:::organization-
recommendation/4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
        "name": "Lambda Runtime Deprecation Warning",
        "awsServices": [
            "lambda"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/L4dfs2Q4C5",
        "id": "4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
        "lifecycleStage": "resolved",
        "pillars": [
            "security"
        ],
        "resourcesAggregates": {
            "errorCount": 0,
            "okCount": 0,
            "warningCount": 0
        },
        "source": "ta_check",
        "status": "warning",
        "type": "priority"
    }]
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizationRecommendations。

`list-recommendation-resources`

以下代码示例演示了如何使用 `list-recommendation-resources`。

Amazon CLI

列出建议资源

以下 list-recommendation-resources 示例按其标识符列出建议的所有资源。

```
aws trustedadvisor list-recommendation-resources \
    --recommendation-
identifier arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578
```

输出：

```
{
    "recommendationResourceSummaries": [
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-
bbb7-491a-833b-5773e9589578/18959a1f1973cff8e706e9d9bde28bba36cd602a6b2cb86c8b61252835236010
            "id":
 "18959a1f1973cff8e706e9d9bde28bba36cd602a6b2cb86c8b61252835236010",
            "awsResourceId": "webcms-dev-01",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
            "metadata": {
                "0": "14",
                "1": "123.12000000000002",
                "2": "webcms-dev-01",
                "3": "db.m6i.large",
                "4": "false",
                "5": "us-east-1",
                "6": "arn:aws:rds:us-east-1:000000000000:db:webcms-dev-01",
                "7": "20"
            },
            "recommendationArn":
 "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
            "regionCode": "us-east-1",
            "status": "warning"
        },
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-bbb7-491a-833b-5773e9589578/
e6367ff500ac90db8e4adeb4892e39ee9c36bbf812dcbce4b9e4fefcec9eb63e",
            "id":
 "e6367ff500ac90db8e4adeb4892e39ee9c36bbf812dcbce4b9e4fefcec9eb63e",
            "awsResourceId": "aws-dev-db-stack-instance-1",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
```

```
            "metadata": {
                "0": "14",
                "1": "29.52",
                "2": "aws-dev-db-stack-instance-1",
                "3": "db.t2.small",
                "4": "false",
                "5": "us-east-1",
                "6": "arn:aws:rds:us-east-1:000000000000:db:aws-dev-db-stack-
instance-1",
                "7": "1"
            },
            "recommendationArn":
 "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
            "regionCode": "us-east-1",
            "status": "warning"
        },
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-
bbb7-491a-833b-5773e9589578/31aa78ba050a5015d2d38cca7f5f1ce88f70857c4e1c3ad03f8f9fd95dad7459
            "id":
 "31aa78ba050a5015d2d38cca7f5f1ce88f70857c4e1c3ad03f8f9fd95dad7459",
            "awsResourceId": "aws-awesome-apps-stack-db",
            "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
            "metadata": {
                "0": "14",
                "1": "114.48000000000002",
                "2": "aws-awesome-apps-stack-db",
                "3": "db.m6g.large",
                "4": "false",
                "5": "us-east-1",
                "6": "arn:aws:rds:us-east-1:000000000000:db:aws-awesome-apps-stack-
db",
                "7": "100"
            },
            "recommendationArn":
 "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
            "regionCode": "us-east-1",
            "status": "warning"
        }
    ],
    "nextToken": "REDACTED"
```

```
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRecommendationResources。

`list-recommendations`

以下代码示例演示了如何使用 `list-recommendations`。

Amazon CLI

示例 1：列出建议

以下 `list-recommendations` 示例列出了所有建议，但不包括筛选条件。

```
aws trustedadvisor list-recommendations
```

输出：

```
{
    "recommendationSummaries": [
        {
            "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
            "name": "MFA Recommendation",
            "awsServices": [
                "iam"
            ],
            "checkArn": "arn:aws:trustedadvisor:::check/7DAFEmoDos",
            "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
            "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
            "pillarSpecificAggregates": {
                "costOptimizing": {
                    "estimatedMonthlySavings": 0.0,
                    "estimatedPercentMonthlySavings": 0.0
                }
            },
            "pillars": [
                "security"
            ],
            "resourcesAggregates": {
```

```
                "errorCount": 1,
                "okCount": 0,
                "warningCount": 0
            },
            "source": "ta_check",
            "status": "error",
            "type": "standard"
        },
        {
            "arn":
 "arn:aws:trustedadvisor::000000000000:recommendation/8b602b6f-452d-4cb2-8a9e-
c7650955d9cd",
            "name": "RDS clusters quota warning",
            "awsServices": [
                "rds"
            ],
            "checkArn": "arn:aws:trustedadvisor:::check/gjqMBn6pjz",
            "id": "8b602b6f-452d-4cb2-8a9e-c7650955d9cd",
            "lastUpdatedAt": "2023-11-01T15:58:17.397Z",
            "pillarSpecificAggregates": {
                "costOptimizing": {
                    "estimatedMonthlySavings": 0.0,
                    "estimatedPercentMonthlySavings": 0.0
                }
            },
            "pillars": [
                "service_limits"
            ],
            "resourcesAggregates": {
                "errorCount": 0,
                "okCount": 3,
                "warningCount": 6
            },
            "source": "ta_check",
            "status": "warning",
            "type": "standard"
        }
    ],
    "nextToken": "REDACTED"
}
```


有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

示例 2：使用筛选条件列出建议

以下 list-recommendations 示例列出建议并包含筛选条件。

```
aws trustedadvisor list-recommendations \
    --aws-service iam \
    --max-items 100
```

输出：

```
{
    "recommendationSummaries": [{
        "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
        "name": "MFA Recommendation",
        "awsServices": [
            "iam"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/7DAFEmoDos",
        "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
        "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
        "pillarSpecificAggregates": {
            "costOptimizing": {
                "estimatedMonthlySavings": 0.0,
                "estimatedPercentMonthlySavings": 0.0
            }
        },
        "pillars": [
            "security"
        ],
        "resourcesAggregates": {
            "errorCount": 1,
            "okCount": 0,
            "warningCount": 0
        },
        "source": "ta_check",
        "status": "error",
        "type": "standard"
    }],
    "nextToken": "REDACTED"
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

示例 3：使用分页令牌列出建议

以下 `list-recommendations` 示例使用从上一个请求返回的 “nextToken” 来获取下一页经过筛选的建议。

```
aws trustedadvisor list-recommendations \
    --aws-service rds \
    --max-items 100 \
    --starting-token <next-token>
```

输出：

```
{
    "recommendationSummaries": [{
        "arn":
 "arn:aws:trustedadvisor::000000000000:recommendation/8b602b6f-452d-4cb2-8a9e-
c7650955d9cd",
        "name": "RDS clusters quota warning",
        "awsServices": [
            "rds"
        ],
        "checkArn": "arn:aws:trustedadvisor:::check/gjqMBn6pjz",
        "id": "8b602b6f-452d-4cb2-8a9e-c7650955d9cd",
        "lastUpdatedAt": "2023-11-01T15:58:17.397Z",
        "pillarSpecificAggregates": {
            "costOptimizing": {
                "estimatedMonthlySavings": 0.0,
                "estimatedPercentMonthlySavings": 0.0
            }
        },
        "pillars": [
            "service_limits"
        ],
        "resourcesAggregates": {
            "errorCount": 0,
            "okCount": 3,
            "warningCount": 6
        },
        "source": "ta_check",
        "status": "warning",
        "type": "standard"
    }]
}
```

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRecommendations。

update-organization-recommendation-lifecycle

以下代码示例演示了如何使用 update-organization-recommendation-lifecycle。

Amazon CLI

更新组织建议生命周期

以下 update-organization-recommendation-lifecycle 示例按其标识符更新组织建议的生命周期。

```
aws trustedadvisor update-organization-recommendation-lifecycle \
    --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-recommendation/96b5e5ca-7930-444c-90c6-06d386128100 \
    --lifecycle-stage dismissed \
    --update-reason-code not_applicable
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateOrganizationRecommendationLifecycle。

update-recommendation-lifecycle

以下代码示例演示了如何使用 update-recommendation-lifecycle。

Amazon CLI

更新建议生命周期

以下 update-recommendation-lifecycle 示例按其标识符更新建议的生命周期。

```
aws trustedadvisor update-recommendation-lifecycle \
    --recommendation-identifier arn:aws:trustedadvisor::000000000000:recommendation/861c9c6e-f169-405a-8b59-537a8caccd7a \
    --lifecycle-stage resolved \
    --update-reason-code valid_business_case
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Trusted Advisor 用户指南》中的 Trusted Advisor API 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRecommendationLifecycle。

使用 Amazon CLI 的 Verified Permissions 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Verified Permissions 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-identity-source`

以下代码示例演示了如何使用 `create-identity-source`。

Amazon CLI

创建身份源

以下 `create-identity-source` 示例创建一个身份源，以便您能够引用存储在指定 Amazon Cognito 用户池中的身份。这些身份作为 `User` 实体类型在 Verified Permissions 中提供。

```
aws verifiedpermissions create-identity-source \
    --configuration file://config.txt \
    --principal-entity-type "User" \
    --policy-store-id PSEXAMPLEabcdefg111111
```

的内容`config.txt`：

```
{
```

```
    "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_1a2b3c4d5",
        "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"]
    }
}
```

输出：

```
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

有关身份源的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIdentitySource。

create-policy-store

以下代码示例演示了如何使用 create-policy-store。

Amazon CLI

创建策略存储

以下 create-policy-store 示例在当前 Amazon 区域中创建策略存储。

```
aws verifiedpermissions create-policy-store \
    --validation-settings "mode=STRICT"
```

输出：

```
{
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-05-16T17:41:29.103459+00:00",
    "lastUpdatedDate": "2023-05-16T17:41:29.103459+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
```

```
}
```

有关策略存储的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicyStore。

`create-policy-template`

以下代码示例演示了如何使用 `create-policy-template`。

Amazon CLI

创建策略模板

以下 `create-policy-template` 示例使用包含主体占位符的声明创建策略模板。

```
aws verifiedpermissions create-policy-template \
    --statement file://template1.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

的内容`template1.txt`：

```
permit(
    principal in ?principal,
    action == Action::"view",
    resource == Photo::"VacationPhoto94.jpg"
);
```

输出：

```
{
    "createdDate": "2023-06-12T20:47:42.804511+00:00",
    "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
}
```

有关更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicyTemplate。

create-policy

以下代码示例演示了如何使用 create-policy。

Amazon CLI

示例 1：创建静态策略

以下 create-policy 示例创建一个静态策略，其策略范围指定了主体和资源。

```
aws verifiedpermissions create-policy \
    --definition file://definition1.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

definition1.txt 文件的内容：

```
{
    "static": {
        "description":  "Grant everyone of janeFriends UserGroup access to the
 vacationFolder Album",
        "statement": "permit(principal in UserGroup::\"janeFriends\", action,
 resource in Album::\"vacationFolder\" );"
    }
}
```

输出：

```
{
    "createdDate": "2023-06-12T20:33:37.382907+00:00",
    "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityId": "janeFriends",
        "entityType": "UserGroup"
    },
    "resource": {
        "entityId": "vacationFolder",
        "entityType": "Album"
    }
}
```

示例 2：创建向所有人授予资源访问权限的静态策略

以下 create-policy 示例创建一个静态策略，其策略范围仅指定了资源。

```
aws verifiedpermissions create-policy \
    --definition file://definition2.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

definition2.txt 文件的内容：

```
{
    "static": {
        "description":  "Grant everyone access to the publicFolder Album",
        "statement": "permit(principal, action, resource in Album::\"publicFolder
\");"
    }
}
```

输出：

```
{
    "createdDate": "2023-06-12T20:39:44.975897+00:00",
    "lastUpdatedDate": "2023-06-12T20:39:44.975897+00:00",
    "policyId": "PbfR73F8oh5MMfr9uRtFDB",
    "policyStoreId": "PSEXAMPLEabcdefg222222",
    "policyType": "STATIC",
    "resource": {
        "entityId": "publicFolder",
        "entityType": "Album"
    }
}
```

示例 3：创建与指定模板关联的模板链接策略

以下 create-policy 示例使用指定策略模板创建模板关联策略，并将要使用的指定主体与新的模板关联策略相关联。

```
aws verifiedpermissions create-policy \
    --definition file://definition.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

的内容definition.txt：

```
{
    "templateLinked": {
        "policyTemplateId": "PTEXAMPLEabcdefg111111",
        "principal": {
            "entityType": "User",
            "entityId": "alice"
        }
    }
}
```

输出：

```
{
    "createdDate": "2023-06-12T20:49:51.490211+00:00",
    "lastUpdatedDate": "2023-06-12T20:49:51.490211+00:00",
    "policyId": "TPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "TEMPLATE_LINKED",
    "principal": {
        "entityId": "alice",
        "entityType": "User"
    },
    "resource": {
        "entityId": "VacationPhoto94.jpg",
        "entityType": "Photo"
    }
}
```

有关策略的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicy。

delete-identity-source

以下代码示例演示了如何使用 delete-identity-source。

Amazon CLI

删除身份源

以下 delete-identity-source 示例删除具有指定 ID 的身份源。

```
aws verifiedpermissions delete-identity-source \
    --identity-source-id ISEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

此命令不生成任何输出。

有关身份源的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIdentitySource。

delete-policy-store

以下代码示例演示了如何使用 delete-policy-store。

Amazon CLI

删除策略存储

以下 delete-policy-store 示例删除具有指定 ID 的策略存储。

```
aws verifiedpermissions delete-policy-store \
    --policy-store-id PSEXAMPLEabcdefg111111
```

此命令不生成任何输出。

有关策略存储的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicyStore。

delete-policy-template

以下代码示例演示了如何使用 delete-policy-template。

Amazon CLI

删除策略模板

以下 delete-policy-template 示例删除具有指定 ID 的策略模板。

```
aws verifiedpermissions delete-policy \
```

```
    --policy-template-id PTEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicyTemplate。

delete-policy

以下代码示例演示了如何使用 delete-policy。

Amazon CLI

删除静态或模板链接策略

以下 delete-policy 示例删除具有指定 ID 的策略。

```
aws verifiedpermissions delete-policy \
    --policy-id SPEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

此命令不生成任何输出。

有关策略的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

get-identity-source

以下代码示例演示了如何使用 get-identity-source。

Amazon CLI

检索有关身份源的详细信息

以下 get-identity-source 示例显示具有指定 ID 的身份源的详细信息。

```
aws verifiedpermissions get-identity-source \
```

```
    --identity-source  ISEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "createdDate": "2023-06-12T22:27:49.150035+00:00",
    "details": {
        "clientIds": [ "a1b2c3d4e5f6g7h8i9j0kalbmc" ],
        "discoveryUrl": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_1a2b3c4d5",
        "openIdIssuer": "COGNITO",
        "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_1a2b3c4d5"
    },
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-06-12T22:27:49.150035+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "principalEntityType": "User"
}
```

有关身份源的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIdentitySource。

get-policy-store

以下代码示例演示了如何使用 get-policy-store。

Amazon CLI

检索有关策略存储的详细信息

以下 get-policy-store 示例显示具有指定 ID 的策略存储的详细信息。

```
aws verifiedpermissions get-policy-store \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
```

```
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-06-05T20:16:46.225598+00:00",
    "lastUpdatedDate": "2023-06-08T20:40:23.173691+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "validationSettings": { "mode": "OFF" }
}
```

有关策略存储的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicyStore。

get-policy-template

以下代码示例演示了如何使用 get-policy-template。

Amazon CLI

检索有关策略模板的详细信息

以下 get-policy-template 示例显示具有指定 ID 的策略模板的详细信息。

```
aws verifiedpermissions get-policy-template \
    --policy-template-id  PTEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "createdDate": "2023-06-12T20:47:42.804511+00:00",
    "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111",
    "statement": "permit(\n    principal in ?principal,\n    action == Action::
\"view\",\n    resource == Photo::\"VacationPhoto94.jpg\"\n);"
}
```

有关更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicyTemplate。

get-policy

以下代码示例演示了如何使用 get-policy。

Amazon CLI

检索有关策略的详细信息

以下 get-policy 示例显示具有指定 ID 的策略的详细信息。

```
aws verifiedpermissions get-policy \
    --policy-id PSEXAMPLEabcdefg111111 \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "createdDate": "2023-06-12T20:33:37.382907+00:00",
    "definition": {
        "static": {
            "description": "Grant everyone of janeFriends UserGroup access to the
 vacationFolder Album",
            "statement": "permit(principal in UserGroup::\"janeFriends\", action,
 resource in Album::\"vacationFolder\" );"
        }
    },
    "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityId": "janeFriends",
        "entityType": "UserGroup"
    },
    "resource": {
        "entityId": "vacationFolder",
        "entityType": "Album"
    }
}
```

有关策略的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetPolicy。

get-schema

以下代码示例演示了如何使用 get-schema。

Amazon CLI

检索策略存储中的架构

以下 get-schema 示例显示指定策略存储中架构的详细信息。

```
aws verifiedpermissions get-schema \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "schema": "{\"MySampleNamespace\":{\"entityTypes\":{\"Employee\":{\"shape
\":{\"attributes\":{\"jobLevel\":{\"type\":\"Long\"},\"name\":{\"type\":\"String
\"}},\"type\":\"Record\"}}},\"actions\":{\"remoteAccess\":{\"appliesTo\":
{\"principalTypes\":[\"Employee\"]}}}}}",
    "createdDate": "2023-06-14T17:47:13.999885+00:00",
    "lastUpdatedDate": "2023-06-14T17:47:13.999885+00:00"
}
```

有关架构的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的策略存储架构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSchema。

is-authorized-with-token

以下代码示例演示了如何使用 is-authorized-with-token。

Amazon CLI

示例 1：请求对用户请求做出授权决定（允许）

以下 is-authorized-with-token 示例请求对已通过 Amazon Cognito 身份验证的用户做出授权决定。该请求使用 Cognito 提供的身份令牌，而不是访问令牌。在此示例中，将指定信息存储配置为将主体作为 CognitoUser 实体类型返回。

```
aws verifiedpermissions is-authorized-with-token \
    --action actionId="View",actionType="Action" \
    --resource entityId="vacationPhoto94.jpg",entityType="Photo" \
    --policy-store-id PSEXAMPLEabcdefg111111 \
    --identity-token "AbCdE12345...long.string...54321EdCbA"
```

策略存储包含一个带以下声明的策略，该声明接受来自指定 Cognito 用户池和应用程序 ID 的身份。

```
permit(
    principal == CognitoUser::"us-east-1_1a2b3c4d5|a1b2c3d4e5f6g7h8i9j0kalbmc",
    action,
    resource == Photo::"VacationPhoto94.jpg"
);
```

输出：

```
{
    "decision":"Allow",
    "determiningPolicies":[
        {
        "determiningPolicyId":"SPEXAMPLEabcdefg111111"
        }
    ],
    "errors":[]
}
```

有关使用 Cognito 用户池中身份的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IsAuthorizedWithToken。

is-authorized

以下代码示例演示了如何使用 `is-authorized`。

Amazon CLI

示例 1：请求对用户请求做出授权决定（允许）

以下 `is-authorized` 示例请求对名为 `Alice` 的主体类型 `User` 做出授权决定，该实体想要对名为 `VacationPhoto94.jpg` 的资源类型 `Photo` 执行 `updatePhoto` 操作。

响应表明某项策略允许该请求。

```
aws verifiedpermissions is-authorized \
    --principal entityType=User,entityId=alice \
    --action actionType=Action,actionId=view \
    --resource entityType=Photo,entityId=VactionPhoto94.jpg \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "decision": "ALLOW",
    "determiningPolicies": [
        {
            "policyId": "SPEXAMPLEabcdefg111111"
        }
    ],
    "errors": []
}
```

示例 2：请求对用户请求做出授权决定（拒绝）

除了主体是 User::"Bob" 之外，以下示例与前面的示例相同。策略存储不包含任何允许该用户访问 Album::"alice_folder" 的策略。

输出表明 Deny 是隐式的，因为列表 DeterminingPolicies 为空。

```
aws verifiedpermissions create-policy \
    --definition file://definition2.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "decision": "DENY",
    "determiningPolicies": [],
    "errors": []
}
```

有关更多信息，请参阅 Amazon Verified Permissions 用户指南。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 IsAuthorized。

list-identity-sources

以下代码示例演示了如何使用 list-identity-sources。

Amazon CLI

列出可用身份源

以下 list-identity-sources 示例列出了指定策略存储中的所有身份源。

```
aws verifiedpermissions list-identity-sources \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "identitySources": [
        {
            "createdDate": "2023-06-12T22:27:49.150035+00:00",
            "details": {
                "clientIds": [ "a1b2c3d4e5f6g7h8i9j0kalbmc" ],
                "discoveryUrl": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_1a2b3c4d5",
                "openIdIssuer": "COGNITO",
                "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/
us-west-2_1a2b3c4d5"
            },
            "identitySourceId": "ISEXAMPLEabcdefg111111",
            "lastUpdatedDate": "2023-06-12T22:27:49.150035+00:00",
            "policyStoreId": "PSEXAMPLEabcdefg111111",
            "principalEntityType": "User"
        }
    ]
}
```

有关身份源的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIdentitySources。

list-policies

以下代码示例演示了如何使用 list-policies。

Amazon CLI

列出可用策略

以下 list-policies 示例列出了指定策略存储中的所有策略。

```
aws verifiedpermissions list-policies \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "policies": [
        {
            "createdDate": "2023-06-12T20:33:37.382907+00:00",
            "definition": {
                "static": {
                    "description": "Grant everyone of janeFriends UserGroup access
 to the vacationFolder Album"
                }
            },
            "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
            "policyId": "SPEXAMPLEabcdefg111111",
            "policyStoreId": "PSEXAMPLEabcdefg111111",
            "policyType": "STATIC",
            "principal": {
                "entityId": "janeFriends",
                "entityType": "UserGroup"
            },
            "resource": {
                "entityId": "vacationFolder",
                "entityType": "Album"
            }
        },
        {
            "createdDate": "2023-06-12T20:39:44.975897+00:00",
            "definition": {
                "static": {
                    "description": "Grant everyone access to the publicFolder Album"
                }
            },
            "lastUpdatedDate": "2023-06-12T20:39:44.975897+00:00",
            "policyId": "SPEXAMPLEabcdefg222222",
```

```
            "policyStoreId": "PSEXAMPLEabcdefg111111",
            "policyType": "STATIC",
            "resource": {
                "entityId": "publicFolder",
                "entityType": "Album"
            }
        },
        {
            "createdDate": "2023-06-12T20:49:51.490211+00:00",
            "definition": {
                "templateLinked": {
                    "policyTemplateId": "PTEXAMPLEabcdefg111111"
                }
            },
            "lastUpdatedDate": "2023-06-12T20:49:51.490211+00:00",
            "policyId": "SPEXAMPLEabcdefg333333",
            "policyStoreId": "PSEXAMPLEabcdefg111111",
            "policyType": "TEMPLATE_LINKED",
            "principal": {
                "entityId": "alice",
                "entityType": "User"
            },
            "resource": {
                "entityId": "VacationPhoto94.jpg",
                "entityType": "Photo"
            }
        }
    ]
}
```

有关策略的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicies。

`list-policy-stores`

以下代码示例演示了如何使用 `list-policy-stores`。

Amazon CLI

列出可用策略存储

以下 `list-policy-stores` 示例列出 Amazon 区域中的所有策略存储。除了 `create-policy-store` 和 `list-policy-stores` 之外，所有 Verified Permissions 命令都要求您指定您要使用的策略存储的 ID。

```
aws verifiedpermissions list-policy-stores
```

输出：

```
{
    "policyStores": [
        {
            "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
            "createdDate": "2023-06-05T20:16:46.225598+00:00",
            "policyStoreId": "PSEXAMPLEabcdefg111111"
        },
        {
            "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg222222",
            "createdDate": "2023-06-08T18:09:37.364356+00:00",
            "policyStoreId": "PSEXAMPLEabcdefg222222"
        },
        {
            "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg333333",
            "createdDate": "2023-06-08T18:09:46.920600+00:00",
            "policyStoreId": "PSEXAMPLEabcdefg333333"
        }
    ]
}
```

有关策略存储的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyStores。

`list-policy-templates`

以下代码示例演示了如何使用 `list-policy-templates`。

Amazon CLI

列出可用策略模板

以下 `list-policy-templates` 示例列出了指定策略存储中的所有策略模板。

```
aws verifiedpermissions list-policy-templates \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "policyTemplates": [
        {
            "createdDate": "2023-06-12T20:47:42.804511+00:00",
            "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
            "policyStoreId": "PSEXAMPLEabcdefg111111",
            "policyTemplateId": "PTEXAMPLEabcdefg111111"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListPolicyTemplates。

put-schema

以下代码示例演示了如何使用 `put-schema`。

Amazon CLI

将架构保存到策略存储

以下 `put-schema` 示例创建或替换指定策略存储中的架构。

输入文件中的 `cedarJson` 参数采用 JSON 对象的字符串表示形式。它在最外面的引号对中包含嵌入式引号（"）。这要求您将 JSON 转换为字符串，方法是在所有嵌入的引号前面加上反斜杠字符（"），然后将所有行合并为一个不带换行符的文本行。

为了便于阅读，可以在此处将字符串示例分成多行显示，但该操作要求将参数作为单行字符串提交。

aws verifiedpermissions put-schema --definition file://schema.txt --policy-store-id PSEXAMPLEabcdefg111111

的内容schema.txt：

```
{
    "cedarJson": "{\"MySampleNamespace\": {\"actions\": {\"remoteAccess\": {
            \"appliesTo\": {\"principalTypes\": [\"Employee\"]}}},\"entityTypes\": {
            \"Employee\": {\"shape\": {\"attributes\": {\"jobLevel\": {\"type\":
            \"Long\"},\"name\": {\"type\": \"String\"}},\"type\": \"Record\"}}}}}"
}
```

输出：

```
{
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "namespaces": [
        "MySampleNamespace"
    ],
    "createdDate": "2023-06-14T17:47:13.999885+00:00",
    "lastUpdatedDate": "2023-06-14T17:47:13.999885+00:00"
}
```

有关架构的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的策略存储架构。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutSchema。

update-identity-source

以下代码示例演示了如何使用 update-identity-source。

Amazon CLI

更新身份源

以下 update-identity-source 示例通过提供新的 Cognito 用户池配置并更改身份源返回的实体类型，来修改指定的身份源。

```
aws verifiedpermissions update-identity-source
    --identity-source-id ISEXAMPLEabcdefg111111 \
    --update-configuration file://config.txt \
    --principal-entity-type "Employee" \
```

```
    --policy-store-id PSEXAMPLEabcdefg111111
```

的内容config.txt：

```
{
        "cognitoUserPoolConfiguration": {
                "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/
us-west-2_1a2b3c4d5",
                "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"]
        }
}
```

输出：

```
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

有关身份源的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的结合使用 Amazon Verified Permissions 与身份提供者。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIdentitySource。

update-policy-store

以下代码示例演示了如何使用 update-policy-store。

Amazon CLI

更新策略存储

以下 update-policy-store 示例通过更改其验证设置来修改策略存储。

```
aws verifiedpermissions update-policy-store \
    --validation-settings "mode=STRICT" \
    --policy-store-id PSEXAMPLEabcdefg111111
```

输出：

```
{
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-05-16T17:41:29.103459+00:00",
    "lastUpdatedDate": "2023-05-16T17:41:29.103459+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

有关策略存储的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略存储。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePolicyStore。

update-policy-template

以下代码示例演示了如何使用 update-policy-template。

Amazon CLI

示例 1：更新策略模板

以下 update-policy-template 示例修改指定的模板链接策略以替换其策略声明。

```
aws verifiedpermissions update-policy-template \
    --policy-template-id PTEXAMPLEabcdefg111111 \
    --statement file://template1.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

template1.txt 文件的内容：

```
permit(
    principal in ?principal,
    action == Action::"view",
    resource == Photo::"VacationPhoto94.jpg"
);
```

输出：

```
{
    "createdDate": "2023-06-12T20:47:42.804511+00:00",
    "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
```

```
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
}
```

有关更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略模板。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePolicyTemplate。

update-policy

以下代码示例演示了如何使用 update-policy。

Amazon CLI

更新静态策略

以下 update-policy 示例通过更新现有静态策略的描述和语句对该策略进行修改。

```
aws verifiedpermissions update-policy \
    --policy-id SPEXAMPLEabcdefg111111 \
    --definition file://updated-definition.txt \
    --policy-store-id PSEXAMPLEabcdefg111111
```

statement 参数采用 JSON 对象的字符串表示形式。它在最外面的引号对中包含嵌入式引号（"）。这要求您将 JSON 转换为字符串，方法是在所有嵌入的引号前面加上反斜杠字符（"），然后将所有行合并为一个不带换行符的文本行。

为了便于阅读，可以将示例字符串分成多行显示，但该操作要求将参数作为单行字符串提交。

updated-definition.txt 文件的内容：

```
{
    "static": {
        "description": "Updated policy to grant janeFriends UserGroup access to the
 vacationFolder Album with view action only",
        "statement": "permit(principal in UserGroup::\"janeFriends\", action ==
 Action::\"view\", resource in Album::\"vacationFolder\" );"
    }
}
```

输出：

```
{
    "createdDate": "2023-06-12T20:33:37.382907+00:00",
    "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityId": "janeFriends",
        "entityType": "UserGroup"
    },
    "resource": {
        "entityId": "vacationFolder",
        "entityType": "Album"
    }
}
```

有关策略的更多信息，请参阅《Amazon Verified Permissions 用户指南》中的 Amazon Verified Permissions 策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePolicy。

使用 Amazon CLI 的 VPC Lattice 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 VPC Lattice 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-listener`

以下代码示例演示了如何使用 `create-listener`。

Amazon CLI

创建侦听器

以下 create-listener 示例创建一个 HTTPS 侦听器，该侦听器具有将流量转发到指定 VPC Lattice 目标组的默认规则。

```
aws vpc-lattice create-listener \
    --name my-service-listener \
    --protocol HTTPS \
    --port 443 \
    --service-identifier svc-0285b53b2eEXAMPLE \
    --default-action file://listener-config.json
```

的内容listener-config.json：

```
{
    "forward": {
        "targetGroups": [
            {
                "targetGroupIdentifier": "tg-0eaa4b9ab4EXAMPLE"
            }
        ]
    }
}
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE/listener/listener-07cc7fb0abEXAMPLE",
    "defaultAction": {
        "forward": {
            "targetGroups": [
                {
                    "targetGroupIdentifier": "tg-0eaa4b9ab4EXAMPLE",
                    "weight": 100
                }
            ]
        }
    },
    "id": "listener-07cc7fb0abEXAMPLE",
    "name": "my-service-listener",
```

```
    "port": 443,
    "protocol": "HTTPS",
    "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "serviceId": "svc-0285b53b2eEXAMPLE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateListener。

create-resource-configuration

以下代码示例演示了如何使用 create-resource-configuration。

Amazon CLI

创建资源配置

以下 create-resource-configuration 示例创建了一个指定单个 IPv4 地址的资源配置。

```
aws vpc-lattice create-resource-configuration \
    --name my-resource-config \
    --type SINGLE \
    --resource-gateway-identifier rgw-0bba03f3d56060135 \
    --resource-configuration-definition 'ipResource={ipAddress=10.0.14.85}'
```

输出：

```
{
    "allowAssociationToShareableServiceNetwork": true,
    "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourceconfiguration/
rcfg-07129f3acded87625",
    "id": "rcfg-07129f3acded87625",
    "name": "my-resource-config",
    "portRanges": [
        "1-65535"
    ],
    "protocol": "TCP",
    "resourceConfigurationDefinition": {
        "ipResource": {
            "ipAddress": "10.0.14.85"
        }
```

```
    },
    "resourceGatewayId": "rgw-0bba03f3d56060135",
    "status": "ACTIVE",
    "type": "SINGLE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource configurations for VPC resources。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateResourceConfiguration。

`create-resource-gateway`

以下代码示例演示了如何使用 `create-resource-gateway`。

Amazon CLI

创建资源网关

以下 `create-resource-gateway` 示例为指定的子网创建资源网关。

```
aws vpc-lattice create-resource-gateway \
    --name my-resource-gateway \
    --vpc-identifier vpc-0bf4c2739bc05a69 \
    --subnet-ids subnet-08e8943905b63a683
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourcegateway/
rgw-0bba03f3d56060135",
    "id": "rgw-0bba03f3d56060135",
    "ipAddressType": "IPV4",
    "name": "my-resource-gateway",
    "securityGroupIds": [
        "sg-087ffd596c5fe962c"
    ],
    "status": "ACTIVE",
    "subnetIds": [
        "subnet-08e8943905b63a683"
    ],
    "vpcIdentifier": "vpc-0bf4c2739bc05a694"
```

```
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 CreateResourceGateway。

`create-service-network-service-association`

以下代码示例演示了如何使用 `create-service-network-service-association`。

Amazon CLI

创建服务关联

以下 `create-service-network-service-association` 示例将指定服务与指定服务网络相关联。

```
aws vpc-lattice create-service-network-service-association \
    --service-identifier svc-0285b53b2eEXAMPLE \
    --service-network-identifier sn-080ec7dc93EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-
east-2:123456789012:servicenetworkserviceassociation/snsa-0e16955a8cEXAMPLE",
    "createdBy": "123456789012",
    "dnsEntry": {
        "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-
svcs.us-east-2.on.aws",
        "hostedZoneId": "Z09127221KTH2CEXAMPLE"
    },
    "id": "snsa-0e16955a8cEXAMPLE",
    "status": "CREATE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理服务关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceNetworkServiceAssociation。

create-service-network-vpc-association

以下代码示例演示了如何使用 create-service-network-vpc-association。

Amazon CLI

创建 VPC 关联

以下 create-service-network-vpc-association 示例将指定 VPC 与指定服务网络相关联。指定的安全组控制 VPC 中的哪些资源可以访问服务网络及其服务。

```
aws vpc-lattice create-service-network-vpc-association \
    --vpc-identifier vpc-0a1b2c3d4eEXAMPLE \
    --service-network-identifier sn-080ec7dc93EXAMPLE \
    --security-group-ids sg-0aee16bc6cEXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/
snva-0821fc8631EXAMPLE",
    "createdBy": "123456789012",
    "id": "snva-0821fc8631EXAMPLE",
    "securityGroupIds": [
        "sg-0aee16bc6cEXAMPLE"
    ],
    "status": "CREATE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceNetworkVpcAssociation。

create-service-network

以下代码示例演示了如何使用 create-service-network。

Amazon CLI

创建服务网络

以下 create-service-network 示例创建一个具有指定名称的服务网络。

```
aws vpc-lattice create-service-network \
    --name my-service-network
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "authType": "NONE",
    "id": "sn-080ec7dc93EXAMPLE",
    "name": "my-service-network"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateServiceNetwork。

create-service

以下代码示例演示了如何使用 create-service。

Amazon CLI

创建服务

以下 create-service 示例创建一项具有指定名称的服务。

```
aws vpc-lattice create-service \
    --name my-lattice-service
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "authType": "NONE",
    "dnsEntry": {
        "domainName": "my-lattice-service-0285b53b2eEXAMPLE.1a2b3c4.vpc-lattice-
svcs.us-east-2.on.aws",
```

```
        "hostedZoneId": "Z09127221KTH2CEXAMPLE"
    },
    "id": "svc-0285b53b2eEXAMPLE",
    "name": "my-lattice-service",
    "status": "CREATE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的 VPC Lattice 中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateService。

create-target-group

以下代码示例演示了如何使用 create-target-group。

Amazon CLI

示例 1：创建 INSTANCE 类型的目标组

以下 create-target-group 示例使用指定的名称、类型和配置创建目标组。

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-instance \
    --type INSTANCE \
    --config file://tg-config.json
```

的内容tg-config.json：

```
{
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
```

```
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-instance",
    "status": "CREATE_IN_PROGRESS",
    "type": "INSTANCE"
}
```

示例 2：创建 IP 类型的目标组

以下 `create-target-group` 示例使用指定的名称、类型和配置创建目标组。

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-ip \
    --type IP \
    --config file://tg-config.json
```

的内容`tg-config.json`：

```
{
    "ipAddressType": "IPV4",
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "ipAddressType": "IPV4",
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
}
```

示例 3：创建 LAMBDA 类型的目标组

以下 `create-target-group` 示例使用指定的名称、类型和配置创建目标组。

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-lambda \
    --type LAMBDA
```

输出：

```
{
```

```
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-lambda",
    "status": "CREATE_IN_PROGRESS",
    "type": "LAMBDA"
}
```

示例 4：创建 ALB 类型的目标组

以下 create-target-group 示例使用指定的名称、类型和配置创建目标组。

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-alb \
    --type ALB \
    --config file://tg-config.json
```

的内容tg-config.json：

```
{
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-alb",
    "status": "CREATE_IN_PROGRESS",
    "type": "ALB"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTargetGroup。

delete-auth-policy

以下代码示例演示了如何使用 delete-auth-policy。

Amazon CLI

删除身份验证策略

以下 delete-auth-policy 示例删除指定服务的身份验证策略。

```
aws vpc-lattice delete-auth-policy \
    --resource-identifier svc-0285b53b2eEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的身份验证策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAuthPolicy。

delete-listener

以下代码示例演示了如何使用 delete-listener。

Amazon CLI

删除侦听器

以下 delete-listener 示例删除指定的侦听器。

```
aws vpc-lattice delete-listener \
    --listener-identifier listener-07cc7fb0abEXAMPLE \
    --service-identifier svc-0285b53b2eEXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的侦听器。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteListener。

delete-resource-configuration

以下代码示例演示了如何使用 delete-resource-configuration。

Amazon CLI

删除资源配置

以下 delete-resource-configuration 示例删除指定的资源配置。

```
aws vpc-lattice delete-resource-configuration \
    --resource-configuration-identifier rcfg-07129f3acded87625
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteResourceConfiguration。

delete-resource-gateway

以下代码示例演示了如何使用 delete-resource-gateway。

Amazon CLI

删除资源网关

以下 delete-resource-gateway 示例删除指定的资源网关。

```
aws vpc-lattice delete-resource-gateway \
    --resource-gateway-identifier rgw-0bba03f3d56060135
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourcegateway/
rgw-0bba03f3d56060135",
    "id": "rgw-0bba03f3d56060135",
    "name": "my-resource-gateway",
    "status": "DELETE_IN_PROGRESS"
```

```
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 DeleteResourceGateway。

delete-service-network-service-association

以下代码示例演示了如何使用 delete-service-network-service-association。

Amazon CLI

删除服务关联

以下 delete-service-network-service-association 示例取消关联指定的服务关联。

```
aws vpc-lattice delete-service-network-service-association \
    --service-network-service-association-identifier snsa-031fabb4d8EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-
east-2:123456789012:servicenetworkserviceassociation/snsa-031fabb4d8EXAMPLE",
    "id": "snsa-031fabb4d8EXAMPLE",
    "status": "DELETE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理服务关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceNetworkServiceAssociation。

delete-service-network-vpc-association

以下代码示例演示了如何使用 delete-service-network-vpc-association。

Amazon CLI

删除 VPC 关联

以下 delete-service-network-vpc-association 示例取消关联指定的 VPC 关联。

```
aws vpc-lattice delete-service-network-vpc-association \
    --service-network-vpc-association-identifier snva-0821fc8631EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/
snva-0821fc8631EXAMPLE",
    "id": "snva-0821fc8631EXAMPLE",
    "status": "DELETE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceNetworkVpcAssociation。

delete-service-network

以下代码示例演示了如何使用 delete-service-network。

Amazon CLI

删除服务网络

以下 delete-service-network 示例删除指定的服务网络。

```
aws vpc-lattice delete-service-network \
    --service-network-identifier sn-080ec7dc93EXAMPLE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteServiceNetwork。

delete-service

以下代码示例演示了如何使用 delete-service。

Amazon CLI

删除服务

以下 delete-service 示例删除指定的服务。

```
aws vpc-lattice delete-service \
    --service-identifier svc-0285b53b2eEXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-west-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "id": "svc-0285b53b2eEXAMPLE",
    "name": "my-lattice-service",
    "status": "DELETE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的 VPC Lattice 中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteService。

delete-target-group

以下代码示例演示了如何使用 delete-target-group。

Amazon CLI

删除目标组

以下 delete-target-group 示例删除指定的目标组。

```
aws vpc-lattice delete-target-group \
    --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
```

```
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "status": "DELETE_IN_PROGRESS"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTargetGroup。

deregister-targets

以下代码示例演示了如何使用 deregister-targets。

Amazon CLI

取消注册目标

以下 deregister-targets 示例从指定目标组中取消注册指定目标。

```
aws vpc-lattice deregister-targets \
    --targets i-07dd579bc5EXAMPLE \
    --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

输出：

```
{
    "successful": [
        {
            "id": "i-07dd579bc5EXAMPLE",
            "port": 443
        }
    ],
    "unsuccessful": []
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的注册目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterTargets。

get-auth-policy

以下代码示例演示了如何使用 get-auth-policy。

Amazon CLI

获取有关身份验证策略的信息

以下 get-auth-policy 示例获取有关指定服务的身份验证策略的信息。

```
aws vpc-lattice get-auth-policy \
    --resource-identifier svc-0285b53b2eEXAMPLE
```

输出：

```
{
    "createdAt": "2023-06-07T03:51:20.266Z",
    "lastUpdatedAt": "2023-06-07T04:39:27.082Z",
    "policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow
\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:role/my-clients\"},
\"Action\":\"vpc-lattice-svcs:Invoke\",\"Resource\":\"arn:aws:vpc-lattice:us-
east-2:123456789012:service/svc-0285b53b2eEXAMPLE\"}]}",
    "state": "Active"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的身份验证策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAuthPolicy。

get-listener

以下代码示例演示了如何使用 get-listener。

Amazon CLI

获取有关服务侦听器的信息

以下 get-listener 示例获取有关指定服务的指定侦听器的信息。

```
aws vpc-lattice get-listener \
    --listener-identifier listener-0ccf55918cEXAMPLE \
    --service-identifier svc-0285b53b2eEXAMPLE
```

输出：

```
{
```

```
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE/listener/listener-0ccf55918cEXAMPLE",
    "createdAt": "2023-05-07T05:08:45.192Z",
    "defaultAction": {
        "forward": {
            "targetGroups": [
                {
                    "targetGroupIdentifier": "tg-0ff213abb6EXAMPLE",
                    "weight": 1
                }
            ]
        }
    },
    "id": "listener-0ccf55918cEXAMPLE",
    "lastUpdatedAt": "2023-05-07T05:08:45.192Z",
    "name": "http-80",
    "port": 80,
    "protocol": "HTTP",
    "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "serviceId": "svc-0285b53b2eEXAMPLE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的定义路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetListener。

get-resource-configuration

以下代码示例演示了如何使用 get-resource-configuration。

Amazon CLI

获取有关资源配置的信息

以下 get-resource-configuration 示例获取有关指定的资源配置的信息。

```
aws vpc-lattice get-resource-configuration \
    --resource-configuration-identifier rcfg-07129f3acded87625
```

输出：

```
{
```

```
    "allowAssociationToShareableServiceNetwork": true,
    "amazonManaged": false,
    "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourceconfiguration/
rcfg-07129f3acded87625",
    "createdAt": "2025-02-01T00:57:35.871000+00:00",
    "id": "rcfg-07129f3acded87625",
    "lastUpdatedAt": "2025-02-01T00:57:46.874000+00:00",
    "name": "my-resource-config",
    "portRanges": [
        "1-65535"
    ],
    "protocol": "TCP",
    "resourceConfigurationDefinition": {
        "ipResource": {
            "ipAddress": "10.0.14.85"
        }
    },
    "resourceGatewayId": "rgw-0bba03f3d56060135",
    "status": "ACTIVE",
    "type": "SINGLE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetResourceConfiguration。

get-resource-gateway

以下代码示例演示了如何使用 get-resource-gateway。

Amazon CLI

获取有关资源网关的信息

以下 get-resource-gateway 示例获取有关指定的资源网关的信息。

```
aws vpc-lattice get-resource-gateway \
    --resource-gateway-identifier rgw-0bba03f3d56060135
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourcegateway/
rgw-0bba03f3d56060135",
    "createdAt": "2025-02-01T00:57:33.241000+00:00",
    "id": "rgw-0bba03f3d56060135",
    "ipAddressType": "IPV4",
    "lastUpdatedAt": "2025-02-01T00:57:44.351000+00:00",
    "name": "my-resource-gateway",
    "securityGroupIds": [
        "sg-087ffd596c5fe962c"
    ],
    "status": "ACTIVE",
    "subnetIds": [
        "subnet-08e8943905b63a683"
    ],
    "vpcId": "vpc-0bf4c2739bc05a694"
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 GetResourceGateway。

get-service-network-service-association

以下代码示例演示了如何使用 get-service-network-service-association。

Amazon CLI

获取有关服务关联的信息

以下 get-service-network-service-association 示例获取有关指定服务关联的信息。

```
aws vpc-lattice get-service-network-service-association \
    --service-network-service-association-identifier snsa-031fabb4d8EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-
east-2:123456789012:servicenetworkserviceassociation/snsa-031fabb4d8EXAMPLE",
```

```
    "createdAt": "2023-05-05T21:48:16.076Z",
    "createdBy": "123456789012",
    "dnsEntry": {
        "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-
svcs.us-east-2.on.aws",
        "hostedZoneId": "Z09127221KTH2CEXAMPLE"
    },
    "id": "snsa-031fabb4d8EXAMPLE",
    "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "serviceId": "svc-0285b53b2eEXAMPLE",
    "serviceName": "my-lattice-service",
    "serviceNetworkArn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "serviceNetworkId": "sn-080ec7dc93EXAMPLE",
    "serviceNetworkName": "my-service-network",
    "status": "ACTIVE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理服务关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceNetworkServiceAssociation。

get-service-network-vpc-association

以下代码示例演示了如何使用 get-service-network-vpc-association。

Amazon CLI

获取有关 VPC 关联的信息

以下 get-service-network-vpc-association 示例获取有关指定 VPC 关联的信息。

```
aws vpc-lattice get-service-network-vpc-association \
    --service-network-vpc-association-identifier snva-0821fc8631EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/
snva-0821fc8631EXAMPLE",
```

```
    "createdAt": "2023-06-06T23:41:08.421Z",
    "createdBy": "123456789012",
    "id": "snva-0c5dcb60d6EXAMPLE",
    "lastUpdatedAt": "2023-06-06T23:41:08.421Z",
    "securityGroupIds": [
        "sg-0aee16bc6cEXAMPLE"
    ],
    "serviceNetworkArn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "serviceNetworkId": "sn-080ec7dc93EXAMPLE",
    "serviceNetworkName": "my-service-network",
    "status": "ACTIVE",
    "vpcId": "vpc-0a1b2c3d4eEXAMPLE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceNetworkVpcAssociation。

get-service-network

以下代码示例演示了如何使用 get-service-network。

Amazon CLI

获取有关服务网络的信息

以下 get-service-network 示例获取有关指定服务网络的信息。

```
aws vpc-lattice get-service-network \
    --service-network-identifier sn-080ec7dc93EXAMPLE
```

输出：

```
{
   "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
   "authType": "AWS_IAM",
   "createdAt": "2023-05-05T15:26:08.417Z",
   "id": "sn-080ec7dc93EXAMPLE",
   "lastUpdatedAt": "2023-05-05T15:26:08.417Z",
```

```
    "name": "my-service-network",
    "numberOfAssociatedServices": 2,
    "numberOfAssociatedVPCs": 3
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceNetwork。

get-service

以下代码示例演示了如何使用 get-service。

Amazon CLI

获取有关服务的信息

以下 get-service 示例获取有关指定服务的信息。

```
aws vpc-lattice get-service \
    --service-identifier svc-0285b53b2eEXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "authType": "AWS_IAM",
    "createdAt": "2023-05-05T21:35:29.339Z",
    "dnsEntry": {
        "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-
svcs.us-east-2.on.aws",
        "hostedZoneId": "Z09127221KTH2CFUOHIZH"
    },
    "id": "svc-0285b53b2eEXAMPLE",
    "lastUpdatedAt": "2023-05-05T21:35:29.339Z",
    "name": "my-lattice-service",
    "status": "ACTIVE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetService。

get-target-group

以下代码示例演示了如何使用 get-target-group。

Amazon CLI

获取有关目标组的信息

以下 get-target-group 示例获取有关目标类型为 INSTANCE 的指定目标组的信息。

```
aws vpc-lattice get-target-group \
    --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

输出：

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "createdAt": "2023-05-06T04:41:04.122Z",
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "lastUpdatedAt": "2023-05-06T04:41:04.122Z",
    "name": "my-target-group",
```

```
    "serviceArns": [
        "arn:aws:vpc-lattice:us-east-2:123456789012:service/svc-0285b53b2eEXAMPLE"
    ],
    "status": "ACTIVE",
    "type": "INSTANCE"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTargetGroup。

`list-listeners`

以下代码示例演示了如何使用 `list-listeners`。

Amazon CLI

列出服务侦听器

以下 `list-listeners` 示例列出指定服务的侦听器。

```
aws vpc-lattice list-listeners \
    --service-identifier svc-0285b53b2eEXAMPLE
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE/listener/listener-0ccf55918cEXAMPLE",
            "createdAt": "2023-05-07T05:08:45.192Z",
            "id": "listener-0ccf55918cEXAMPLE",
            "lastUpdatedAt": "2023-05-07T05:08:45.192Z",
            "name": "http-80",
            "port": 80,
            "protocol": "HTTP"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的定义路由。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListListeners。

`list-resource-configurations`

以下代码示例演示了如何使用 `list-resource-configurations`。

Amazon CLI

列出资源配置

以下 `list-resource-configurations` 示例列出了资源配置。

```
aws vpc-lattice list-resource-configurations
```

输出：

```
{
    "items": [
        {
            "amazonManaged": false,
            "arn": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-07129f3acded87625",
            "createdAt": "2025-02-01T00:57:35.871000+00:00",
            "id": "rcfg-07129f3acded87625",
            "lastUpdatedAt": "2025-02-01T00:57:46.874000+00:00",
            "name": "my-resource-config",
            "resourceGatewayId": "rgw-0bba03f3d56060135",
            "status": "ACTIVE",
            "type": "SINGLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource configurations。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceConfigurations。

`list-resource-endpoint-associations`

以下代码示例演示了如何使用 `list-resource-endpoint-associations`。

Amazon CLI

列出 VPC 端点关联

以下 `list-resource-endpoint-associations` 示例列出与指定的资源配置关联的 VPC 端点。

```
aws vpc-lattice list-resource-endpoint-associations \
    --resource-configuration-identifier rcfg-07129f3acded87625
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceendpointassociation/rea-0956a7435baf89326",
            "createdAt": "2025-02-01T00:57:38.998000+00:00",
            "id": "rea-0956a7435baf89326",
            "resourceConfigurationArn": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-07129f3acded87625",
            "resourceConfigurationId": "rcfg-07129f3acded87625",
            "vpcEndpointId": "vpce-019b90d6f16d4f958",
            "vpcEndpointOwner": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Manage associations for a VPC Lattice resource configuration。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceEndpointAssociations。

`list-resource-gateways`

以下代码示例演示了如何使用 `list-resource-gateways`。

Amazon CLI

列出资源网关

以下 `list-resource-gateways` 示例列出了资源网关。

```
aws vpc-lattice list-resource-gateways
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:vpc-lattice:us-east-1:123456789012:resourcegateway/
rgw-0bba03f3d56060135",
            "createdAt": "2025-02-01T00:57:33.241000+00:00",
            "id": "rgw-0bba03f3d56060135",
            "ipAddressType": "IPV4",
            "lastUpdatedAt": "2025-02-01T00:57:44.351000+00:00",
            "name": "my-resource-gateway",
            "seurityGroupIds": [
                "sg-087ffd596c5fe962c"
            ],
            "status": "ACTIVE",
            "subnetIds": [
                "subnet-08e8943905b63a683"
            ],
            "vpcIdentifier": "vpc-0bf4c2739bc05a694"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Resource gateways in VPC Lattice。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListResourceGateways。

`list-service-network-service-associations`

以下代码示例演示了如何使用 `list-service-network-service-associations`。

Amazon CLI

列出服务关联

以下 `list-service-network-service-associations` 示例列出指定服务网络的服务关联。该 `--query` 选项将输出范围限定为服务关联的 ID。

```
aws vpc-lattice list-service-network-service-associations \
    --service-network-identifier sn-080ec7dc93EXAMPLE \
    --query items[*].id
```

输出：

```
[
    "snsa-031fabb4d8EXAMPLE",
    "snsa-0e16955a8cEXAMPLE"
]
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理服务关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceNetworkServiceAssociations。

list-service-network-vpc-associations

以下代码示例演示了如何使用 list-service-network-vpc-associations。

Amazon CLI

列出 VPC 关联

以下 list-service-network-vpc-associations 示例列出指定服务网络的 VPC 关联。该 --query 选项将输出范围限定为 VPC 关联的 ID。

```
aws vpc-lattice list-service-network-vpc-associations \
    --service-network-identifier sn-080ec7dc93EXAMPLE \
    --query items[*].id
```

输出：

```
[
    "snva-0821fc8631EXAMPLE",
    "snva-0c5dcb60d6EXAMPLE"
]
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的管理 VPC 关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceNetworkVpcAssociations。

`list-service-network-vpc-endpoint-associations`

以下代码示例演示了如何使用 `list-service-network-vpc-endpoint-associations`。

Amazon CLI

列出 VPC 端点关联

以下 `list-service-network-vpc-endpoint-associations` 示例列出了与特定服务网络关联的 VPC 端点。

```
aws vpc-lattice list-service-network-vpc-endpoint-associations \
    --service-network-identifier sn-0808d1748faee0c1e
```

输出：

```
{
    "items": [
        {
            "createdAt": "2025-02-01T01:21:36.667000+00:00",
            "serviceNetworkArn": "arn:aws:vpc-lattice:us-
east-1:123456789012:servicenetwork/sn-0808d1748faee0c1e",
            "state": "ACTIVE",
            "vpcEndpointId": "vpce-0cc199f605eaeace7",
            "vpcEndpointOwnerId": "123456789012"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice User Guide》中的 Manage the associations for a VPC Lattice service network。

- 有关 API 详细信息，请参阅《Amazon CLI Command Reference》中的 ListServiceNetworkVpcEndpointAssociations。

`list-service-networks`

以下代码示例演示了如何使用 `list-service-networks`。

Amazon CLI

列出服务网络

以下 `list-service-networks` 示例列出调用账户拥有或与之共享的服务网络。该 `--query` 选项将结果范围限定为服务网络的 Amazon 资源名称（ARN）。

```
aws vpc-lattice list-service-networks \
    --query items[*].arn
```

输出：

```
[
    "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "arn:aws:vpc-lattice:us-east-2:111122223333:servicenetwork/sn-0ec4d436cfEXAMPLE"
]
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务网络。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServiceNetworks。

list-services

以下代码示例演示了如何使用 `list-services`。

Amazon CLI

列出服务

以下 `list-services` 示例列出调用账户拥有或与之共享的服务。该 `--query` 选项将结果的范围限定为服务的 Amazon 资源名称（ARN）。

```
aws vpc-lattice list-services \
    --query items[*].arn
```

输出：

```
[
    "arn:aws:vpc-lattice:us-east-2:123456789012:service/svc-0285b53b2eEXAMPLE",
    "arn:aws:vpc-lattice:us-east-2:111122223333:service/svc-0b8ac96550EXAMPLE"
```

```
]
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的服务。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListServices。

`list-target-groups`

以下代码示例演示了如何使用 `list-target-groups`。

Amazon CLI

列出目标组

以下 `list-target-groups` 示例列出目标类型为 LAMBDA 的目标组。

```
aws vpc-lattice list-target-groups \
    --target-group-type LAMBDA
```

输出：

```
{
    "items": [
        {
            "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-045c1b7d9dEXAMPLE",
            "createdAt": "2023-05-06T05:22:16.637Z",
            "id": "tg-045c1b7d9dEXAMPLE",
            "lastUpdatedAt": "2023-05-06T05:22:16.637Z",
            "name": "my-target-group-lam",
            "serviceArns": [
                "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE"
            ],
            "status": "ACTIVE",
            "type": "LAMBDA"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargetGroups。

`list-targets`

以下代码示例演示了如何使用 `list-targets`。

Amazon CLI

列出目标组中的目标

以下 `list-targets` 示例列出指定目标组中的目标。

```
aws vpc-lattice list-targets \
    --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

输出：

```
{
    "items": [
        {
            "id": "i-07dd579bc5EXAMPLE",
            "port": 443,
            "status": "HEALTHY"
        },
        {
            "id": "i-047b3c9078EXAMPLE",
            "port": 443,
            "reasonCode": "HealthCheckFailed",
            "status": "UNHEALTHY"
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的目标组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTargets。

`put-auth-policy`

以下代码示例演示了如何使用 `put-auth-policy`。

Amazon CLI

为服务创建身份验证策略

以下 put-auth-policy 示例授予对任何经过身份验证的主体所发出请求的访问权限，该主体使用指定 IAM 角色。该资源是向其中附加了策略的服务的 ARN。

```
aws vpc-lattice put-auth-policy \
    --resource-identifier svc-0285b53b2eEXAMPLE \
    --policy file://auth-policy.json
```

的内容auth-policy.json：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/my-clients"
            },
            "Action": "vpc-lattice-svcs:Invoke",
            "Resource": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE"
        }
    ]
}
```

输出：

```
{
    "policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow
\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:role/my-clients\"},
\"Action\":\"vpc-lattice-svcs:Invoke\",\"Resource\":\"arn:aws:vpc-lattice:us-
east-2:123456789012:service/svc-0285b53b2eEXAMPLE\"}]}",
    "state": "Active"
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的身份验证策略。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAuthPolicy。

register-targets

以下代码示例演示了如何使用 register-targets。

Amazon CLI

注册目标

以下 `register-targets` 示例向指定目标组注册指定目标。

```
aws vpc-lattice register-targets \
    --targets id=i-047b3c9078EXAMPLE id=i-07dd579bc5EXAMPLE \
    --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

输出：

```
{
    "successful": [
        {
            "id": "i-07dd579bc5EXAMPLE",
            "port": 443
        }
    ],
    "unsuccessful": [
        {
            "failureCode": "UnsupportedTarget",
            "failureMessage": "Instance targets must be in the same VPC as their
 target group",
            "id": "i-047b3c9078EXAMPLE",
            "port": 443
        }
    ]
}
```

有关更多信息，请参阅《Amazon VPC Lattice 用户指南》中的注册目标。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterTargets。

Amazon WAF Classic使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon WAF Classic 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

put-logging-configuration

以下代码示例演示了如何使用 put-logging-configuration。

Amazon CLI

使用指定的 Kinesis Firehose 流 ARN 为 Web ACL ARN 创建日志记录配置

以下 put-logging-configuration 示例显示了使用 CloudFront 的 WAF 的日志记录配置。

```
aws waf put-logging-configuration \
    --logging-configuration ResourceArn=arn:aws:waf::123456789012:webacl/3bffd3ed-fa2e-445e-869f-a6a7cf153fd3,LogDestinationConfigs=arn:aws:firehose:us-east-1:123456789012:deliverystream/aws-waf-logs-firehose-stream,RedactedFields=[]
```

输出：

```
{
    "LoggingConfiguration": {
        "ResourceArn": "arn:aws:waf::123456789012:webacl/3bffd3ed-fa2e-445e-869f-a6a7cf153fd3",
        "LogDestinationConfigs": [
            "arn:aws:firehose:us-east-1:123456789012:deliverystream/aws-waf-logs-firehose-stream"
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingConfiguration。

update-byte-match-set

以下代码示例演示了如何使用 update-byte-match-set。

Amazon CLI

更新字节匹配集

以下 update-byte-match-set 命令删除 ByteMatchSet 中的 ByteMatchTuple 对象（筛选条件）：

```
aws waf update-byte-match-set --byte-match-set-id a123fae4-
b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --
updates
 Action="DELETE",ByteMatchTuple={FieldToMatch={Type="HEADER",Data="referer"},TargetString="b
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用字符串匹配条件”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateByteMatchSet。

update-ip-set

以下代码示例演示了如何使用 update-ip-set。

Amazon CLI

更新 IP 集

以下 update-ip-set 命令使用 IPv4 地址更新 IPSet 并删除 IPv6 地址：

```
aws waf update-ip-set --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90
 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
 Action="INSERT",IPSetDescriptor={Type="IPV4",Value="12.34.56.78/16"},Action="DELETE",IPSetD
```

或者，也可以使用 JSON 文件来指定输入。例如：

```
aws waf update-ip-set --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --change-
token 12cs345-67cd-890b-1cd2-c3a4567d89f1  --updates file://change.json
```

以下是 JSON 文件的内容：

```
[
{
"Action": "INSERT",
"IPSetDescriptor":
{
```

```
"Type": "IPV4",
"Value": "12.34.56.78/16"
}
},
{
"Action": "DELETE",
"IPSetDescriptor":
{
"Type": "IPV6",
"Value": "1111:0000:0000:0000:0000:0000:0000:0111/128"
}
}
]
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用 IP 匹配条件”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIpSet。

update-rule

以下代码示例演示了如何使用 update-rule。

Amazon CLI

更新规则

以下 update-rule 命令删除规则中的 Predicate 对象：

```
aws waf update-rule --rule-id a123fae4-b567-8e90-1234-5ab67ac8ca90
 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
 Action="DELETE",Predicate={Negated=false,Type="ByteMatch",DataId="MyByteMatchSetID"}
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用规则”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRule。

update-size-constraint-set

以下代码示例演示了如何使用 update-size-constraint-set。

Amazon CLI

更新大小约束集

以下 update-size-constraint-set 命令删除大小约束集中的 SizeConstraint 对象（筛选条件）：

```
aws waf update-size-constraint-set --size-constraint-set-id a123fae4-
b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --
updates
 Action="DELETE",SizeConstraint={FieldToMatch={Type="QUERY_STRING"},TextTransformation="NONE
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用大小约束条件”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSizeConstraintSet。

update-sql-injection-match-set

以下代码示例演示了如何使用 update-sql-injection-match-set。

Amazon CLI

更新 SQL 注入匹配集

以下 update-sql-injection-match-set 命令删除 SQL 注入匹配集中的 SqlInjectionMatchTuple 对象（筛选条件）：

```
aws waf update-sql-injection-match-set --sql-injection-
match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --
change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
 Action="DELETE",SqlInjectionMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformati
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用 SQL 注入匹配条件”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSqlInjectionMatchSet。

update-web-acl

以下代码示例演示了如何使用 update-web-acl。

Amazon CLI

更新 Web ACL

以下 update-web-acl 命令删除 Web ACL 中的 ActivatedRule 对象。

aws waf update-web-acl --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates Action="DELETE",ActivatedRule='{Priority=1,RuleId="WAFRule-1-Example",Action={Type="ALLOW"},Type="REGULAR"}'

输出：

```
{
    "ChangeToken": "12cs345-67cd-890b-1cd2-c3a4567d89f1"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的使用 Web ACL。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWebAcl。

update-xss-match-set

以下代码示例演示了如何使用 update-xss-match-set。

Amazon CLI

更新 XSSMatchSet

以下 update-xss-match-set 命令删除 XssMatchSet 中的 XssMatchTuple 对象（筛选条件）：

```
aws waf update-xss-match-set --xss-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90
 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
 Action="DELETE",XssMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation="URL_D
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的“使用跨站点脚本匹配条件”。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateXssMatchSet。

Amazon WAF Classic Regional使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon WAF Classic Regional 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-web-acl`

以下代码示例演示了如何使用 `associate-web-acl`。

Amazon CLI

将 Web ACL 与资源相关联

以下 `associate-web-acl` 命令将由 web-acl-id 指定的 Web ACL 与 resource-arn 指定的资源相关联。资源 ARN 可以指应用程序负载均衡器或 API Gateway：

```
aws waf-regional associate-web-acl \
    --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --resource-arn 12cs345-67cd-890b-1cd2-c3a4567d89f1
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用 Web ACL。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateWebAcl。

`put-logging-configuration`

以下代码示例演示了如何使用 `put-logging-configuration`。

Amazon CLI

使用指定的 Kinesis Firehose 流 ARN 为 Web ACL ARN 创建日志记录配置

以下 `put-logging-configuration` 示例显示 `us-east-1` 区域中具有 ALB/APIGateway 的 WAF 日志记录配置。

```
aws waf-regional put-logging-configuration \
```

```
    --logging-configuration ResourceArn=arn:aws:waf-
regional:us-east-1:123456789012:webacl/3bffd3ed-fa2e-445e-869f-
a6a7cf153fd3,LogDestinationConfigs=arn:aws:firehose:us-
east-1:123456789012:deliverystream/aws-waf-logs-firehose-stream,RedactedFields=[] \
    --region us-east-1
```

输出：

```
{
    "LoggingConfiguration": {
        "ResourceArn": "arn:aws:waf-regional:us-east-1:123456789012:webacl/3bffd3ed-
fa2e-445e-869f-a6a7cf153fd3",
        "LogDestinationConfigs": [
            "arn:aws:firehose:us-east-1:123456789012:deliverystream/aws-waf-logs-
firehose-stream"
        ]
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingConfiguration。

update-byte-match-set

以下代码示例演示了如何使用 update-byte-match-set。

Amazon CLI

更新字节匹配集

以下 update-byte-match-set 命令删除 ByteMatchSet 中的 ByteMatchTuple 对象（筛选条件）。由于该 updates 值嵌入了双引号，因此必须用单引号将该值括起来。

```
aws waf-regional update-byte-match-set \
    --byte-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
    --updates
 'Action="DELETE",ByteMatchTuple={FieldToMatch={Type="HEADER",Data="referer"},TargetString="
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用字符串匹配条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateByteMatchSet。

update-ip-set

以下代码示例演示了如何使用 update-ip-set。

Amazon CLI

更新 IP 集

以下 update-ip-set 命令使用 IPv4 地址更新 IPSet 并删除 IPv6 地址。通过运行 get-change-token 命令获取 change-token 的值。由于更新值包含嵌入式双引号，因此必须用单引号将该值括起来。

```
aws waf update-ip-set \
    --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
    --updates
 'Action="INSERT",IPSetDescriptor={Type="IPV4",Value="12.34.56.78/16"},Action="DELETE",IPSet
```

或者，也可以使用 JSON 文件来指定输入。例如：

```
aws waf-regional update-ip-set \
    --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1  \
    --updates file://change.json
```

change.json 的内容

```
[
    {
        "Action": "INSERT",
        "IPSetDescriptor":
        {
            "Type": "IPV4",
            "Value": "12.34.56.78/16"
        }
    },
    {
        "Action": "DELETE",
        "IPSetDescriptor":
        {
            "Type": "IPV6",
            "Value": "1111:0000:0000:0000:0000:0000:0000:0111/128"
```

```
        }
    }
]
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用 IP 匹配条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIpSet。

update-rule

以下代码示例演示了如何使用 update-rule。

Amazon CLI

更新规则

以下 update-rule 命令删除规则中的 Predicate 对象。由于 updates 值嵌入了双引号，因此必须用单引号将整个值括起来。

```
aws waf-regional update-rule \
    --rule-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
    --updates
 'Action="DELETE",Predicate={Negated=false,Type="ByteMatch",DataId="MyByteMatchSetID"}'
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRule。

update-size-constraint-set

以下代码示例演示了如何使用 update-size-constraint-set。

Amazon CLI

更新大小约束集

以下 update-size-constraint-set 命令删除大小约束集中的 SizeConstraint 对象（筛选条件）。由于 updates 值包含嵌入式双引号，因此必须用单引号将整个值括起来。

```
aws waf-regional update-size-constraint-set \
    --size-constraint-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
```

```
    --updates
'Action="DELETE",SizeConstraint={FieldToMatch={Type="QUERY_STRING"},TextTransformation="NON
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用大小约束条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSizeConstraintSet。

update-sql-injection-match-set

以下代码示例演示了如何使用 update-sql-injection-match-set。

Amazon CLI

更新 SQL 注入匹配集

以下 update-sql-injection-match-set 命令删除 SQL 注入匹配集中的 SqlInjectionMatchTuple 对象（筛选条件）。由于 updates 值包含嵌入式双引号，因此必须用单引号将整个值括起来。

aws waf-regional update-sql-injection-match-set --sql-injection-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates 'Action="DELETE",SqlInjectionMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformatic

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用 SQL 注入匹配条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSqlInjectionMatchSet。

update-web-acl

以下代码示例演示了如何使用 update-web-acl。

Amazon CLI

更新 Web ACL

以下 update-web-acl 命令删除 Web ACL 中的 ActivatedRule 对象。由于 updates 值包含嵌入式双引号，因此必须用单引号将整个值括起来。

```
aws waf-regional update-web-acl \
    --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
```

```
    --updates Action="DELETE",ActivatedRule='{Priority=1,RuleId="WAFRule-1-
Example",Action={Type="ALLOW"},Type="ALLOW"}'
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用 Web ACL。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWebAcl。

update-xss-match-set

以下代码示例演示了如何使用 update-xss-match-set。

Amazon CLI

更新 XSSMatchSet

以下 update-xss-match-set 命令删除 XssMatchSet 中的 XssMatchTuple 对象（筛选条件）。由于 updates 值包含嵌入式双引号，因此必须用单引号将整个值括起来。

```
aws waf-regional update-xss-match-set \
    --xss-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
    --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
    --updates
 'Action="DELETE",XssMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation="URL_
```

有关更多信息，请参阅《Amazon WAF 开发人员指南》中的使用跨站点脚本匹配条件。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateXssMatchSet。

Amazon WAFV2使用 的 示例Amazon CLI

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon WAFV2 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-web-acl`

以下代码示例演示了如何使用 `associate-web-acl`。

Amazon CLI

将 Web ACL 与区域 Amazon 资源相关联

以下 `associate-web-acl` 示例将指定 Web ACL 与应用程序负载均衡器相关联。

```
aws wafv2 associate-web-acl \
    --web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-cli/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/waf-cli-alb/1ea17125f8b25a2a \
    --region us-west-2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的将 Web ACL 与 Amazon 资源关联或取消关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateWebAcl。

`check-capacity`

以下代码示例演示了如何使用 `check-capacity`。

Amazon CLI

获取一组规则所使用的容量

以下 `check-capacity` 命令检索某规则集的容量要求，该规则集包含基于速率的规则声明和包含嵌套规则的 AND 规则声明。

```
aws wafv2 check-capacity \
    --scope REGIONAL \
    --rules file://waf-rule-list.json \
    --region us-west-2
```

file://waf-rule-list.json 的内容：

```
[
    {
        "Name":"basic-rule",
        "Priority":0,
        "Statement":{
            "AndStatement":{
                "Statements":[
                    {
                        "ByteMatchStatement":{
                            "SearchString":"example.com",
                            "FieldToMatch":{
                                "SingleHeader":{
                                    "Name":"host"
                                }
                            },
                            "TextTransformations":[
                                {
                                    "Priority":0,
                                    "Type":"LOWERCASE"
                                }
                            ],
                            "PositionalConstraint":"EXACTLY"
                        }
                    },
                    {
                        "GeoMatchStatement":{
                            "CountryCodes":[
                                "US",
                                "IN"
                            ]
                        }
                    }
                ]
            }
        },
        "Action":{
            "Allow":{

            }
        },
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
```

```
            "CloudWatchMetricsEnabled":true,
            "MetricName":"basic-rule"
        }
    },
    {
        "Name":"rate-rule",
        "Priority":1,
        "Statement":{
            "RateBasedStatement":{
                "Limit":1000,
                "AggregateKeyType":"IP"
            }
        },
        "Action":{
            "Block":{

            }
        },
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
            "CloudWatchMetricsEnabled":true,
            "MetricName":"rate-rule"
        }
    }
]
```

输出：

```
{
    "Capacity":15
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon WAF Web ACL 容量单位（WCU）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CheckCapacity。

create-ip-set

以下代码示例演示了如何使用 create-ip-set。

Amazon CLI

创建用于 Web ACL 和规则组的 IP 集

以下 create-ip-set 命令创建具有单一地址范围规范的 IP 集。

```
aws wafv2 create-ip-set \
    --name testip \
    --scope REGIONAL \
    --ip-address-version IPV4 \
    --addresses 198.51.100.0/16
```

输出：

```
{
    "Summary":{
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Description":"",
        "Name":"testip",
        "LockToken":"447e55ac-0000-0000-0000-86b67c17f8b5",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateIpSet。

create-regex-pattern-set

以下代码示例演示了如何使用 create-regex-pattern-set。

Amazon CLI

创建用于 Web ACL 和规则组的正则表达式模式集

以下 create-regex-pattern-set 命令创建指定了两个正则表达式模式的正则表达式模式集。

```
aws wafv2 create-regex-pattern-set \
    --name regexPatterSet01 \
    --scope REGIONAL \
```

```
    --description 'Test web-acl' \
    --regular-expression-list '[{"RegexString": "/[0-9]*/"},{"RegexString": "/[a-
z]*/"}]'
```

输出：

```
{
    "Summary":{
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Description":"Test web-acl",
        "Name":"regexPatterSet01",
        "LockToken":"0bc01e21-03c9-4b98-9433-6229cbf1ef1c",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRegexPatternSet。

create-rule-group

以下代码示例演示了如何使用 create-rule-group。

Amazon CLI

创建用于 Web ACL 的自定义规则组

以下 create-rule-group 命令创建供区域使用的自定义规则组。该组的规则声明以 JSON 格式的文件提供。

```
aws wafv2 create-rule-group \
    --name "TestRuleGroup" \
    --scope REGIONAL \
    --capacity 250 \
    --rules file://waf-rule.json \
    --visibility-
config SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=TestRuleGroupMet
 \
    --region us-west-2
```

file://waf-rule.json 的内容：

```
[
    {
        "Name":"basic-rule",
        "Priority":0,
        "Statement":{
            "AndStatement":{
                "Statements":[
                    {
                        "ByteMatchStatement":{
                            "SearchString":"example.com",
                            "FieldToMatch":{
                                "SingleHeader":{
                                    "Name":"host"
                                }
                            },
                            "TextTransformations":[
                                {
                                    "Priority":0,
                                    "Type":"LOWERCASE"
                                }
                            ],
                            "PositionalConstraint":"EXACTLY"
                        }
                    },
                    {
                        "GeoMatchStatement":{
                            "CountryCodes":[
                                "US",
                                "IN"
                            ]
                        }
                    }
                ]
            }
        },
        "Action":{
            "Allow":{

            }
        },
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
```

```
            "CloudWatchMetricsEnabled":true,
            "MetricName":"basic-rule"
        }
    }
]
```

输出：

```
{
    "Summary":{
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/
TestRuleGroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Description":"",
        "Name":"TestRuleGroup",
        "LockToken":"7b3bcec2-374e-4c5a-b2b9-563bf47249f0",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理您自己的规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRuleGroup。

create-web-acl

以下代码示例演示了如何使用 create-web-acl。

Amazon CLI

创建 Web ACL

以下 create-web-acl 命令创建供区域使用的 Web ACL。Web ACL 的规则声明在 JSON 格式的文件中提供。

```
aws wafv2 create-web-acl \
    --name TestWebAcl \
    --scope REGIONAL \
    --default-action Allow={} \
    --visibility-
config SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=TestWebAclMetric
 \
```

```
    --rules file://waf-rule.json \
    --region us-west-2
```

file://waf-rule.json 的内容：

```
[
    {
        "Name":"basic-rule",
        "Priority":0,
        "Statement":{
            "AndStatement":{
                "Statements":[
                    {
                        "ByteMatchStatement":{
                            "SearchString":"example.com",
                            "FieldToMatch":{
                                "SingleHeader":{
                                    "Name":"host"
                                }
                            },
                            "TextTransformations":[
                                {
                                    "Priority":0,
                                    "Type":"LOWERCASE"
                                }
                            ],
                            "PositionalConstraint":"EXACTLY"
                        }
                    },
                    {
                        "GeoMatchStatement":{
                            "CountryCodes":[
                                "US",
                                "IN"
                            ]
                        }
                    }
                ]
            }
        },
        "Action":{
            "Allow":{
```

```
            }
        },
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
            "CloudWatchMetricsEnabled":true,
            "MetricName":"basic-rule"
        }
    }
]
```

输出：

```
{
    "Summary":{
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/TestWebAcl/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Description":"",
        "Name":"TestWebAcl",
        "LockToken":"2294b3a1-eb60-4aa0-a86f-a3ae04329de9",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理和使用 Web 访问控制列表（Web ACL）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWebAcl。

delete-ip-set

以下代码示例演示了如何使用 delete-ip-set。

Amazon CLI

删除 IP 集

以下 delete-ip-set 命令删除指定 IP 集。此调用需要一个 ID（可以从调用 list-ip-sets 中获取）和一个锁定令牌（可以从调用 list-ip-sets 和 get-ip-set 中获取）。

```
aws wafv2 delete-ip-set \
    --name test1 \
    --scope REGIONAL \
```

```
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token 46851772-db6f-459d-9385-49428812e357
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteIpSet。

delete-logging-configuration

以下代码示例演示了如何使用 delete-logging-configuration。

Amazon CLI

禁用 Web ACL 日志记录

以下 delete-logging-configuration 命令从指定 Web ACL 中删除任何日志记录配置。

```
aws wafv2 delete-logging-configuration \
    --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的记录 Web ACL 流量日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLoggingConfiguration。

delete-regex-pattern-set

以下代码示例演示了如何使用 delete-regex-pattern-set。

Amazon CLI

删除正则表达式模式集

以下 delete-regex-pattern-set 命令更新指定正则表达式模式集的设置。此调用需要一个 ID（可以从调用 list-regex-pattern-sets 中获取）和一个锁定令牌（可以从调用 list-regex-pattern-sets 或 get-regex-pattern-set 中获取）。

```
aws wafv2 delete-regex-pattern-set \
    --name regexPatterSet01 \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token 0bc01e21-03c9-4b98-9433-6229cbf1ef1c
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRegexPatternSet。

delete-rule-group

以下代码示例演示了如何使用 delete-rule-group。

Amazon CLI

删除自定义规则组

以下 delete-rule-group 命令删除指定的自定义规则组。此调用需要一个 ID（可以从调用 list-rule-groups 中获取）和一个锁定令牌（可以从调用 list-rule-groups 或 get-rule-group 中获取）。

```
aws wafv2 delete-rule-group \
    --name TestRuleGroup \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token 7b3bcec2-0000-0000-0000-563bf47249f0
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理您自己的规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRuleGroup。

delete-web-acl

以下代码示例演示了如何使用 delete-web-acl。

Amazon CLI

删除 Web ACL

以下 `delete-web-acl` 命令从您的账户中删除指定 Web ACL。只有在 Web ACL 不与任何资源关联的情况下才能将其删除。此调用需要一个 ID（可以从调用 `list-web-acls` 中获取）和一个锁定令牌（可以从调用 `list-web-acls` 或 `get-web-acl` 中获取）。

```
aws wafv2 delete-web-acl \
    --name test \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token ebab4ed2-155e-4c9a-9efb-e4c45665b1f5
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理和使用 Web 访问控制列表（Web ACL）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteWebAcl。

describe-managed-rule-group

以下代码示例演示了如何使用 `describe-managed-rule-group`。

Amazon CLI

检索托管规则组的描述

以下 `describe-managed-rule-group` 命令检索 Amazon 托管规则组的相关描述。

```
aws wafv2 describe-managed-rule-group \
    --vendor-name AWS \
    --name AWSManagedRulesCommonRuleSet \
    --scope REGIONAL
```

输出：

```
{
    "Capacity": 700,
    "Rules": [
        {
            "Name": "NoUserAgent_HEADER",
```

```
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "UserAgent_BadBots_HEADER",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "SizeRestrictions_QUERYSTRING",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "SizeRestrictions_Cookie_HEADER",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "SizeRestrictions_BODY",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "SizeRestrictions_URIPATH",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "EC2MetaDataSSRF_BODY",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "EC2MetaDataSSRF_COOKIE",
            "Action": {
                "Block": {}
```

```
            }
        },
        {
            "Name": "EC2MetaDataSSRF_URIPATH",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "EC2MetaDataSSRF_QUERYARGUMENTS",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "GenericLFI_QUERYARGUMENTS",
            "Action": {
                "Block": {}
            }
        },
        {
            }
            "Name": "GenericLFI_URIPATH",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "GenericLFI_BODY",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "RestrictedExtensions_URIPATH",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "RestrictedExtensions_QUERYARGUMENTS",
            "Action": {
                "Block": {}
            }
```

```
        },
        {
            "Name": "GenericRFI_QUERYARGUMENTS",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "GenericRFI_BODY",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "GenericRFI_URIPATH",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "CrossSiteScripting_COOKIE",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "CrossSiteScripting_QUERYARGUMENTS",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "CrossSiteScripting_BODY",
            "Action": {
                "Block": {}
            }
        },
        {
            "Name": "CrossSiteScripting_URIPATH",
            "Action": {
                "Block": {}
            }
        }
    ]
```

```
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的托管规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeManagedRuleGroup。

`disassociate-web-acl`

以下代码示例演示了如何使用 `disassociate-web-acl`。

Amazon CLI

取消 Web ACL 与区域 Amazon 资源的关联

以下 `disassociate-web-acl` 示例从指定应用程序负载均衡器中删除任何现有 Web ACL 关联。

```
aws wafv2 disassociate-web-acl \
    --resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/waf-cli-alb/1ea17125f8b25a2a \
    --region us-west-2
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的将 Web ACL 与 Amazon 资源关联或取消关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateWebAcl。

`get-ip-set`

以下代码示例演示了如何使用 `get-ip-set`。

Amazon CLI

检索特定 IP 集

以下 `get-ip-set` 命令检索具有指定名称、范围和 ID 的 IP 集。可以从命令 `create-ip-set` 和 `list-ip-sets` 中获取 IP 集的 ID。

```
aws wafv2 get-ip-set \
    --name testip \
    --scope REGIONAL \
```

```
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "IPSet":{
        "Description":"",
        "Name":"testip",
        "IPAddressVersion":"IPV4",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE1111",
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE1111",
        "Addresses":[
            "192.0.2.0/16"
        ]
    },
    "LockToken":"447e55ac-2396-4c6d-b9f9-86b67c17f8b5"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetIpSet。

get-logging-configuration

以下代码示例演示了如何使用 get-logging-configuration。

Amazon CLI

检索 Web ACL 的日志记录配置

以下 get-logging-configuration 命令检索指定 Web ACL 的日志记录配置。

```
aws wafv2 get-logging-configuration \
    --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
    --region us-west-2
```

输出：

```
{
```

```
    "LoggingConfiguration":{
        "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "RedactedFields":[
            {
                "Method":{

                }
            }
        ],
        "LogDestinationConfigs":[
            "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-logs-
custom-transformation"
        ]
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的记录 Web ACL 流量日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetLoggingConfiguration。

get-rate-based-statement-managed-keys

以下代码示例演示了如何使用 get-rate-based-statement-managed-keys。

Amazon CLI

检索基于速率的规则阻止的 IP 地址的列表。

以下 get-rate-based-statement-managed-keys 命令检索区域应用程序中使用的基于速率的规则当前阻止的 IP 地址。

```
aws wafv2 get-rate-based-statement-managed-keys \
    --scope REGIONAL \
    --web-acl-name testwebacl2 \
    --web-acl-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --rule-name ratebasedtest
```

输出：

```
{
    "ManagedKeysIPV4":{
```

```
        "IPAddressVersion":"IPV4",
        "Addresses":[
            "198.51.100.0/32"
        ]
    },
    "ManagedKeysIPV6":{
        "IPAddressVersion":"IPV6",
        "Addresses":[

        ]
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的基于速率的规则声明。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRateBasedStatementManagedKeys。

get-regex-pattern-set

以下代码示例演示了如何使用 get-regex-pattern-set。

Amazon CLI

检索特定正则表达式模式集

以下 get-regex-pattern-set 命令检索具有指定名称、范围、区域和 ID 的正则表达式模式集。可以从命令 create-regex-pattern-set 和 list-regex-pattern-sets 中获取正则表达式模式集的 ID。

```
aws wafv2 get-regex-pattern-set \
    --name regexPatterSet01 \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --region us-west-2
```

输出：

```
{
    "RegexPatternSet":{
        "Description":"Test web-acl",
```

```
        "RegularExpressionList":[
            {
                "RegexString":"/[0-9]*/"
            },
            {
                "RegexString":"/[a-z]*/"
            }
        ],
        "Name":"regexPatterSet01",
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "LockToken":"c8abf33f-b6fc-46ae-846e-42f994d57b29"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRegexPatternSet。

get-rule-group

以下代码示例演示了如何使用 get-rule-group。

Amazon CLI

检索特定的自定义规则组

以下 get-rule-group 命令检索具有指定名称、范围和 ID 的自定义规则组。可以从命令 create-rule-group 和 list-rule-groups 中获取规则组的 ID。

```
aws wafv2 get-rule-group \
    --name ff \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "RuleGroup":{
        "Capacity":1,
        "Description":"",
```

```
        "Rules":[
            {
                "Priority":0,
                "Action":{
                    "Block":{

                    }
                },
                "VisibilityConfig":{
                    "SampledRequestsEnabled":true,
                    "CloudWatchMetricsEnabled":true,
                    "MetricName":"jj"
                },
                "Name":"jj",
                "Statement":{
                    "SizeConstraintStatement":{
                        "ComparisonOperator":"LE",
                        "TextTransformations":[
                            {
                                "Priority":0,
                                "Type":"NONE"
                            }
                        ],
                        "FieldToMatch":{
                            "UriPath":{

                            }
                        },
                        "Size":7
                    }
                }
            }
        ],
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
            "CloudWatchMetricsEnabled":true,
            "MetricName":"ff"
        },
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/ff/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Name":"ff"
    },
    "LockToken":"485458c9-1830-4234-af31-ec4d52ced1b3"
```

```
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理您自己的规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRuleGroup。

get-sampled-requests

以下代码示例演示了如何使用 get-sampled-requests。

Amazon CLI

检索 Web ACL 的 Web 请求样本

以下 get-sampled-requests 命令检索指定 Web ACL、规则指标和时间范围的 Web 请求样本。

```
aws wafv2 get-sampled-requests \
    --web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-cli/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --rule-metric-name AWS-AWSManagedRulesSQLiRuleSet \
    --scope=REGIONAL \
    --time-window StartTime=2020-02-12T20:00Z,EndTime=2020-02-12T21:10Z \
    --max-items 100
```

输出：

```
{
    "TimeWindow": {
    "EndTime": 1581541800.0,
    "StartTime": 1581537600.0
    },
    "SampledRequests": [
        {
            "Action": "BLOCK",
            "Timestamp": 1581541799.564,
            "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
            "Request": {
                "Country": "US",
                "URI": "/",
                "Headers": [
                    {
```

```
                        "Name": "Host",
                        "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
                    },
                    {
                        "Name": "Content-Length",
                        "Value": "7456"
                    },
                    {
                        "Name": "User-Agent",
                        "Value": "curl/7.53.1"
                    },
                    {
                        "Name": "Accept",
                        "Value": "/"
                    },
                    {
                        "Name": "Content-Type",
                        "Value": "application/x-www-form-urlencoded"
                    }
                ],
                "ClientIP": "198.51.100.08",
                "Method": "POST",
                "HTTPVersion": "HTTP/1.1"
            },
            "Weight": 1
        },
        {
            "Action": "BLOCK",
            "Timestamp": 1581541799.988,
            "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
            "Request": {
                "Country": "US",
                "URI": "/",
                "Headers": [
                    {
                        "Name": "Host",
                        "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
                    },
                    {
                        "Name": "Content-Length",
                        "Value": "7456"
                    },
                    {
                        "Name": "User-Agent",
```

```
                    "Value": "curl/7.53.1"
                },
                {
                    "Name": "Accept",
                    "Value": "/"
                },
                {
                    "Name": "Content-Type",
                    "Value": "application/x-www-form-urlencoded"
                }
            ],
            "ClientIP": "198.51.100.08",
            "Method": "POST",
            "HTTPVersion": "HTTP/1.1"
        },
        "Weight": 3
    },
    {
        "Action": "BLOCK",
        "Timestamp": 1581541799.846,
        "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
        "Request": {
            "Country": "US",
            "URI": "/",
            "Headers": [
                {
                    "Name": "Host",
                    "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
                },
                {
                    "Name": "Content-Length",
                    "Value": "7456"
                },
                {
                    "Name": "User-Agent",
                    "Value": "curl/7.53.1"
                },
                {
                    "Name": "Accept",
                    "Value": "/"
                },
                {
                    "Name": "Content-Type",
                    "Value": "application/x-www-form-urlencoded"
```

```
                    }
                ],
                "ClientIP": "198.51.100.08",
                "Method": "POST",
                "HTTPVersion": "HTTP/1.1"
            },
            "Weight": 1
        },
        {
            "Action": "BLOCK",
            "Timestamp": 1581541799.4,
            "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
            "Request": {
                "Country": "US",
                "URI": "/",
                "Headers": [
                    {
                        "Name": "Host",
                        "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
                    },
                    {
                        "Name": "Content-Length",
                        "Value": "7456"
                    },
                    {
                        "Name": "User-Agent",
                        "Value": "curl/7.53.1"
                    },
                    {
                        "Name": "Accept",
                        "Value": "/"
                    },
                    {
                        "Name": "Content-Type",
                        "Value": "application/x-www-form-urlencoded"
                    }
                ],
                "ClientIP": "198.51.100.08",
                "Method": "POST",
                "HTTPVersion": "HTTP/1.1"
            },
            "Weight": 1
        }
    ],
```

```
    "PopulationSize": 4
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的查看 Web 请求样本。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSampledRequests。

get-web-acl-for-resource

以下代码示例演示了如何使用 get-web-acl-for-resource。

Amazon CLI

检索与 Amazon 资源关联的 Web ACL

以下 get-web-acl-for-resource 命令检索与指定资源关联的 Web ACL 的 JSON。

```
aws wafv2 get-web-acl-for-resource \
    --resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/waf-cli-alb/1ea17125f8b25a2a
```

输出：

```
{
    "WebACL":{
        "Capacity":3,
        "Description":"",
        "Rules":[
            {
                "Priority":1,
                "Action":{
                    "Block":{

                    }
                },
                "VisibilityConfig":{
                    "SampledRequestsEnabled":true,
                    "CloudWatchMetricsEnabled":true,
                    "MetricName":"testrule01"
                },
                "Name":"testrule01",
                "Statement":{
```

```
                "AndStatement":{
                    "Statements":[
                        {
                            "ByteMatchStatement":{
                                "PositionalConstraint":"EXACTLY",
                                "TextTransformations":[
                                    {
                                        "Priority":0,
                                        "Type":"NONE"
                                    }
                                ],
                                "SearchString":"dGVzdHN0cmluZw==",
                                "FieldToMatch":{
                                    "UriPath":{

                                    }
                                }
                            }
                        },
                        {
                            "SizeConstraintStatement":{
                                "ComparisonOperator":"EQ",
                                "TextTransformations":[
                                    {
                                        "Priority":0,
                                        "Type":"NONE"
                                    }
                                ],
                                "FieldToMatch":{
                                    "QueryString":{

                                   }
                                },
                                "Size":0
                            }
                        }
                    ]
                }
            }
        }
    ],
    "VisibilityConfig":{
        "SampledRequestsEnabled":true,
        "CloudWatchMetricsEnabled":true,
```

```
            "MetricName":"test01"
        },
        "DefaultAction":{
            "Allow":{

            }
        },
        "Id":"9a1b2c3d4-5678-90ab-cdef-EXAMPLE11111  ",
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test01/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111  ",
        "Name":"test01"
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的将 Web ACL 与 Amazon 资源关联或取消关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWebAclForResource。

get-web-acl

以下代码示例演示了如何使用 get-web-acl。

Amazon CLI

检索 Web ACL

以下 get-web-acl 命令检索具有指定名称、范围和 ID 的 Web ACL。可以从命令 create-web-acl 和 list-web-acls 中获取 Web ACL 的 ID。

```
aws wafv2 get-web-acl \
    --name test01 \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "WebACL":{
        "Capacity":3,
        "Description":"",
        "Rules":[
            {
```

```
                "Priority":1,
                "Action":{
                    "Block":{

                    }
                },
                "VisibilityConfig":{
                   "SampledRequestsEnabled":true,
                    "CloudWatchMetricsEnabled":true,
                    "MetricName":"testrule01"
                },
                "Name":"testrule01",
                "Statement":{
                    "AndStatement":{
                        "Statements":[
                            {
                                "ByteMatchStatement":{
                                    "PositionalConstraint":"EXACTLY",
                                    "TextTransformations":[
                                        {
                                            "Priority":0,
                                            "Type":"NONE"
                                        }
                                    ],
                                    "SearchString":"dGVzdHN0cmluZw==",
                                    "FieldToMatch":{
                                        "UriPath":{

                                        }
                                    }
                                }
                            },
                            {
                                "SizeConstraintStatement":{
                                    "ComparisonOperator":"EQ",
                                    "TextTransformations":[
                                        {
                                            "Priority":0,
                                            "Type":"NONE"
                                        }
                                    ],
                                    "FieldToMatch":{
                                        "QueryString":{
```

```
                                    }
                                },
                                "Size":0
                            }
                        }
                    ]
                }
            }
        ],
        "VisibilityConfig":{
            "SampledRequestsEnabled":true,
            "CloudWatchMetricsEnabled":true,
            "MetricName":"test01"
        },
        "DefaultAction":{
            "Allow":{

            }
        },
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test01/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Name":"test01"
    },
    "LockToken":"e3db7e2c-d58b-4ee6-8346-6aec5511c6fb"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理和使用 Web 访问控制列表（Web ACL）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetWebAcl。

`list-available-managed-rule-groups`

以下代码示例演示了如何使用 `list-available-managed-rule-groups`。

Amazon CLI

检索托管规则组

以下 `list-available-managed-rule-groups` 命令返回您的 Web ACL 中当前可供使用的所有托管规则组列表。

```
aws wafv2 list-available-managed-rule-groups \
    --scope REGIONAL
```

输出：

```
{
   "ManagedRuleGroups": [
       {
           "VendorName": "AWS",
           "Name": "AWSManagedRulesCommonRuleSet",
           "Description": "Contains rules that are generally applicable to web
applications. This provides protection against exploitation of a wide range of
vulnerabilities, including those described in OWASP publications and common Common
Vulnerabilities and Exposures (CVE)."
       },
       {
           "VendorName": "AWS",
           "Name": "AWSManagedRulesAdminProtectionRuleSet",
           "Description": "Contains rules that allow you to block external access
to exposed admin pages. This may be useful if you are running third-party software
or would like to reduce the risk of a malicious actor gaining administrative access
to your application."
       },
       {
           "VendorName": "AWS",
           "Name": "AWSManagedRulesKnownBadInputsRuleSet",
           "Description": "Contains rules that allow you to block request patterns
that are known to be invalid and are associated with exploitation or discovery of
vulnerabilities. This can help reduce the risk of a malicious actor discovering a
vulnerable application."
       },
       {
           "VendorName": "AWS",
           "Name": "AWSManagedRulesSQLiRuleSet",
           "Description": "Contains rules that allow you to block request patterns
associated with exploitation of SQL databases, like SQL injection attacks. This can
help prevent remote injection of unauthorized queries."
       },
       {
           "VendorName": "AWS",
           "Name": "AWSManagedRulesLinuxRuleSet",
           "Description": "Contains rules that block request patterns associated
with exploitation of vulnerabilities specific to Linux, including LFI attacks. This
```

```
can help prevent attacks that expose file contents or execute code for which the
attacker should not have had access."
        },
        {
            "VendorName": "AWS",
            "Name": "AWSManagedRulesUnixRuleSet",
            "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to POSIX/POSIX-like OS, including LFI
attacks. This can help prevent attacks that expose file contents or execute code
for which access should not been allowed."
        },
        {
            "VendorName": "AWS",
            "Name": "AWSManagedRulesWindowsRuleSet",
            "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to Windows, (e.g., PowerShell commands).
This can help prevent exploits that allow attacker to run unauthorized commands or
execute malicious code."
        },
        {
            "VendorName": "AWS",
            "Name": "AWSManagedRulesPHPRuleSet",
            "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to the use of the PHP, including injection
of unsafe PHP functions. This can help prevent exploits that allow an attacker to
remotely execute code or commands."
        },
        {
            "VendorName": "AWS",
            "Name": "AWSManagedRulesWordPressRuleSet",
            "Description": "The WordPress Applications group contains rules that
block request patterns associated with the exploitation of vulnerabilities specific
to WordPress sites."
        },
        {
            "VendorName": "AWS",
            "Name": "AWSManagedRulesAmazonIpReputationList",
            "Description": "This group contains rules that are based on Amazon
threat intelligence. This is useful if you would like to block sources associated
with bots or other threats."
        }
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的托管规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAvailableManagedRuleGroups。

`list-ip-sets`

以下代码示例演示了如何使用 `list-ip-sets`。

Amazon CLI

检索 IP 集列表

以下 `list-ip-sets` 命令检索账户中具有区域范围的所有 IP 集。

```
aws wafv2 list-ip-sets \
    --scope REGIONAL
```

输出：

```
{
    "IPSets":[
        {
            "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description":"",
            "Name":"testip",
            "LockToken":"0674c84b-0304-47fe-8728-c6bff46af8fc",
            "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111  "
        }
    ],
    "NextMarker":"testip"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListIpSets。

`list-logging-configurations`

以下代码示例演示了如何使用 `list-logging-configurations`。

Amazon CLI

检索某个区域的所有日志记录配置列表

以下 `list-logging-configurations` 命令检索 Web ACL 的所有日志记录配置，这些配置仅供 `us-west-2` 区域的用户使用。

```
aws wafv2 list-logging-configurations \
    --scope REGIONAL \
    --region us-west-2
```

输出：

```
{
    "LoggingConfigurations":[
        {
            "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
test-2/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "RedactedFields":[
                {
                    "QueryString":{

                    }
                }
            ],
            "LogDestinationConfigs":[
                "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-
logs-test"
            ]
        },
        {
            "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
test/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "RedactedFields":[
                {
                    "Method":{

                    }
                }
            ],
            "LogDestinationConfigs":[
                "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-
logs-custom-transformation"
```

```
            ]
        }
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的记录 Web ACL 流量日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListLoggingConfigurations。

`list-regex-pattern-sets`

以下代码示例演示了如何使用 `list-regex-pattern-sets`。

Amazon CLI

检索正则表达式模式集列表

以下 `list-regex-pattern-sets` 命令检索在区域 `us-west-2` 中定义的账户的所有正则表达式模式集。

```
aws wafv2 list-regex-pattern-sets \
--scope REGIONAL \
--region us-west-2
```

输出：

```
{
    "NextMarker":"regexPatterSet01",
    "RegexPatternSets":[
        {
            "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description":"Test web-acl",
            "Name":"regexPatterSet01",
            "LockToken":"f17743f7-0000-0000-0000-19a8b93bfb01",
            "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRegexPatternSets。

`list-resources-for-web-acl`

以下代码示例演示了如何使用 `list-resources-for-web-acl`。

Amazon CLI

检索与 Web ACL 关联的资源

以下 `list-resources-for-web-acl` 命令检索当前与区域 `us-west-2` 中指定 Web ACL 关联的 API Gateway REST API 资源。

```
aws wafv2 list-resources-for-web-acl \
    --web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/TestWebAcl/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --resource-type API_GATEWAY \
    --region us-west-2
```

输出：

```
{
    "ResourceArns": [
        "arn:aws:apigateway:us-west-2::/restapis/EXAMPLE111/stages/testing"
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的将 Web ACL 与 Amazon 资源关联或取消关联。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourcesForWebAcl。

`list-rule-groups`

以下代码示例演示了如何使用 `list-rule-groups`。

Amazon CLI

检索自定义规则组列表

以下 `list-rule-groups` 命令检索为指定范围和区域位置的账户定义的所有自定义规则组。

```
aws wafv2 list-rule-groups \
    --scope REGIONAL \
    --region us-west-2
```

输出：

```
{
    "RuleGroups":[
        {
            "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/
TestRuleGroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description":"",
            "Name":"TestRuleGroup",
            "LockToken":"1eb5ec48-0000-0000-0000-ee9b906c541e",
            "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        },
        {
            "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "Description":"",
            "Name":"test",
            "LockToken":"b0f4583e-998b-4880-9069-3fbe45738b43",
            "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    ],
    "NextMarker":"test"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理您自己的规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListRuleGroups。

`list-tags-for-resource`

以下代码示例演示了如何使用 `list-tags-for-resource`。

Amazon CLI

检索 Amazon WAF 资源的所有标签

以下 `list-tags-for-resource` 命令检索指定 Web ACL 的所有标签键和值对的列表。

```
aws wafv2 list-tags-for-resource \
    --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/testwebacl2/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

输出：

```
{
    "NextMarker":"",
    "TagInfoForResource":{
        "ResourceARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/testwebacl2/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "TagList":[

        ]
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon WAF 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

`list-web-acls`

以下代码示例演示了如何使用 `list-web-acls`。

Amazon CLI

检索指定范围内的 Web ACL

以下 `list-web-acls` 命令检索为指定范围内的账户定义的所有 Web ACL。

```
aws wafv2 list-web-acls \
    --scope REGIONAL
```

输出：

```
{
    "NextMarker":"Testt",
    "WebACLs":[
```

```
        {
            "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/Testt/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description":"sssss",
            "Name":"Testt",
            "LockToken":"7f36cb30-74ef-4cff-8cd4-a77e1aba1746",
            "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理和使用 Web 访问控制列表（Web ACL）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListWebAcls。

put-logging-configuration

以下代码示例演示了如何使用 put-logging-configuration。

Amazon CLI

向 Web ACL 中添加日志记录配置

以下 put-logging-configuration 命令将 Amazon Kinesis Data Firehose 日志记录配置 aws-waf-logs-custom-transformation 添加到指定的 Web ACL 中，但不从日志中加密任何字段。

```
aws wafv2 put-logging-configuration \
    --logging-configuration ResourceArn=arn:aws:wafv2:us-
west-2:123456789012:regional/webacl/test-cli/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111,LogDestinationConfigs=arn:aws:firehose:us-
west-2:123456789012:deliverystream/aws-waf-logs-custom-transformation \
            --region us-west-2
```

输出：

```
{
    "LoggingConfiguration":{
        "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-
cli/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "LogDestinationConfigs":[
```

```
            "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-logs-
custom-transformation"
        ]
    }
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的记录 Web ACL 流量日志。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutLoggingConfiguration。

`tag-resource`

以下代码示例演示了如何使用 `tag-resource`。

Amazon CLI

向 Amazon WAF 资源中添加标签

以下 `tag-resource` 示例向指定的 Web ACL 中添加具有 `Name` 键且值设为 `AWSWAF` 的标签。

```
aws wafv2 tag-resource \
    --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
apiGatewayWebAcl/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --tags Key=Name,Value=AWSWAF
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon WAF 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

`untag-resource`

以下代码示例演示了如何使用 `untag-resource`。

Amazon CLI

从 Amazon WAF 资源中删除标签

以下 `untag-resource` 示例从指定 Web ACL 中删除具有 `KeyName` 键的标签。

```
aws wafv2 untag-resource \
    --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
apiGatewayWebAcl/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --tag-keys "KeyName"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 Amazon WAF 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-ip-set

以下代码示例演示了如何使用 update-ip-set。

Amazon CLI

修改现有 IP 集的设置

以下 update-ip-set 命令更新指定 IP 集的设置。此调用需要一个 ID（可以从调用 list-ip-sets 中获取）和一个锁定令牌（可以从调用 list-ip-sets 和 get-ip-set 中获取）。此调用还返回一个可用于后续更新的锁定令牌。

```
aws wafv2 update-ip-set \
    --name testip \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --addresses 198.51.100.0/16 \
    --lock-token 447e55ac-2396-4c6d-b9f9-86b67c17f8b5
```

输出：

```
{
    "NextLockToken": "0674c84b-0304-47fe-8728-c6bff46af8fc"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateIpSet。

update-regex-pattern-set

以下代码示例演示了如何使用 update-regex-pattern-set。

Amazon CLI

修改现有正则表达式模式集的设置

以下 update-regex-pattern-set 命令更新指定正则表达式模式集的设置。此调用需要一个 ID（可以从调用 list-regex-pattern-sets 中获取）和一个锁定令牌（可以从调用 list-regex-pattern-sets 和 get-regex-pattern-set 中获取）。此调用还返回一个可用于后续更新的锁定令牌。

```
aws wafv2 update-regex-pattern-set \
    --name ExampleRegex \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --regular-expression-list RegexString="^.+$"  \
    --lock-token ed207e9c-82e9-4a77-aadd-81e6173ab7eb
```

输出：

```
{
    "NextLockToken": "12ebc73e-fa68-417d-a9b8-2bdd761a4fa5"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的 IP 集和正则表达式模式集。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRegexPatternSet。

update-rule-group

以下代码示例演示了如何使用 update-rule-group。

Amazon CLI

更新自定义规则组

以下 update-rule-group 命令更改现有自定义规则组的可见性配置。此调用需要一个 ID（可以从调用 list-rule-groups 中获取）和一个锁定令牌（可以从调用 list-rule-groups 和 get-rule-group 中获取）。此调用还返回一个可用于后续更新的锁定令牌。

```
aws wafv2 update-rule-group \
    --name TestRuleGroup \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token 7b3bcec2-0000-0000-0000-563bf47249f0 \
    --visibility-
config SampledRequestsEnabled=false,CloudWatchMetricsEnabled=false,MetricName=TestMetricsFor
 \
    --region us-west-2
```

输出：

```
{
    "NextLockToken": "1eb5ec48-0000-0000-0000-ee9b906c541e"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理您自己的规则组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateRuleGroup。

update-web-acl

以下代码示例演示了如何使用 update-web-acl。

Amazon CLI

更新 Web ACL

以下 update-web-acl 命令更改现有 Web ACL 的设置。此调用需要一个 ID（可以从调用 list-web-acls 中获取），以及一个锁定令牌和其他设置（可以从调用 get-web-acl 中获取）。此调用还返回一个可用于后续更新的锁定令牌。

```
aws wafv2 update-web-acl \
    --name TestWebAcl \
    --scope REGIONAL \
    --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
    --lock-token 2294b3a1-0000-0000-0000-a3ae04329de9 \
    --default-action Block={} \
    --visibility-
config SampledRequestsEnabled=false,CloudWatchMetricsEnabled=false,MetricName=NewMetricTestW
 \
```

```
    --rules file://waf-rule.json \
    --region us-west-2
```

输出：

```
{
    "NextLockToken": "714a0cfb-0000-0000-0000-2959c8b9a684"
}
```

有关更多信息，请参阅《Amazon WAF、Amazon Firewall Manager 和 Amazon Shield Advanced 开发人员指南》中的管理和使用 Web 访问控制列表（Web ACL）。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateWebAcl。

使用 Amazon CLI 的 WorkDocs 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 WorkDocs 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`abort-document-version-upload`

以下代码示例演示了如何使用 `abort-document-version-upload`。

Amazon CLI

停止上传文档版本

此示例停止上传先前启动的文档版本。

命令:

```
aws workdocs abort-document-version-upload --document-
id feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f012673132098 35f2 --version-
id 1536773972914-ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dcccd417da9313
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AbortDocumentVersionUpload。

activate-user

以下代码示例演示了如何使用 activate-user。

Amazon CLI

激活用户

此示例激活一个不活跃的用户。

命令:

```
aws workdocs activate-user --user-
id "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

输出：

```
{
  "User": {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Username": "exampleUser",
      "EmailAddress": "exampleUser@site.awsapps.com",
      "GivenName": "Example",
      "Surname": "User",
      "OrganizationId": "d-926726012c",
      "RootFolderId":
 "75f67c183aa1217409ac87576a45c03a5df5e6d8c51c35c01669970538e86cd0",
      "RecycleBinFolderId":
 "642b7dd3e60b14204534f3df7b1959e01b5d170f8c2707f410e40a8149120a57",
      "Status": "ACTIVE",
      "Type": "MINIMALUSER",
```

```
        "CreatedTimestamp": 1521226107.747,
        "ModifiedTimestamp": 1525297406.462,
        "Storage": {
            "StorageUtilizedInBytes": 0,
            "StorageRule": {
                "StorageAllocatedInBytes": 0,
                "StorageType": "QUOTA"
            }
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ActivateUser。

add-resource-permissions

以下代码示例演示了如何使用 add-resource-permissions。

Amazon CLI

为资源添加权限

此示例为指定主体添加资源访问权限。

命令:

```
aws workdocs add-resource-permissions --resource-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --
principals Id=anonymous,Type=ANONYMOUS,Role=VIEWER
```

输出：

```
{
    "ShareResults": [
        {
            "PrincipalId": "anonymous",
            "Role": "VIEWER",
            "Status": "SUCCESS",
            "ShareId":
 "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
            "StatusMessage": ""
        }
```

```
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AddResourcePermissions。

create-comment

以下代码示例演示了如何使用 create-comment。

Amazon CLI

添加新注释

此示例将新注释添加到指定的文档版本中。

命令:

```
aws workdocs create-comment --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-
id 1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
text "This is a comment."
```

输出：

```
{
  "Comment": {
      "CommentId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
      "ThreadId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
      "Text": "This is a comment.",
      "Contributor": {
          "Id": "arn:aws:iam::123456789123:user/exampleUser",
          "Username": "exampleUser",
          "GivenName": "Example",
          "Surname": "User",
          "Status": "ACTIVE"
      },
      "CreatedTimestamp": 1534799058.197,
      "Status": "PUBLISHED",
      "Visibility": "PUBLIC"
  }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateComment。

create-custom-metadata

以下代码示例演示了如何使用 create-custom-metadata。

Amazon CLI

创建自定义元数据

此示例为指定文档创建自定义元数据。

命令:

```
aws workdocs create-custom-metadata --resource-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --custom-
metadata KeyName1=example,KeyName2=example2
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateCustomMetadata。

create-folder

以下代码示例演示了如何使用 create-folder。

Amazon CLI

创建文件夹

此示例创建一个文件夹。

命令:

```
aws workdocs create-folder --name documents --parent-folder-
id 1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

输出：

```
{
  "Metadata": {
      "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
      "Name": "documents",
      "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "ParentFolderId":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
      "CreatedTimestamp": 1534450467.622,
      "ModifiedTimestamp": 1534450467.622,
      "ResourceState": "ACTIVE",
      "Signature": "",
      "Size": 0,
      "LatestVersionSize": 0
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateFolder。

create-labels

以下代码示例演示了如何使用 create-labels。

Amazon CLI

创建标签

此示例为文档创建一系列标签。

命令:

```
aws workdocs create-labels --resource-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --
labels "documents" "examples" "my_documents"
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateLabels。

create-notification-subscription

以下代码示例演示了如何使用 create-notification-subscription。

Amazon CLI

创建通知订阅

以下 create-notification-subscription 示例为指定的 Amazon WorkDocs 组织配置通知订阅。

```
aws workdocs create-notification-subscription \
    --organization-id d-123456789c \
    --protocol HTTPS \
    --subscription-type ALL \
    --notification-endpoint "https://example.com/example"
```

输出：

```
{
    "Subscription": {
        "SubscriptionId": "123ab4c5-678d-901e-f23g-45h6789j0123",
        "EndPoint": "https://example.com/example",
        "Protocol": "HTTPS"
    }
}
```

有关更多信息，请参阅《Amazon WorkDocs 开发人员指南》中的订阅通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateNotificationSubscription。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

创建新用户

此示例在 Simple AD 或 Microsoft AD 目录中创建一个新用户。

命令:

```
aws workdocs create-user --organization-id d-926726012c --username exampleUser2
 --email-address exampleUser2@site.awsapps.com --given-name example2Name --
surname example2Surname --password examplePa$$w0rd
```

输出：

```
{
  "User": {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Username": "exampleUser2",
      "EmailAddress": "exampleUser2@site.awsapps.com",
      "GivenName": "example2Name",
      "Surname": "example2Surname",
      "OrganizationId": "d-926726012c",
      "RootFolderId":
 "35b886cb17198cbd547655e58b025dff0cf34aaed638be52009567e23dc67390",
      "RecycleBinFolderId":
 "9858c3e9ed4c2460dde9aadb4c69fde998070dd46e5e985bd08ec6169ea249ff",
      "Status": "ACTIVE",
      "Type": "MINIMALUSER",
      "CreatedTimestamp": 1535478836.584,
      "ModifiedTimestamp": 1535478836.584,
      "Storage": {
          "StorageUtilizedInBytes": 0,
          "StorageRule": {
              "StorageAllocatedInBytes": 0,
              "StorageType": "QUOTA"
          }
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

deactivate-user

以下代码示例演示了如何使用 deactivate-user。

Amazon CLI

停用用户

此示例停用一个活跃用户。

命令:

```
aws workdocs deactivate-user --user-
id "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeactivateUser。

delete-comment

以下代码示例演示了如何使用 delete-comment。

Amazon CLI

从文档版本中删除指定注释

此示例从指定的文档版本中删除指定注释。

命令:

```
aws workdocs delete-comment --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-
id 1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
comment-id 1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteComment。

delete-custom-metadata

以下代码示例演示了如何使用 delete-custom-metadata。

Amazon CLI

从资源中删除自定义元数据

此示例从指定资源中删除所有自定义元数据。

命令:

```
aws workdocs delete-custom-metadata --resource-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --delete-all
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteCustomMetadata。

delete-document

以下代码示例演示了如何使用 delete-document。

Amazon CLI

删除文档

此示例删除指定文档。

命令:

```
aws workdocs delete-document --document-
id b83ed5e5b167b65ef69de9d597627ff1a0d4f07a45e67f1fab7d26b54427de0a
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDocument。

delete-folder-contents

以下代码示例演示了如何使用 delete-folder-contents。

Amazon CLI

删除文件夹的内容

此示例删除指定文件夹的内容。

命令:

```
aws workdocs delete-folder-contents --folder-
id 26fa8aa4ba2071447c194f7b150b07149dbdb9e1c8a301872dcd93a4735ce65d
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFolderContents。

delete-folder

以下代码示例演示了如何使用 delete-folder。

Amazon CLI

删除文件夹

此示例删除指定文件夹。

命令:

```
aws workdocs delete-folder --folder-
id 26fa8aa4ba2071447c194f7b150b07149dbdb9e1c8a301872dcd93a4735ce65d
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteFolder。

delete-labels

以下代码示例演示了如何使用 delete-labels。

Amazon CLI

删除标签

此示例删除文档中的指定标签。

命令:

```
aws workdocs delete-labels --resource-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --
labels "documents" "examples"
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteLabels。

delete-notification-subscription

以下代码示例演示了如何使用 delete-notification-subscription。

Amazon CLI

删除通知订阅

以下 delete-notification-subscription 示例删除指定通知订阅。

```
aws workdocs delete-notification-subscription \
    --subscription-id 123ab4c5-678d-901e-f23g-45h6789j0123 \
    --organization-id d-123456789c
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkDocs 开发人员指南》中的订阅通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteNotificationSubscription。

delete-user

以下代码示例演示了如何使用 delete-user。

Amazon CLI

删除用户

此示例删除一个用户。

命令:

```
aws workdocs delete-user --user-
id "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

describe-activities

以下代码示例演示了如何使用 describe-activities。

Amazon CLI

获取用户活动列表

此示例返回指定组织的最新用户活动列表，并对最新的两个活动设置了限制。

命令:

```
aws workdocs describe-activities --organization-id d-926726012c --limit 2
```

输出：

```
{
  "UserActivities": [
      {
          "Type": "DOCUMENT_VERSION_DOWNLOADED",
          "TimeStamp": 1534800122.17,
          "Initiator": {
              "Id": "arn:aws:iam::123456789123:user/exampleUser"
          },
          "ResourceMetadata": {
```

```
                "Type": "document",
                "Name": "updatedDoc",
                "Id":
 "15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3",
                "Owner": {
                    "Id":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
                    "GivenName": "exampleName",
                    "Surname": "exampleSurname"
                }
            }
        },
        {
            "Type": "DOCUMENT_VERSION_VIEWED",
            "TimeStamp": 1534799079.207,
            "Initiator": {
                "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
                "GivenName": "exampleName",
                "Surname": "exampleSurname"
            },
            "ResourceMetadata": {
                "Type": "document",
                "Name": "updatedDoc",
                "Id":
 "15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3",
                "Owner": {
                    "Id":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
                    "GivenName": "exampleName",
                    "Surname": "exampleSurname"
                }
            }
        }
    ],
    "Marker":
 "DnF1ZXJ5VGhlbkZldGNoAgAAAAAAAAS7FmlTaU1OdlFTU1h1UU00VVFIbDlRWHcAAAAAAAAJTRY3bWh5eUgzaVF1ZX
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeActivities。

describe-comments

以下代码示例演示了如何使用 describe-comments。

Amazon CLI

列出有关指定文档版本的所有注释

此示例列出指定文档版本的所有注释。

命令:

```
aws workdocs describe-comments --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-
id 1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920
```

输出：

```
{
  "Comments": [
      {
          "CommentId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
          "ThreadId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
          "Text": "This is a comment.",
          "Contributor": {
              "Username": "arn:aws:iam::123456789123:user/exampleUser",
              "Type": "USER"
          },
          "CreatedTimestamp": 1534799058.197,
          "Status": "PUBLISHED",
          "Visibility": "PUBLIC"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeComments。

describe-document-versions

以下代码示例演示了如何使用 describe-document-versions。

Amazon CLI

检索文档的版本

此示例检索指定文档的文档版本，包括初始化版本和源文档的 URL。

命令:

```
aws workdocs describe-document-versions --document-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --fields SOURCE
```

输出：

```
{
  "DocumentVersions": [
      {
          "Id":
 "1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
          "Name": "exampleDoc.docx",
          "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
          "Size": 13922,
          "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
          "Status": "ACTIVE",
          "CreatedTimestamp": 1534452029.587,
          "ModifiedTimestamp": 1534452029.849,
          "CreatorId":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
          "Source": {
              "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65/1534452029587-15e129dfc1875
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc29.docx&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180816T204149Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180816%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k2345678l901234mno56pqr78EXAMPLE"
          }
      },
      {
          "Id": "1529005196082-
bb75fa19abc287699cb07147f75816dce43a53a10f28dc001bf61ef2fab01c59",
          "Name": "exampleDoc.pdf",
          "ContentType": "application/pdf",
          "Size": 425916,
          "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
          "Status": "ACTIVE",
```

```
            "CreatedTimestamp": 1529005196.082,
            "ModifiedTimestamp": 1529005196.796,
            "CreatorId":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
            "Source": {
                "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65/1529005196082-
bb75fa19abc287699cb07147f75816dce43a53a10f28dc001bf61ef2fab01c59?
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc29.pdf&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180816T204149Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180816%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k234567l901234mno56pqr78EXAMPLE"
            }
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDocumentVersions。

describe-folder-contents

以下代码示例演示了如何使用 describe-folder-contents。

Amazon CLI

描述文件夹的内容

此示例描述指定文件夹的所有活动内容，包括文件夹中按日期升序排序的文档和子文件夹。

命令:

```
aws workdocs describe-folder-contents --folder-
id 1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678 --sort DATE --
order ASCENDING --type ALL
```

输出：

```
{
  "Folders": [
```

```
        {
            "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
            "Name": "testing",
            "CreatorId":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
            "ParentFolderId":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
            "CreatedTimestamp": 1534450467.622,
            "ModifiedTimestamp": 1534451113.504,
            "ResourceState": "ACTIVE",
            "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
            "Size": 23019,
            "LatestVersionSize": 11537
        }
    ],
    "Documents": [
        {
            "Id": "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
            "CreatorId":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
            "ParentFolderId":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
            "CreatedTimestamp": 1529005196.082,
            "ModifiedTimestamp": 1534452483.01,
            "LatestVersionMetadata": {
                "Id":
 "1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
                "Name": "exampleDoc.docx",
                "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
                "Size": 13922,
                "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
                "Status": "ACTIVE",
                "CreatedTimestamp": 1534452029.587,
                "ModifiedTimestamp": 1534452029.587,
                "CreatorId":
 "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
            },
            "ResourceState": "ACTIVE"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeFolderContents。

describe-groups

以下代码示例演示了如何使用 describe-groups。

Amazon CLI

检索组列表

以下 describe-groups 示例列出与指定 Amazon WorkDocs 组织关联的组。

```
aws workdocs describe-groups \
    --search-query "e" \
    --organization-id d-123456789c
```

输出：

```
{
    "Groups": [
        {
            "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444&d-123456789c",
            "Name": "Example Group 1"
        },
        {
            "Id": "S-1-1-11-1122222222-2222233333-3333334444-5555&d-123456789c",
            "Name": "Example Group 2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkDocs 管理指南》https://docs.amazonaws.cn/workdocs/latest/adminguide/getting_started.html中的 Amazon WorkDocs 入门。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGroups。

describe-notification-subscriptions

以下代码示例演示了如何使用 describe-notification-subscriptions。

Amazon CLI

检索通知订阅列表

以下 describe-notification-subscriptions 示例检索指定 Amazon WorkDocs 组织的通知订阅。

```
aws workdocs describe-notification-subscriptions \
    --organization-id d-123456789c
```

输出：

```
{
    "Subscriptions": [
        {
            "SubscriptionId": "123ab4c5-678d-901e-f23g-45h6789j0123",
            "EndPoint": "https://example.com/example",
            "Protocol": "HTTPS"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkDocs 开发人员指南》中的订阅通知。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeNotificationSubscriptions。

describe-resource-permissions

以下代码示例演示了如何使用 describe-resource-permissions。

Amazon CLI

获取资源权限列表

此示例返回指定资源（文档或文件夹）的权限列表。

命令:

```
aws workdocs describe-resource-permissions --resource-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3
```

输出：

```
{
```

```
  "Principals": [
      {
          "Id": "anonymous",
          "Type": "ANONYMOUS",
          "Roles": [
              {
                  "Role": "VIEWER",
                  "Type": "DIRECT"
              }
          ]
      },
      {
          "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
          "Type": "USER",
          "Roles": [
              {
                  "Role": "OWNER",
                  "Type": "DIRECT"
              }
          ]
      },
      {
          "Id": "d-926726012c",
          "Type": "ORGANIZATION",
          "Roles": [
              {
                  "Role": "VIEWER",
                  "Type": "INHERITED"
              }
          ]
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResourcePermissions。

describe-users

以下代码示例演示了如何使用 `describe-users`。

Amazon CLI

检索指定用户的详细信息

此示例检索指定组织中所有用户的详细信息。

命令:

```
aws workdocs describe-users --organization-id d-926726012c
```

输出：

```
{
  "Users": [
      {
          "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
          "Username": "example1User",
          "OrganizationId": "d-926726012c",
          "RootFolderId":
 "3c0e3f849dd20a9771d937b9bbcc97e18796150ae56c26d64a4fa0320a2dedc9",
          "RecycleBinFolderId":
 "c277f4c4d647be1f5147b3184ffa96e1e2bf708278b696cacba68ba13b91f4fe",
          "Status": "INACTIVE",
          "Type": "USER",
          "CreatedTimestamp": 1535478999.452,
          "ModifiedTimestamp": 1535478999.452
      },
      {
          "Id": "S-1-1-11-1111111111-2222222222-3333333333-4444&d-926726012c",
          "Username": "example2User",
          "EmailAddress": "example2User@site.awsapps.com",
          "GivenName": "example2Name",
          "Surname": "example2Surname",
          "OrganizationId": "d-926726012c",
          "RootFolderId":
 "35b886cb17198cbd547655e58b025dff0cf34aaed638be52009567e23dc67390",
          "RecycleBinFolderId":
 "9858c3e9ed4c2460dde9aadb4c69fde998070dd46e5e985bd08ec6169ea249ff",
          "Status": "ACTIVE",
          "Type": "MINIMALUSER",
          "CreatedTimestamp": 1535478836.584,
          "ModifiedTimestamp": 1535478836.584
      }
  ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUsers。

get-document-path

以下代码示例演示了如何使用 get-document-path。

Amazon CLI

检索文档的路径信息

此示例检索指定文档的路径信息（根文件夹中的层次结构），并包括父文件夹的名称。

命令:

```
aws workdocs get-document-path --document-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --fields NAME
```

输出：

```
{
  "Path": {
      "Components": [
          {
              "Id":
 "a43d29cbb8e7c4d25cfee8b803a504b0dc63e760b55ad0c611c6b87691eb6ff3",
              "Name": "/"
          },
          {
              "Id":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
              "Name": "Top Level Folder"
          },
          {
              "Id":
 "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
              "Name": "exampleDoc.docx"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocumentPath。

get-document-version

以下代码示例演示了如何使用 get-document-version。

Amazon CLI

检索指定文档的版本元数据

此示例检索指定文档的版本元数据，包括源 URL 和自定义元数据。

命令:

```
aws workdocs get-document-version --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-
id 1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
fields SOURCE --include-custom-metadata
```

输出：

```
{
  "Metadata": {
      "Id":
 "1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920",
      "Name": "exampleDoc",
      "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
      "Size": 11537,
      "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
      "Status": "ACTIVE",
      "CreatedTimestamp": 1521672507.741,
      "ModifiedTimestamp": 1534451113.504,
      "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Source": {
          "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3/152167
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180820T212202Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180820%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k2345678l901234mno56pqr78EXAMPLE"
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocumentVersion。

get-document

以下代码示例演示了如何使用 get-document。

Amazon CLI

检索文档详细信息

此示例检索指定文档的详细信息。

命令:

```
aws workdocs get-document --document-
id d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65
```

输出：

```
{
  "Metadata": {
      "Id": "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
      "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "ParentFolderId":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
      "CreatedTimestamp": 1529005196.082,
      "ModifiedTimestamp": 1534452483.01,
      "LatestVersionMetadata": {
          "Id":
 "1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
          "Name": "exampleDoc.docx",
          "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
          "Size": 13922,
          "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
          "Status": "ACTIVE",
          "CreatedTimestamp": 1534452029.587,
          "ModifiedTimestamp": 1534452029.587,
          "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
      },
      "ResourceState": "ACTIVE"
  }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDocument。

get-folder-path

以下代码示例演示了如何使用 get-folder-path。

Amazon CLI

检索文件夹的路径信息

此示例检索指定文件夹的路径信息（从根文件夹向下的层次结构），并包括父文件夹的名称。

命令:

```
aws workdocs get-folder-path --folder-
id 50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08 --fields NAME
```

输出：

```
{
  "Path": {
      "Components": [
          {
              "Id":
 "a43d29cbb8e7c4d25cfee8b803a504b0dc63e760b55ad0c611c6b87691eb6ff3",
              "Name": "/"
          },
          {
              "Id":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
              "Name": "Top Level Folder"
          },
          {
              "Id":
 "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
              "Name": "Sublevel Folder"
          }
      ]
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFolderPath。

get-folder

以下代码示例演示了如何使用 get-folder。

Amazon CLI

检索文件夹的元数据

此示例检索指定文件夹的元数据。

命令:

```
aws workdocs get-folder --folder-
id 50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08
```

输出：

```
{
  "Metadata": {
      "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
      "Name": "exampleFolder",
      "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "ParentFolderId":
 "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
      "CreatedTimestamp": 1534450467.622,
      "ModifiedTimestamp": 1534451113.504,
      "ResourceState": "ACTIVE",
      "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
      "Size": 23019,
      "LatestVersionSize": 11537
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetFolder。

get-resources

以下代码示例演示了如何使用 get-resources。

Amazon CLI

检索共享资源

以下 get-resources 示例检索与指定 Amazon WorkDocs 用户共享的资源。

```
aws workdocs get-resources \
    --user-id "S-1-1-11-1111111111-2222222222-3333333333-3333" \
    --collection-type SHARED_WITH_ME
```

输出：

```
{
"Folders": [],
"Documents": []
}
```

有关更多信息，请参阅《Amazon WorkDocs 用户指南》中的共享文件和文件夹。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetResources。

initiate-document-version-upload

以下代码示例演示了如何使用 initiate-document-version-upload。

Amazon CLI

启动文档版本上传

以下 initiate-document-upload 示例新建文档对象和版本对象。

```
aws workdocs initiate-document-version-upload \
    --name exampledocname \
    --parent-folder-
id eacd546d952531c633452ed67cac23161aa0d5df2e8061223a59e8f67e7b6189
```

输出：

```
{
    "Metadata": {
        "Id": "feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f01267313209835f2",
```

```
        "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
        "ParentFolderId":
 "eacd546d952531c633452ed67cac23161aa0d5df2e8061223a59e8f67e7b6189",
        "CreatedTimestamp": 1536773972.914,
        "ModifiedTimestamp": 1536773972.914,
        "LatestVersionMetadata": {
            "Id": "1536773972914-
ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dcccd417da9313",
            "Name": "exampledocname",
            "ContentType": "application/octet-stream",
            "Size": 0,
            "Status": "INITIALIZED",
            "CreatedTimestamp": 1536773972.914,
            "ModifiedTimestamp": 1536773972.914,
            "CreatorId": "arn:aws:iam::123456789123:user/EXAMPLE"
        },
        "ResourceState": "ACTIVE"
    },
    "UploadMetadata": {
        "UploadUrl": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f01267313209835f2/1536773972914-
ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dcccd417da9313?X-Amz-
Algorithm=AWS1-ABCD-EFG234&X-Amz-Date=20180912T173932Z&X-Amz-SignedHeaders=content-
type%3Bhost%3Bx-amz-server-side-encryption&X-Amz-Expires=899&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180912%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k2345678l901234mno56pqr78EXAMPLE",
        "SignedHeaders": {
            "Content-Type": "application/octet-stream",
            "x-amz-server-side-encryption": "ABC123"
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 InitiateDocumentVersionUpload。

`remove-all-resource-permissions`

以下代码示例演示了如何使用 `remove-all-resource-permissions`。

Amazon CLI

删除指定资源的所有权限

此示例删除指定资源的所有权限。

命令:

```
aws workdocs remove-all-resource-permissions --resource-
id 1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveAllResourcePermissions。

`remove-resource-permission`

以下代码示例演示了如何使用 `remove-resource-permission`。

Amazon CLI

从资源中删除权限

此示例从相关资源中删除对指定主体的权限。

命令:

```
aws workdocs remove-resource-permission --resource-
id 1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678 --principal-
id anonymous
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RemoveResourcePermission。

`update-document-version`

以下代码示例演示了如何使用 `update-document-version`。

Amazon CLI

将文档版本状态更改为 “活动”

此示例将文档版本的状态更改为“活动”。

命令:

```
aws workdocs update-document-version --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-
id 1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
version-status ACTIVE
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDocumentVersion。

update-document

以下代码示例演示了如何使用 update-document。

Amazon CLI

更新文档

此示例更新文档的名称和父文件夹。

命令:

```
aws workdocs update-document --document-
id 15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --
name updatedDoc --parent-folder-
id 50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateDocument。

update-folder

以下代码示例演示了如何使用 update-folder。

Amazon CLI

更新文件夹

此示例更新文件夹的名称和父文件夹。

命令:

```
aws workdocs update-folder --folder-
id 50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08 --
name exampleFolder1 --parent-folder-
id 1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

输出：

```
None
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateFolder。

update-user

以下代码示例演示了如何使用 update-user。

Amazon CLI

更新用户

此示例更新指定用户所在的时区。

命令:

```
aws workdocs update-user --user-
id "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c" --time-zone-
id "America/Los_Angeles"
```

输出：

```
{
  "User": {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Username": "exampleUser",
      "EmailAddress": "exampleUser@site.awsapps.com",
      "GivenName": "Example",
      "Surname": "User",
      "OrganizationId": "d-926726012c",
      "RootFolderId":
 "c5eceb5e1a2d1d460c9d1af8330ae117fc8d39bb1d3ed6acd0992d5ff192d986",
      "RecycleBinFolderId":
 "6ca20102926ad15f04b1d248d6d6e44f2449944eda5c758f9a1e9df6a6b7fa66",
      "Status": "ACTIVE",
      "Type": "USER",
      "TimeZoneId": "America/Los_Angeles",
      "Storage": {
          "StorageUtilizedInBytes": 0,
          "StorageRule": {
              "StorageAllocatedInBytes": 53687091200,
              "StorageType": "QUOTA"
          }
      }
  }
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateUser。

使用 Amazon CLI 的 Amazon WorkMail 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon WorkMail 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`associate-delegate-to-resource`

以下代码示例演示了如何使用 `associate-delegate-to-resource`。

Amazon CLI

添加资源委托

以下 `associate-delegate-to-resource` 命令为资源添加委托。

```
aws workmail associate-delegate-to-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-68bf2d3b1c0244aab7264c24b9217443 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateDelegateToResource。

`associate-member-to-group`

以下代码示例演示了如何使用 `associate-member-to-group`。

Amazon CLI

向组中添加成员

以下 `associate-member-to-group` 命令将指定成员添加到组中。

```
aws workmail associate-member-to-group \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --group-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --member-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateMemberToGroup。

`create-alias`

以下代码示例演示了如何使用 `create-alias`。

Amazon CLI

创建别名

以下 create-alias 命令为指定实体（用户或组）创建别名。

```
aws workmail create-alias \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --alias exampleAlias@site.awsapps.com
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAlias。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

创建新组

以下 create-group 命令为指定组织创建新组。

```
aws workmail create-group \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --name exampleGroup1
```

输出：

```
{
    "GroupId": "S-1-1-11-1122222222-2222233333-3333334444-4444"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-resource

以下代码示例演示了如何使用 create-resource。

Amazon CLI

创建新资源

以下 create-resource 命令为指定组织创建新资源（会议室）。

```
aws workmail create-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --name exampleRoom1 \
    --type ROOM
```

输出：

```
{
    "ResourceId": "r-7afe0efbade843a58cdc10251fce992c"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateResource。

create-user

以下代码示例演示了如何使用 create-user。

Amazon CLI

创建新用户

以下 create-user 命令创建一个新用户。

```
aws workmail create-user \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --name exampleName \
    --display-name exampleDisplayName \
    --password examplePa$$w0rd
```

输出：

```
{
    "UserId": "S-1-1-11-1111111111-2222222222-3333333333-3333"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

delete-access-control-rule

以下代码示例演示了如何使用 delete-access-control-rule。

Amazon CLI

删除访问控制规则

以下 delete-access-control-rule 示例从指定 Amazon WorkMail 组织中删除指定的访问控制规则。

```
aws workmail delete-access-control-rule \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza \
    --name "myRule"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用访问控制规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessControlRule。

delete-alias

以下代码示例演示了如何使用 delete-alias。

Amazon CLI

删除别名

以下 delete-alias 命令删除指定实体（用户或组）的别名。

```
aws workmail delete-alias \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --alias exampleAlias@site.awsapps.com
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAlias。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除现有组

以下 delete-group 命令从 Amazon WorkMail 中删除现有组。

```
aws workmail delete-group \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-mailbox-permissions

以下代码示例演示了如何使用 delete-mailbox-permissions。

Amazon CLI

删除邮箱权限

以下 delete-mailbox-permissions 命令删除先前授予用户或组的邮箱权限。实体代表拥有邮箱的用户，被授权者代表要删除其权限的用户或组。

```
aws workmail delete-mailbox-permissions \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --grantee-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteMailboxPermissions。

delete-resource

以下代码示例演示了如何使用 delete-resource。

Amazon CLI

删除现有资源

以下 `delete-resource` 命令从 Amazon WorkMail 中删除现有资源。

```
aws workmail delete-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-7afe0efbade843a58cdc10251fce992c
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteResource。

delete-user

以下代码示例演示了如何使用 `delete-user`。

Amazon CLI

删除用户

以下 `delete-user` 命令从 Amazon WorkMail 和所有后续系统中删除指定用户。

```
aws workmail delete-user \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

deregister-from-work-mail

以下代码示例演示了如何使用 `deregister-from-work-mail`。

Amazon CLI

禁用现有实体

以下 `deregister-from-work-mail` 命令禁止现有实体（用户、组或资源）使用 Amazon WorkMail。

```
aws workmail deregister-from-work-mail \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterFromWorkMail。

describe-group

以下代码示例演示了如何使用 describe-group。

Amazon CLI

检索组信息

以下 describe-group 命令检索有关指定组的信息。

```
aws workmail describe-group \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

输出：

```
{
    "GroupId": "S-1-1-11-1122222222-2222233333-3333334444-4444",
    "Name": "exampleGroup1",
    "State": "ENABLED"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeGroup。

describe-organization

以下代码示例演示了如何使用 describe-organization。

Amazon CLI

检索组织信息

以下 describe-organization 命令检索指定 Amazon WorkMail 组织的相关信息。

```
aws workmail describe-organization \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

输出：

```
{
    "OrganizationId": "m-d281d0a2fd824be5b6cd3d3ce909fd27",
    "Alias": "alias",
    "State": "Active",
    "DirectoryId": "d-926726012c",
    "DirectoryType": "VpcDirectory",
    "DefaultMailDomain": "site.awsapps.com",
    "CompletedDate": 1522693605.468,
    "ARN": "arn:aws:workmail:us-west-2:111122223333:organization/m-
n1pq2345678r901st2u3vx45x6789yza"
}
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用组织。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeOrganization。

describe-resource

以下代码示例演示了如何使用 describe-resource。

Amazon CLI

检索资源信息

以下 describe-resource 命令检索有关指定资源的信息。

```
aws workmail describe-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-7afe0efbade843a58cdc10251fce992c
```

输出：

```
{
    "ResourceId": "r-7afe0efbade843a58cdc10251fce992c",
    "Name": "exampleRoom1",
    "Type": "ROOM",
```

```
    "BookingOptions": {
        "AutoAcceptRequests": true,
        "AutoDeclineRecurringRequests": false,
        "AutoDeclineConflictingRequests": true
    },
    "State": "ENABLED"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeResource。

describe-user

以下代码示例演示了如何使用 describe-user。

Amazon CLI

检索用户信息

以下 describe-user 命令检索有关指定用户的信息。

```
aws workmail describe-user \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

输出：

```
{
    "UserId": "S-1-1-11-1111111111-2222222222-3333333333-3333",
    "Name": "exampleUser1",
    "Email": "exampleUser1@site.awsapps.com",
    "DisplayName": "",
    "State": "ENABLED",
    "UserRole": "USER",
    "EnabledDate": 1532459261.827
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeUser。

disassociate-delegate-from-resource

以下代码示例演示了如何使用 disassociate-delegate-from-resource。

Amazon CLI

从资源中删除标签

以下 disassociate-delegate-from-resource 命令从资源中移除指定成员。

```
ws workmail disassociate-delegate-from-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-68bf2d3b1c0244aab7264c24b9217443 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateDelegateFromResource。

disassociate-member-from-group

以下代码示例演示了如何使用 disassociate-member-from-group。

Amazon CLI

从组中删除成员

以下 disassociate-member-from-group 命令从组中删除指定成员。

```
aws workmail disassociate-member-from-group \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --group-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --member-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateMemberFromGroup。

get-access-control-effect

以下代码示例演示了如何使用 get-access-control-effect。

Amazon CLI

获得访问控制规则的效果

以下 get-access-control-effect 示例检索指定 Amazon WorkMail 组织的访问控制规则应用于指定 IP 地址、访问协议操作和用户 ID 时的效果。

```
aws workmail get-access-control-effect \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza \
    --ip-address "192.0.2.0" \
    --action "WindowsOutlook" \
    --user-id "S-1-1-11-1111111111-2222222222-3333333333-3333"
```

输出：

```
{
    "Effect": "DENY",
    "MatchedRules": [
        "myRule"
    ]
}
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用访问控制规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetAccessControlEffect。

get-mailbox-details

以下代码示例演示了如何使用 get-mailbox-details。

Amazon CLI

获取用户的邮箱详细信息

以下 get-mailbox-details 命令检索有关指定用户邮箱的详细信息。

```
aws workmail get-mailbox-details \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza \
    --user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

输出：

```
{
    "MailboxQuota": 51200,
    "MailboxSize": 0.03890800476074219
```

```
}
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的管理用户账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetMailboxDetails。

list-access-control-rules

以下代码示例演示了如何使用 list-access-control-rules。

Amazon CLI

列出访问控制规则

以下 list-access-control-rules 示例列出指定 Amazon WorkMail 组织的访问控制规则。

```
aws workmail list-access-control-rules \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza
```

输出：

```
{
    "Rules": [
        {
            "Name": "default",
            "Effect": "ALLOW",
            "Description": "Default WorkMail Rule",
            "DateCreated": 0.0,
            "DateModified": 0.0
        },
        {
            "Name": "myRule",
            "Effect": "DENY",
            "Description": "my rule",
            "UserIds": [
            "S-1-1-11-1111111111-2222222222-3333333333-3333"
            ],
            "DateCreated": 1581635628.0,
            "DateModified": 1581635628.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用访问控制规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessControlRules。

list-aliases

以下代码示例演示了如何使用 list-aliases。

Amazon CLI

列出成员的别名

以下 list-aliases 命令列出指定成员（用户或组）的别名。

```
aws workmail list-aliases \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

输出：

```
{
    "Aliases": [
        "exampleAlias@site.awsapps.com",
        "exampleAlias1@site.awsapps.com"
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAliases。

list-group-members

以下代码示例演示了如何使用 list-group-members。

Amazon CLI

列出组成员

以下 list-group-members 命令列出指定组的成员。

```
aws workmail list-group-members \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
```

```
    --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

输出：

```
{
    "Members": [
        {
            "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",
            "Name": "exampleUser1",
            "Type": "USER",
            "State": "ENABLED",
            "EnabledDate": 1532459261.827
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroupMembers。

list-groups

以下代码示例演示了如何使用 list-groups。

Amazon CLI

检索组列表

以下 list-groups 命令检索指定组织中组的摘要。

```
aws workmail list-groups \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

输出：

```
{
    "Groups": [
        {
            "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444",
            "Name": "exampleGroup1",
            "State": "DISABLED"
        },
        {
            "Id": "S-4-4-44-1122222222-2222233333-3333334444-4444",
```

```
            "Name": "exampleGroup2",
            "State": "ENABLED"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListGroups。

list-mailbox-permissions

以下代码示例演示了如何使用 list-mailbox-permissions。

Amazon CLI

删除邮箱权限

以下 list-mailbox-permissions 命令检索与指定实体的邮箱关联的邮箱权限。

```
aws workmail list-mailbox-permissions \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

输出：

```
{
    "Permissions": [
        {
            "GranteeId": "S-1-1-11-1122222222-2222233333-3333334444-4444",
            "GranteeType": "USER",
            "PermissionValues": [
                "FULL_ACCESS"
            ]
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListMailboxPermissions。

list-organizations

以下代码示例演示了如何使用 list-organizations。

Amazon CLI

检索组织列表

以下 `list-organizations` 命令检索客户组织的摘要。

```
aws workmail list-organizations
```

输出：

```
{
    "OrganizationSummaries": [
        {
            "OrganizationId": "m-d281d0a2fd824be5b6cd3d3ce909fd27",
            "Alias": "exampleAlias",
            "State": "Active"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListOrganizations。

`list-resource-delegates`

以下代码示例演示了如何使用 `list-resource-delegates`。

Amazon CLI

列出资源委托

以下 `list-resource-delegates` 命令检索与指定资源关联的委托。

```
aws workmail list-resource-delegates \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-68bf2d3b1c0244aab7264c24b9217443
```

输出：

```
{
    "Delegates": [
        {
            "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",
```

```
            "Type": "USER"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResourceDelegates。

list-resources

以下代码示例演示了如何使用 list-resources。

Amazon CLI

检索资源列表

以下 list-resources 命令检索指定组织的资源的摘要。

```
aws workmail list-resources \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

输出：

```
{
    "Resources": [
        {
            "Id": "r-7afe0efbade843a58cdc10251fce992c",
            "Name": "exampleRoom1",
            "Type": "ROOM",
            "State": "ENABLED"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListResources。

list-tags-for-resource

以下代码示例演示了如何使用 list-tags-for-resource。

Amazon CLI

列出资源标签

以下 `list-tags-for-resource` 示例列出指定 Amazon WorkMail 组织的标签。

```
aws workmail list-tags-for-resource \
    --resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-n1pq2345678r901st2u3vx45x6789yza
```

输出：

```
{
    "Tags": [
        {
            "Key": "priority",
            "Value": "1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的标记组织。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTagsForResource。

list-users

以下代码示例演示了如何使用 `list-users`。

Amazon CLI

如何检索用户列表

以下 `list-users` 命令检索指定组织中用户的摘要。

```
aws workmail list-users \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

输出：

```
{
    "Users": [
        {
            "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",
            "Email": "exampleUser1@site.awsapps.com",
```

```
            "Name": "exampleUser1",
            "State": "ENABLED",
            "UserRole": "USER",
            "EnabledDate": 1532459261.827
        },
        {
            "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444",
            "Name": "exampleGuestUser",
            "State": "DISABLED",
            "UserRole": "SYSTEM_USER"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

put-access-control-rule

以下代码示例演示了如何使用 put-access-control-rule。

Amazon CLI

创建新的访问控制规则

以下 put-access-control-rule 示例拒绝指定用户访问指定的 Amazon WorkMail 组织。

```
aws workmail put-access-control-rule \
    --name "myRule" \
    --effect "DENY" \
    --description "my rule" \
    --user-ids "S-1-1-11-1111111111-2222222222-3333333333-3333" \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用访问控制规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutAccessControlRule。

put-mailbox-permissions

以下代码示例演示了如何使用 put-mailbox-permissions。

Amazon CLI

设置邮箱权限

以下 put-mailbox-permissions 命令为指定的被授权者（用户或组）设置完全访问权限。实体代表邮箱的所有者。

```
aws workmail put-mailbox-permissions \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333 \
    --grantee-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --permission-values FULL_ACCESS
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutMailboxPermissions。

register-to-work-mail

以下代码示例演示了如何使用 register-to-work-mail。

Amazon CLI

注册现有或已禁用的实体

以下 register-to-work-mail 命令允许指定的现有实体（用户、组或资源）使用 Amazon WorkMail。

```
aws workmail register-to-work-mail \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \
    --email exampleGroup1@site.awsapps.com
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterToWorkMail。

reset-password

以下代码示例演示了如何使用 reset-password。

Amazon CLI

重置用户密码

以下 reset-password 命令重置指定用户的密码。

```
aws workmail reset-password \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --user-id S-1-1-11-1111111111-2222222222-3333333333-3333 \
    --password examplePa$$w0rd
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ResetPassword。

tag-resource

以下代码示例演示了如何使用 tag-resource。

Amazon CLI

将标签添加到资源中

以下 tag-resource 示例将键为 “优先级” 且值为 “1” 的标签应用于指定的 Amazon WorkMail 组织。

```
aws workmail tag-resource \
    --resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-
n1pq2345678r901st2u3vx45x6789yza \
    --tags "Key=priority,Value=1"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的标记组织。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TagResource。

untag-resource

以下代码示例演示了如何使用 untag-resource。

Amazon CLI

取消标记资源

以下 untag-resource 示例从指定 Amazon WorkMail 组织中移除指定的标签。

```
aws workmail untag-resource \
    --resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-n1pq2345678r901st2u3vx45x6789yza \
    --tag-keys "priority"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的标记组织。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UntagResource。

update-mailbox-quota

以下代码示例演示了如何使用 update-mailbox-quota。

Amazon CLI

更新用户的邮箱配额

以下 update-mailbox-quota 命令更改指定用户的邮箱配额。

```
aws workmail update-mailbox-quota \
    --organization-id m-n1pq2345678r901st2u3vx45x6789yza \
    --user-id S-1-1-11-1111111111-2222222222-3333333333-3333 \
    --mailbox-quota 40000
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的管理用户账户。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateMailboxQuota。

update-primary-email-address

以下代码示例演示了如何使用 update-primary-email-address。

Amazon CLI

更新主电子邮件地址

以下 update-primary-email-address 命令更新指定实体（用户、组或资源）的主电子邮件地址。

```
aws workmail update-primary-email-address \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333 \
    --email exampleUser2@site.awsapps.com
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdatePrimaryEmailAddress。

update-resource

以下代码示例演示了如何使用 update-resource。

Amazon CLI

更新资源

以下 update-resource 命令更新指定资源的名称。

```
aws workmail update-resource \
    --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
    --resource-id r-7afe0efbade843a58cdc10251fce992c \
    --name exampleRoom2
```

此命令不生成任何输出。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateResource。

使用 Amazon CLI 的 Amazon WorkMail Message Flow 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 Amazon WorkMail Message Flow 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`get-raw-message-content`

以下代码示例演示了如何使用 `get-raw-message-content`。

Amazon CLI

获取电子邮件的原始内容

以下 `get-raw-message-content` 示例获取传输中电子邮件的原始内容并将其发送到名 `test` 为的文本文件。

```
aws workmailmessageflow get-raw-message-content \
    --message-id a1b2cd34-ef5g-6h7j-kl8m-npq9012345rs \
    test
```

运行以下命令后 `test` 文件的内容：

```
Subject: Hello World
From: =?UTF-8?Q?marymajor_marymajor?= <marymajor@example.com>
To: =?UTF-8?Q?mateojackson=40example=2Enet?= <mateojackson@example.net>
Date: Thu, 7 Nov 2019 19:22:46 +0000
Mime-Version: 1.0
Content-Type: multipart/alternative;
 boundary="=_EXAMPLE+"
References: <mail.1ab23c45.5de6.7f890g123hj45678@storage.wm.amazon.com>
X-Priority: 3 (Normal)
X-Mailer: Amazon WorkMail
Thread-Index: EXAMPLE
Thread-Topic: Hello World
Message-Id: <mail.1ab23c45.5de6.7f890g123hj45678@storage.wm.amazon.com>

This is a multi-part message in MIME format. Your mail reader does not
understand MIME message format.
```

```
--=_EXAMPLE+
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

hello world


--=_EXAMPLE+
Content-Type: text/html; charset=utf-8
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML><html>
<head>
<meta name=3D"Generator" content=3D"Amazon WorkMail v3.0-4510">
<meta http-equiv=3D"Content-Type" content=3D"text/html; charset=3Dutf-8">=

<title>testing</title>
</head>
<body>
<p style=3D"margin: 0px; font-family: Arial, Tahoma, Helvetica, sans-seri=
f; font-size: small;">hello world</p>
</body>
</html>
--=_EXAMPLE+--
```

有关更多信息，请参阅《Amazon WorkMail 管理员指南》中的使用 Amazon Lambda 检索消息内容。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetRawMessageContent。

使用 Amazon CLI 的 WorkSpaces 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 WorkSpaces 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`create-tags`

以下代码示例演示了如何使用 `create-tags`。

Amazon CLI

向 WorkSpace 中添加标签

以下 `create-tags` 示例将指定标签添加到指定的 WorkSpace 中。

```
aws workspaces create-tags \
    --resource-id ws-dk1xzr417 \
    --tags Key=Department,Value=Finance
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的标记 WorkSpaces 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTags。

`create-workspaces`

以下代码示例演示了如何使用 `create-workspaces`。

Amazon CLI

示例 1：创建 AlwaysOn WorkSpace

以下 `create-workspaces` 示例使用指定目录和捆绑包为指定用户创建 AlwaysOn WorkSpace。

```
aws workspaces create-workspaces \
    --workspaces DirectoryId=d-926722edaf,UserName=Mateo,BundleId=wsb-0zsvgp8fc
```

输出：

```
{
    "FailedRequests": [],
    "PendingRequests": [
```

```
        {
            "WorkspaceId": "ws-kcqms853t",
            "DirectoryId": "d-926722edaf",
            "UserName": "Mateo",
            "State": "PENDING",
            "BundleId": "wsb-0zsvgp8fc"
        }
    ]
}
```

示例 2：创建 AutoStop WorkSpace

以下 `create-workspaces` 示例使用指定目录和捆绑包为指定用户创建 AutoStop WorkSpace。

```
aws workspaces create-workspaces \
    --
workspaces DirectoryId=d-926722edaf,UserName=Mary,BundleId=wsb-0zsvgp8fc,WorkspaceProperties
```

输出：

```
{
    "FailedRequests": [],
    "PendingRequests": [
        {
            "WorkspaceId": "ws-dk1xzr417",
            "DirectoryId": "d-926722edaf",
            "UserName": "Mary",
            "State": "PENDING",
            "BundleId": "wsb-0zsvgp8fc"
        }
    ]
}
```

示例 3：创建用户解耦的 WorkSpace

以下 `create-workspaces` 示例通过将用户名设置为 `[UNDEFINED]`，以及指定 WorkSpace 名称、目录 ID 和捆绑包 ID 来创建用户解耦的 WorkSpace。

```
aws workspaces create-workspaces \
    --workspaces
 DirectoryId=d-926722edaf,UserName='"[UNDEFINED]"',WorkspaceName=MaryWorkspace1,BundleId=wsb
```

输出：

```
{
    "FailedRequests": [],
    "PendingRequests": [
        {
            "WorkspaceId": "ws-abcd1234",
            "DirectoryId": "d-926722edaf",
            "UserName": "[UNDEFINED]",
            "State": "PENDING",
            "BundleId": "wsb-0zsvgp8fc",
            "WorkspaceName": "MaryWorkspace1"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的启动虚拟桌面。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateWorkspaces。

delete-tags

以下代码示例演示了如何使用 delete-tags。

Amazon CLI

从 WorkSpace 中删除标签

以下 delete-tags 示例从指定 WorkSpace 中删除指定标签。

```
aws workspaces delete-tags \
    --resource-id ws-dk1xzr417 \
    --tag-keys Department
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的标记 WorkSpaces 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTags。

deregister-workspace-directory

以下代码示例演示了如何使用 deregister-workspace-directory。

Amazon CLI

取消注册目录

以下 deregister-workspace-directory 示例取消注册指定目录。

```
aws workspaces deregister-workspace-directory \
    --directory-id d-926722edaf
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的向 WorkSpaces 注册目录。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeregisterWorkspaceDirectory。

describe-tags

以下代码示例演示了如何使用 describe-tags。

Amazon CLI

描述 WorkSpace 标签

以下 describe-tags 示例描述指定 WorkSpace 的标签。

```
aws workspaces describe-tags \
    --resource-id ws-dk1xzr417
```

输出：

```
{
    "TagList": [
        {
            "Key": "Department",
            "Value": "Finance"
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的标记 WorkSpaces 资源。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTags。

describe-workspace-bundles

以下代码示例演示了如何使用 describe-workspace-bundles。

Amazon CLI

列出 Amazon 提供的捆绑包

以下 describe-workspace-bundles 示例列出由 Amazon 提供的捆绑包的名称和 ID，采用表格格式并按名称排序。

```
aws workspaces describe-workspace-bundles \
    --owner AMAZON \
    --query "Bundles[*].[Name, BundleId]"
```

输出：

```
[
    [
        "Standard with Amazon Linux 2",
        "wsb-clj85qzj1"
    ],
    [
        "Performance with Windows 10 (Server 2016 based)",
        "wsb-gm4d5tx2v"
    ],
    [
        "PowerPro with Windows 7",
        "wsb-1pzkp0bx4"
    ],
    [
        "Power with Amazon Linux 2",
        "wsb-2bs6k5lgn"
    ],
    [
        "Graphics with Windows 10 (Server 2019 based)",
        "wsb-03gyjnfyy"
    ],
    ...
]
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的 WorkSpaces 捆绑包和映像。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeWorkspaceBundles。

describe-workspace-directories

以下代码示例演示了如何使用 describe-workspace-directories。

Amazon CLI

描述已注册的目录

以下 describe-workspace-directories 示例描述指定的注册目录。

```
aws workspaces describe-workspace-directories \
    --directory-ids d-926722edaf
```

输出：

```
{
    "Directories": [
        {
            "DirectoryId": "d-926722edaf",
            "Alias": "d-926722edaf",
            "DirectoryName": "example.com",
            "RegistrationCode": "WSpdx+9RJ8JT",
            "SubnetIds": [
                "subnet-9d19c4c6",
                "subnet-500d5819"
            ],
            "DnsIpAddresses": [
                "172.16.1.140",
                "172.16.0.30"
            ],
            "CustomerUserName": "Administrator",
            "IamRoleId": "arn:aws:iam::123456789012:role/workspaces_DefaultRole",
            "DirectoryType": "SIMPLE_AD",
            "WorkspaceSecurityGroupId": "sg-0d89e927e5645d7c5",
            "State": "REGISTERED",
            "WorkspaceCreationProperties": {
                "EnableInternetAccess": false,
                "UserEnabledAsLocalAdministrator": true,
                "EnableMaintenanceMode": true
            },
```

```
            "WorkspaceAccessProperties": {
                "DeviceTypeWindows": "ALLOW",
                "DeviceTypeOsx": "ALLOW",
                "DeviceTypeWeb": "DENY",
                "DeviceTypeIos": "ALLOW",
                "DeviceTypeAndroid": "ALLOW",
                "DeviceTypeChromeOs": "ALLOW",
                "DeviceTypeZeroClient": "ALLOW",
                "DeviceTypeLinux": "DENY"
            },
            "Tenancy": "SHARED",
            "SelfservicePermissions": {
                "RestartWorkspace": "ENABLED",
                "IncreaseVolumeSize": "DISABLED",
                "ChangeComputeType": "DISABLED",
                "SwitchRunningMode": "DISABLED",
                "RebuildWorkspace": "DISABLED"
            }
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkSpaces Administration Guide》中的 Manage directories for WorkSpaces Personal。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeWorkspaceDirectories。

`describe-workspaces-connection-status`

以下代码示例演示了如何使用 `describe-workspaces-connection-status`。

Amazon CLI

描述 WorkSpace 的连接状态

以下 `describe-workspaces-connection-status` 示例描述指定 WorkSpace 的连接状态。

```
aws workspaces describe-workspaces-connection-status \
    --workspace-ids ws-dk1xzr417
```

输出：

```
{
```

```
    "WorkspacesConnectionStatus": [
        {
            "WorkspaceId": "ws-dk1xzr417",
            "ConnectionState": "CONNECTED",
            "ConnectionStateCheckTimestamp": 1662526214.744
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的管理您的 WorkSpaces。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeWorkspacesConnectionStatus。

`describe-workspaces`

以下代码示例演示了如何使用 `describe-workspaces`。

Amazon CLI

描述 WorkSpace

以下 `describe-workspaces` 示例描述指定的 WorkSpace。

```
aws workspaces describe-workspaces \
    --workspace-ids ws-dk1xzr417
```

输出：

```
{
    "Workspaces": [
        {
            "WorkspaceId": "ws-dk1xzr417",
            "DirectoryId": "d-926722edaf",
            "UserName": "Mary",
            "IpAddress": "172.16.0.175",
            "State": "STOPPED",
            "BundleId": "wsb-0zsvgp8fc",
            "SubnetId": "subnet-500d5819",
            "ComputerName": "WSAMZN-RBSLTTD9",
            "WorkspaceProperties": {
                "RunningMode": "AUTO_STOP",
```

```
                "RunningModeAutoStopTimeoutInMinutes": 60,
                "RootVolumeSizeGib": 80,
                "UserVolumeSizeGib": 10,
                "ComputeTypeName": "VALUE"
            },
            "ModificationStates": []
        }
    ]
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的管理您的 WorkSpaces。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeWorkspaces。

`migrate-workspace`

以下代码示例演示了如何使用 `migrate-workspace`。

Amazon CLI

迁移 WorkSpace

以下 `migrate-workspace` 示例将指定 WorkSpace 迁移到指定的捆绑包。

```
aws workspaces migrate-workspace \
    --source-workspace-id ws-dk1xzr417 \
    --bundle-id wsb-j4dky1gs4
```

输出：

```
{
    "SourceWorkspaceId": "ws-dk1xzr417",
    "TargetWorkspaceId": "ws-x5h1lbkp5"
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的迁移 WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 MigrateWorkspace。

`modify-workspace-creation-properties`

以下代码示例演示了如何使用 `modify-workspace-creation-properties`。

Amazon CLI

修改目录 WorkSpace 创建属性

以下 modify-workspace-creation-properties 示例为指定目录启用 EnableInternetAccess 属性。这样可以针对为目录创建的 WorkSpaces 自动分配公共 IP 地址。

```
aws workspaces modify-workspace-creation-properties \
    --resource-id d-926722edaf \
    --workspace-creation-properties EnableInternetAccess=true
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的更新 WorkSpaces 的目录详细信息。

- 有关 API 详细信息，请参阅《 Amazon CLI 命令参考》中的 ModifyWorkspaceCreationProperties。

modify-workspace-properties

以下代码示例演示了如何使用 modify-workspace-properties。

Amazon CLI

修改 WorkSpace 的运行模式

以下 modify-workspace-properties 示例将指定 WorkSpace 的运行模式设置为 AUTO_STOP。

```
aws workspaces modify-workspace-properties \
    --workspace-id ws-dk1xzr417 \
    --workspace-properties RunningMode=AUTO_STOP
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的修改 WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyWorkspaceProperties。

`modify-workspace-state`

以下代码示例演示了如何使用 `modify-workspace-state`。

Amazon CLI

修改 WorkSpace 的状态

以下 `modify-workspace-state` 示例将指定 WorkSpace 的状态设置为 `ADMIN_MAINTENANCE`。

```
aws workspaces modify-workspace-state \
    --workspace-id ws-dk1xzr417 \
    --workspace-state ADMIN_MAINTENANCE
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的 WorkSpaces 维护。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyWorkspaceState。

`reboot-workspaces`

以下代码示例演示了如何使用 `reboot-workspaces`。

Amazon CLI

重启 WorkSpace

以下 `reboot-workspaces` 示例重启指定的 WorkSpace。

```
aws workspaces reboot-workspaces \
    --reboot-workspace-requests ws-dk1xzr417
```

输出：

```
{
    "FailedRequests": []
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的重启 WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootWorkspaces。

rebuild-workspaces

以下代码示例演示了如何使用 rebuild-workspaces。

Amazon CLI

重建 WorkSpace

以下 rebuild-workspaces 示例重建指定的 WorkSpace。

```
aws workspaces rebuild-workspaces \
    --rebuild-workspace-requests ws-dk1xzr417
```

输出：

```
{
    "FailedRequests": []
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的重建 WorkSpace。

- 有关 API 详细信息，请参阅《命令参考》中的 RebuildWorkspaces。Amazon CLI

register-workspace-directory

以下代码示例演示了如何使用 register-workspace-directory。

Amazon CLI

注册目录

以下 register-workspace-directory 示例注册指定目录，以将其与 Amazon WorkSpaces 一起使用。

```
aws workspaces register-workspace-directory \
    --directory-id d-926722edaf
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces Administration Guide》中的 Register an existing Amazon Directory Service directory with WorkSpaces Personal。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RegisterWorkspaceDirectory。

`restore-workspace`

以下代码示例演示了如何使用 `restore-workspace`。

Amazon CLI

还原 WorkSpace

以下 `restore-workspace` 示例还原指定的 WorkSpace。

```
aws workspaces restore-workspace \
    --workspace-id ws-dk1xzr417
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的还原 WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RestoreWorkspace。

`start-workspaces`

以下代码示例演示了如何使用 `start-workspaces`。

Amazon CLI

启动 AutoStop WorkSpace

以下 `start-workspaces` 示例启动指定的 WorkSpace。WorkSpace 的运行模式必须为 AutoStop。

```
aws workspaces start-workspaces \
    --start-workspace-requests WorkspaceId=ws-dk1xzr417
```

输出：

```
{
    "FailedRequests": []
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的停止和启动 AutoStop WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartWorkspaces。

stop-workspaces

以下代码示例演示了如何使用 stop-workspaces。

Amazon CLI

停止 AutoStop WorkSpace

以下 stop-workspaces 示例停止指定的 WorkSpace。WorkSpace 的运行模式必须为 AutoStop。

```
aws workspaces stop-workspaces \
    --stop-workspace-requests WorkspaceId=ws-dk1xzr417
```

输出：

```
{
    "FailedRequests": []
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的停止和启动 AutoStop WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopWorkspaces。

terminate-workspaces

以下代码示例演示了如何使用 terminate-workspaces。

Amazon CLI

终止 WorkSpace

以下 terminate-workspaces 示例终止指定的 Workspace。

```
aws workspaces terminate-workspaces \
    --terminate-workspace-requests ws-dk1xzr417
```

输出：

```
{
    "FailedRequests": []
```

```
}
```

有关更多信息，请参阅《Amazon WorkSpaces 管理指南》中的删除 WorkSpace。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateWorkspaces。

使用 Amazon CLI 的 X-Ray 示例

以下代码示例演示如何通过将 Amazon Command Line Interface与 X-Ray 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`batch-traces-get`

以下代码示例演示了如何使用 `batch-traces-get`。

Amazon CLI

获取跟踪列表

以下 `batch-get-traces` 示例检索按 ID 指定的跟踪列表。完整跟踪为每个分段包括一个文档，该文档根据收到的具有相同跟踪 ID 的所有分段文档编译。

```
aws xray batch-get-traces \
    --trace-ids 1-5d82881a-0a9126e92a73e971eed891b9
```

输出：

```
{
    "Traces": [
        {
```

```
            "Id": "1-5d82881a-0a9126e92a73e971eed891b9",
            "Duration": 0.232,
            "Segments": [
                {
                    "Id": "54aff5735b12dd28",
                    "Document": "{\"id\":\"54aff5735b12dd28\",\"name\":
\"Scorekeep\",\"start_time\":1.568835610432E9,\"end_time\":1.568835610664E9,
\"http\":{\"request\":{\"url\":\"http://scorekeep-env-1.m4fg2pfzpv.us-
east-2.elasticbeanstalk.com/api/user\",\"method\":\"POST\",\"user_agent\":
\"curl/7.59.0\",\"client_ip\":\"52.95.4.28\",\"x_forwarded_for\":true},
\"response\":{\"status\":200}},\"aws\":{\"elastic_beanstalk\":{\"version_label
\":\"Sample Application-1\",\"deployment_id\":3,\"environment_name\":\"Scorekeep-
env-1\"},\"ec2\":{\"availability_zone\":\"us-east-2b\",\"instance_id\":
\"i-0e3cf4d2de0f3f37a\"},\"xray\":{\"sdk_version\":\"1.1.0\",\"sdk\":\"X-Ray for
 Java\"}},\"service\":{\"runtime\":\"OpenJDK 64-Bit Server VM\",\"runtime_version
\":\"1.8.0_222\"},\"trace_id\":\"1-5d82881a-0a9126e92a73e971eed891b9\",
\"origin\":\"AWS::ElasticBeanstalk::Environment\",\"subsegments\":[{\"id\":
\"2d6900034ccfe558\",\"name\":\"DynamoDB\",\"start_time\":1.568835610658E9,
\"end_time\":1.568835610664E9,\"http\":{\"response\":{\"status\":200,
\"content_length\":61}},\"aws\":{\"table_name\":\"scorekeep-user\",\"operation\":
\"UpdateItem\",\"request_id\":\"TPEIDNDUROMLPOV17U4A79555NVV4KQNSO5AEMVJF66Q9ASUAAJG
\",\"resource_names\":[\"scorekeep-user\"]},\"namespace\":\"aws\"}]}"
                },
                {
                    "Id": "0f278b6334c34e6b",
                    "Document": "{\"id\":\"0f278b6334c34e6b\",\"name\":
\"DynamoDB\",\"start_time\":1.568835610658E9,\"end_time\":1.568835610664E9,
\"parent_id\":\"2d6900034ccfe558\",\"inferred\":true,\"http\":{\"response
\":{\"status\":200,\"content_length\":61}},\"aws\":{\"table_name
\":\"scorekeep-user\",\"operation\":\"UpdateItem\",\"request_id\":
\"TPEIDNDUROMLPOV17U4A79555NVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"resource_names\":
[\"scorekeep-user\"]},\"trace_id\":\"1-5d82881a-0a9126e92a73e971eed891b9\",\"origin
\":\"AWS::DynamoDB::Table\"}"
                }
            ]
        }
    ],
    "UnprocessedTraceIds": []
}
```


有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的结合使用 Amazon X-Ray API 和 Amazon CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchTracesGet。

create-group

以下代码示例演示了如何使用 create-group。

Amazon CLI

创建组

以下 create-group 示例创建一个名为 AdminGroup 的组资源。该组将获得一个筛选表达式，以便将组标准定义为与导致故障或错误的特定服务相关的分段。

```
aws xray create-group \
   --group-name "AdminGroup" \
   --filter-expression "service(\"mydomain.com\") {fault OR error}"
```

输出：

```
{
    "GroupName": "AdminGroup",
    "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/AdminGroup/123456789",
    "FilterExpression": "service(\"mydomain.com\") {fault OR error}"
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateGroup。

create-sampling-rule

以下代码示例演示了如何使用 create-sampling-rule。

Amazon CLI

创建采样规则

以下 create-sampling-rule 示例创建规则以控制仪器应用程序的采样行为。这些规则由 JSON 文件提供。大多数采样规则字段都是创建规则所必需的。

```
aws xray create-sampling-rule \
```

```
    --cli-input-json file://9000-base-scorekeep.json
```

的内容9000-base-scorekeep.json：

```
{
    "SamplingRule": {
        "RuleName": "base-scorekeep",
        "ResourceARN": "*",
        "Priority": 9000,
        "FixedRate": 0.1,
        "ReservoirSize": 5,
        "ServiceName": "Scorekeep",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1
    }
}
```

输出：

```
{
    "SamplingRuleRecord": {
        "SamplingRule": {
            "RuleName": "base-scorekeep",
            "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/base-
scorekeep",
            "ResourceARN": "*",
            "Priority": 9000,
            "FixedRate": 0.1,
            "ReservoirSize": 5,
            "ServiceName": "Scorekeep",
            "ServiceType": "*",
            "Host": "*",
            "HTTPMethod": "*",
            "URLPath": "*",
            "Version": 1,
            "Attributes": {}
        },
        "CreatedAt": 1530574410.0,
        "ModifiedAt": 1530574410.0
    }
```

```
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSamplingRule。

delete-group

以下代码示例演示了如何使用 delete-group。

Amazon CLI

删除组

以下 delete-group 示例删除指定的组资源。

```
aws xray delete-group \
    --group-name "AdminGroup" \
    --group-arn "arn:aws:xray:us-east-2:123456789012:group/AdminGroup/123456789"
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteGroup。

delete-sampling-rule

以下代码示例演示了如何使用 delete-sampling-rule。

Amazon CLI

删除采样规则

以下 delete-sampling-rule 示例删除指定的采用规则。可以使用组名称或组 ARN 来指定组。

```
aws xray delete-sampling-rule \
    --rule-name polling-scorekeep
```

输出：

```
{
    "SamplingRuleRecord": {
        "SamplingRule": {
            "RuleName": "polling-scorekeep",
            "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/polling-
scorekeep",
            "ResourceARN": "*",
            "Priority": 5000,
            "FixedRate": 0.003,
            "ReservoirSize": 0,
            "ServiceName": "Scorekeep",
            "ServiceType": "*",
            "Host": "*",
            "HTTPMethod": "GET",
            "URLPath": "/api/state/*",
            "Version": 1,
            "Attributes": {}
        },
        "CreatedAt": 1530574399.0,
        "ModifiedAt": 1530574399.0
    }
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSamplingRule。

get-encryption-config

以下代码示例演示了如何使用 get-encryption-config。

Amazon CLI

检索加密配置

以下 get-encryption-config 示例检索 Amazon X-Ray 数据的当前加密配置。

```
aws xray get-encryption-config
```

输出：

```
{
    "EncryptionConfig": {
        "KeyId": "ae4aa6d49-a4d8-9df9-a475-4ff6d7898456",
        "Status": "ACTIVE",
        "Type": "NONE"
    }
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetEncryptionConfig。

get-group

以下代码示例演示了如何使用 get-group。

Amazon CLI

检索组

以下 get-group 示例显示指定组资源的详细信息。这些详细信息包括组名称、组 ARN 以及定义该组标准的筛选表达式。也可以按 ARN 检索组。

```
aws xray get-group \
    --group-name "AdminGroup"
```

输出：

```
{
    "Group": [
        {
            "GroupName": "AdminGroup",
            "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/
AdminGroup/123456789",
            "FilterExpression": "service(\"mydomain.com\") {fault OR error}"
        }
    ]
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroup。

get-groups

以下代码示例演示了如何使用 get-groups。

Amazon CLI

检索所有组

以下示例显示所有活动组的详细信息。

```
aws xray get-groups
```

输出：

```
{
    "Groups": [
        {
            "GroupName": "AdminGroup",
            "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/
AdminGroup/123456789",
            "FilterExpression": "service(\"example.com\") {fault OR error}"
        },
        {
            "GroupName": "SDETGroup",
            "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/
SDETGroup/987654321",
            "FilterExpression": "responsetime > 2"
        }
    ]
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetGroups。

get-sampling-rules

以下代码示例演示了如何使用 get-sampling-rules。

Amazon CLI

检索所有采样规则

以下 get-sampling-rules 示例显示所有可用采样规则的详细信息：

```
aws xray get-sampling-rules
```

输出：

```
{
    "SamplingRuleRecords": [
        {
            "SamplingRule": {
                "RuleName": "Default",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/Default",
                "ResourceARN": "*",
                "Priority": 10000,
                "FixedRate": 0.01,
                "ReservoirSize": 0,
                "ServiceName": "*",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 0.0,
            "ModifiedAt": 1530558121.0
        },
        {
            "SamplingRule": {
                "RuleName": "base-scorekeep",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/base-scorekeep",
                "ResourceARN": "*",
                "Priority": 9000,
                "FixedRate": 0.1,
                "ReservoirSize": 2,
                "ServiceName": "Scorekeep",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
```

```
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 1530573954.0,
            "ModifiedAt": 1530920505.0
        },
        {
            "SamplingRule": {
                "RuleName": "polling-scorekeep",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/polling-
scorekeep",
                "ResourceARN": "*",
                "Priority": 5000,
                "FixedRate": 0.003,
                "ReservoirSize": 0,
                "ServiceName": "Scorekeep",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "GET",
                "URLPath": "/api/state/*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 1530918163.0,
            "ModifiedAt": 1530918163.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的通过 X-Ray API 使用采样规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSamplingRules。

get-sampling-targets

以下代码示例演示了如何使用 get-sampling-targets。

Amazon CLI

请求抽样配额

以下 get-sampling-targets 示例请求将服务用于采样请求的规则的采样配额。来自 Amazon X-Ray 的响应包含可以使用（而不是从容器借用）的配额。

```
aws xray get-sampling-targets \
    --sampling-statistics-documents '[ { "RuleName": "base-scorekeep", "ClientID":
 "ABCDEF1234567890ABCDEF10", "Timestamp": "2018-07-07T00:20:06, "RequestCount": 110,
 "SampledCount": 20, "BorrowCount": 10 }, { "RuleName": "polling-scorekeep", 31,
 "BorrowCount": 0 } ]'
```

输出：

```
{
    "SamplingTargetDocuments": [
        {
            "RuleName": "base-scorekeep",
            "FixedRate": 0.1,
            "ReservoirQuota": 2,
            "ReservoirQuotaTTL": 1530923107.0,
            "Interval": 10
        },
        {
            "RuleName": "polling-scorekeep",
            "FixedRate": 0.003,
            "ReservoirQuota": 0,
            "ReservoirQuotaTTL": 1530923107.0,
            "Interval": 10
        }
    ],
    "LastRuleModification": 1530920505.0,
    "UnprocessedStatistics": []
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的通过 X-Ray API 使用采样规则。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetSamplingTargets。

get-service-graph

以下代码示例演示了如何使用 get-service-graph。

Amazon CLI

获取服务图

以下示例显示指定时间段内的文档，其中描述了处理传入请求的服务，以及这些请求作为结果调用的下游服务：

```
aws xray get-service-graph \
    --start-time 1568835392.0
    --end-time 1568835446.0
```

输出：

```
{
    "Services": [
        {
            "ReferenceId": 0,
            "Name": "Scorekeep",
            "Names": [
                "Scorekeep"
            ],
            "Root": true,
            "Type": "AWS::ElasticBeanstalk::Environment",
            "State": "active",
            "StartTime": 1568835392.0,
            "EndTime": 1568835446.0,
            "Edges": [
                {
                    "ReferenceId": 1,
                    "StartTime": 1568835392.0,
                    "EndTime": 1568835446.0,
                    "SummaryStatistics": {
                        "OkCount": 14,
                        "ErrorStatistics": {
                            "ThrottleCount": 0,
                            "OtherCount": 0,
                            "TotalCount": 0
                        },
                        "FaultStatistics": {
                            "OtherCount": 0,
                            "TotalCount": 0
                        },
                        "TotalCount": 14,
                        "TotalResponseTime": 0.13
                    },
                    "ResponseTimeHistogram": [
                        {
```

```
                            "Value": 0.008,
                            "Count": 1
                        },
                        {
                            "Value": 0.005,
                            "Count": 7
                        },
                        {
                            "Value": 0.009,
                            "Count": 1
                        },
                        {
                            "Value": 0.021,
                            "Count": 1
                        },
                        {
                            "Value": 0.038,
                            "Count": 1
                        },
                        {
                            "Value": 0.007,
                            "Count": 1
                        },
                        {
                            "Value": 0.006,
                            "Count": 2
                        }
                    ],
                    "Aliases": []
                },

                ... TRUNCATED FOR BREVITY ...

            ]
        }
    ],
    "StartTime": 1568835392.0,
    "EndTime": 1568835446.0,
    "ContainsOldGroupVersions": false
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的结合使用 Amazon X-Ray API 和 Amazon CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetServiceGraph。

`get-trace-summaries`

以下代码示例演示了如何使用 `get-trace-summaries`。

Amazon CLI

获取跟踪摘要

以下 `get-trace-summaries` 示例检索指定时间段内可用的跟踪的 ID 和元数据。

```
aws xray get-trace-summaries \
    --start-time 1568835392.0 \
    --end-time 1568835446.0
```

输出：

```
[
    "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/move/
VSAE93HF/GSSD2NTB/DP0PCC09",
    "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/move/
GCQ2B35P/FREELDFT/4LRE643M",
    "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/game/
VSAE93HF/GSSD2NTB/starttime/1568835513",
    "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/
move/4MQNA5NN/L99KK2RF/null"
]
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的结合使用 Amazon X-Ray API 和 Amazon CLI。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetTraceSummaries。

`put-encryption-config`

以下代码示例演示了如何使用 `put-encryption-config`。

Amazon CLI

更新加密配置

以下 put-encryption-config``example updates the encryption configuration for AWS X-Ray data to use the default AWS managed KMS key ``aws/xray。

```
aws xray put-encryption-config \
    --type KMS \
    --key-id alias/aws/xray
```

输出：

```
{
    "EncryptionConfig": {
        "KeyId": "arn:aws:kms:us-west-2:123456789012:key/c234g4e8-39e9-4gb0-84e2-
b0ea215cbba5",
        "Status": "UPDATING",
        "Type": "KMS"
    }
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutEncryptionConfig。

put-trace-segments

以下代码示例演示了如何使用 put-trace-segments。

Amazon CLI

上传分段

以下 put-trace-segments 示例将分段文档上传到 Amazon X-Ray。分段文档作为一系列 JSON 分段文档使用。

```
aws xray put-trace-segments \
    --trace-segment-documents "{\"id\":\"20312a0e2b8809f4\",\"name
\":\"DynamoDB\",\"trace_id\":\"1-5832862d-a43aafded3334a971fe312db\",
\"start_time\":1.479706157195E9,\"end_time\":1.479706157202E9,\"parent_id\":
\"79736b962fe3239e\",\"http\":{\"response\":{\"content_length\":60,\"status
\":200}},\"inferred\":true,\"aws\":{\"consistent_read\":false,\"table_name
\":\"scorekeep-session-xray\",\"operation\":\"GetItem\",\"request_id\":
```

```
\"SCAU23OM6M8FO38UASGC7785ARVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"resource_names\":
[\"scorekeep-session-xray\"]},\"origin\":\"AWS::DynamoDB::Table\"}"
```

输出：

```
{
    "UnprocessedTraceSegments": []
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的将跟踪数据发送到 Amazon X-Ray。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutTraceSegments。

update-group

以下代码示例演示了如何使用 update-group。

Amazon CLI

更新组

以下 update-group 示例更新了相关标准，系统按该标准在名为 AdminGroup 的组中接受跟踪。您可以使用组名或组 ARN 来指定所需的组。

```
aws xray update-group \
    --group-name "AdminGroup" \
    --group-arn "arn:aws:xray:us-west-2:123456789012:group/AdminGroup/123456789" \
    --filter-expression "service(\"mydomain.com\") {fault}"
```

输出：

```
{
    "GroupName": "AdminGroup",
    "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/AdminGroup/123456789",
    "FilterExpression": "service(\"mydomain.com\") {fault}"
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateGroup。

update-sampling-rule

以下代码示例演示了如何使用 update-sampling-rule。

Amazon CLI

更新采样规则

以下 update-sampling-rule 示例修改采样规则的配置。将从 JSON 文件应用规则。只有要更新的字段才是必填字段。

```
aws xray update-sampling-rule \
   --cli-input-json file://1000-default.json
```

的内容1000-default.json：

```
{
    "SamplingRuleUpdate": {
        "RuleName": "Default",
        "FixedRate": 0.01,
        "ReservoirSize": 0
    }
}
```

输出：

```
{
    "SamplingRuleRecords": [
        {
            "SamplingRule": {
                "RuleName": "Default",
                "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/
Default",
                "ResourceARN": "*",
                "Priority": 10000,
                "FixedRate": 0.01,
                "ReservoirSize": 0,
                "ServiceName": "*",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
```

```
                "Attributes": {}
            },
            "CreatedAt": 0.0,
            "ModifiedAt": 1529959993.0
        }
    ]
}
```

有关更多信息，请参阅《Amazon X-Ray 开发人员指南》中的使用 Amazon X-Ray API 配置采样、分组和加密设置。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateSamplingRule。

Amazon CLI 与 Bash 脚本结合使用的代码示例

本主题中的代码示例向您展示了如何通过 Amazon 将 Amazon Command Line Interface 与 Bash 脚本结合使用。

基础知识是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您展示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 服务结合来完成特定任务的代码示例。

某些服务包含其他示例类别，这些类别说明如何利用特定于服务的库或函数。

Services

- 使用 Amazon CLI 及 Bash 脚本的 Amazon Batch 示例
- 使用 Amazon CLI 及 Bash 脚本的 Amazon Cloud Map 示例
- 使用 Amazon CLI 及 Bash 脚本的 CloudFront 示例
- 将 Amazon CLI 与 Bash 脚本结合使用的 DynamoDB 示例
- 使用 Amazon CLI 及 Bash 脚本的 Amazon EC2 示例
- 使用 Amazon CLI 及 Bash 脚本的 HealthImaging 示例
- 使用 Amazon CLI 及 Bash 脚本的 IAM 示例
- 使用 Amazon CLI 及 Bash 脚本的 Amazon KMS 示例
- 使用 Amazon CLI 及 Bash 脚本的 Lightsail 示例

- 使用 Amazon CLI 及 Bash 脚本的 Amazon S3 示例
- 使用 Amazon CLI 及 Bash 脚本的 Amazon STS 示例

使用 Amazon CLI 及 Bash 脚本的 Amazon Batch 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon Batch 结合使用，来执行操作和实现常见场景。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 场景

场景

Batch 和 Fargate 入门

以下代码示例展示了如何：

- 为 ECS 任务创建 IAM 执行角色
- 创建托管式 Fargate 计算环境
- 创建具有优先级设置的作业队列
- 为容器化工作负载注册作业定义
- 提交并监控批量作业执行
- 在 CloudWatch Logs 中查看作业输出
- 按正确的依赖项顺序清理资源

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# AWS Batch Fargate Getting Started Script - Fixed Version
# This script demonstrates creating AWS Batch resources with Fargate orchestration
#
# HIGH SEVERITY FIXES APPLIED:
# 1. Added IAM role propagation delay after role creation
# 2. Added resource state validation before deletion attempts

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="batch-fargate-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
RANDOM_SUFFIX=$(openssl rand -hex 6)
COMPUTE_ENV_NAME="batch-fargate-compute-${RANDOM_SUFFIX}"
JOB_QUEUE_NAME="batch-fargate-queue-${RANDOM_SUFFIX}"
JOB_DEF_NAME="batch-fargate-jobdef-${RANDOM_SUFFIX}"
JOB_NAME="batch-hello-world-${RANDOM_SUFFIX}"
ROLE_NAME="BatchEcsTaskExecutionRole-${RANDOM_SUFFIX}"
TRUST_POLICY_FILE="batch-trust-policy-${RANDOM_SUFFIX}.json"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources created so far..."
```

```
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to wait for resource to be ready
wait_for_compute_env() {
    local env_name=$1
    log "Waiting for compute environment $env_name to be VALID..."

    while true; do
        local status=$(aws batch describe-compute-environments \
            --compute-environments "$env_name" \
            --query 'computeEnvironments[0].status' \
            --output text 2>/dev/null || echo "NOT_FOUND")

        if [ "$status" = "VALID" ]; then
            log "Compute environment $env_name is ready"
            break
        elif [ "$status" = "INVALID" ] || [ "$status" = "NOT_FOUND" ]; then
            log "ERROR: Compute environment $env_name failed to create properly"
            return 1
        fi

        log "Compute environment status: $status. Waiting 10 seconds..."
        sleep 10
    done
}

# Function to wait for job queue to be ready
wait_for_job_queue() {
    local queue_name=$1
    log "Waiting for job queue $queue_name to be VALID..."

    while true; do
        local state=$(aws batch describe-job-queues \
            --job-queues "$queue_name" \
            --query 'jobQueues[0].state' \
            --output text 2>/dev/null || echo "NOT_FOUND")

        if [ "$state" = "ENABLED" ]; then
            log "Job queue $queue_name is ready"
```

```
            break
        elif [ "$state" = "DISABLED" ] || [ "$state" = "NOT_FOUND" ]; then
            log "ERROR: Job queue $queue_name failed to create properly"
            return 1
        fi

        log "Job queue state: $state. Waiting 10 seconds..."
        sleep 10
    done
}

# Function to wait for job completion
wait_for_job() {
    local job_id=$1
    log "Waiting for job $job_id to complete..."

    while true; do
        local status=$(aws batch describe-jobs \
            --jobs "$job_id" \
            --query 'jobs[0].status' \
            --output text 2>/dev/null || echo "NOT_FOUND")

        if [ "$status" = "SUCCEEDED" ]; then
            log "Job $job_id completed successfully"
            break
        elif [ "$status" = "FAILED" ]; then
            log "ERROR: Job $job_id failed"
            return 1
        fi

        log "Job status: $status. Waiting 30 seconds..."
        sleep 30
    done
}

# FIXED: Added function to wait for resource state before deletion
wait_for_resource_state() {
    local resource_type=$1
    local resource_name=$2
    local expected_state=$3
    local max_attempts=30
    local attempt=0

    log "Waiting for $resource_type $resource_name to reach state: $expected_state"
```

```
    while [ $attempt -lt $max_attempts ]; do
        local current_state=""

        case $resource_type in
            "JOB_QUEUE")
                current_state=$(aws batch describe-job-queues \
                    --job-queues "$resource_name" \
                    --query 'jobQueues[0].state' \
                    --output text 2>/dev/null || echo "NOT_FOUND")
                ;;
            "COMPUTE_ENV")
                current_state=$(aws batch describe-compute-environments \
                    --compute-environments "$resource_name" \
                    --query 'computeEnvironments[0].status' \
                    --output text 2>/dev/null || echo "NOT_FOUND")
                ;;
        esac

        if [ "$current_state" = "$expected_state" ]; then
            log "$resource_type $resource_name is now in state: $expected_state"
            return 0
        fi

        log "$resource_type $resource_name state: $current_state (waiting for
 $expected_state)"
        sleep 10
        ((attempt++))
    done

    log "WARNING: $resource_type $resource_name did not reach expected state after
 $max_attempts attempts"
    return 1
}

# Cleanup function
cleanup_resources() {
    log "Starting cleanup of created resources..."

    # Clean up in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local resource="${CREATED_RESOURCES[i]}"
        local resource_type=$(echo "$resource" | cut -d: -f1)
        local resource_name=$(echo "$resource" | cut -d: -f2)
```

```
        log "Cleaning up $resource_type: $resource_name"

        case $resource_type in
            "JOB_QUEUE")
                # FIXED: Validate state before deletion
                aws batch update-job-queue --job-queue "$resource_name" --state
 DISABLED 2>/dev/null || true
                wait_for_resource_state "JOB_QUEUE" "$resource_name" "DISABLED" ||
 true
                aws batch delete-job-queue --job-queue "$resource_name" 2>/dev/null
 || true
                ;;
            "COMPUTE_ENV")
                # FIXED: Validate state before deletion
                aws batch update-compute-environment --compute-environment
 "$resource_name" --state DISABLED 2>/dev/null || true
                wait_for_resource_state "COMPUTE_ENV" "$resource_name" "DISABLED" ||
 true
                aws batch delete-compute-environment --compute-environment
 "$resource_name" 2>/dev/null || true
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_name" --policy-arn
 "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/dev/null
 || true
                aws iam delete-role --role-name "$resource_name" 2>/dev/null || true
                ;;
            "FILE")
                rm -f "$resource_name" 2>/dev/null || true
                ;;
        esac
    done

    log "Cleanup completed"
}

# Main script execution
main() {
    log "Starting AWS Batch Fargate tutorial script - Fixed Version"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    log "Getting AWS account ID..."
```

```
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Account ID: $ACCOUNT_ID"

    # Get default VPC and subnets
    log "Getting default VPC and subnets..."
    DEFAULT_VPC=$(aws ec2 describe-vpcs \
        --filters "Name=is-default,Values=true" \
        --query 'Vpcs[0].VpcId' \
        --output text)

    if [ "$DEFAULT_VPC" = "None" ] || [ "$DEFAULT_VPC" = "null" ]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi

    log "Default VPC: $DEFAULT_VPC"

    # Get subnets in the default VPC
    SUBNETS=$(aws ec2 describe-subnets \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC" \
        --query 'Subnets[*].SubnetId' \
        --output text)

    if [ -z "$SUBNETS" ]; then
        log "ERROR: No subnets found in default VPC"
        exit 1
    fi

    # Convert tab/space-separated subnets to JSON array format
    SUBNET_ARRAY=$(echo "$SUBNETS" | tr '\t ' '\n' | sed 's/^/"/;s/$/"/' | paste -sd
','  -)
    log "Subnets: $SUBNETS"
    log "Subnet array: [$SUBNET_ARRAY]"

    # Get default security group for the VPC
    DEFAULT_SG=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC" "Name=group-name,Values=default"
\
        --query 'SecurityGroups[0].GroupId' \
        --output text)

    if [ "$DEFAULT_SG" = "None" ] || [ "$DEFAULT_SG" = "null" ]; then
        log "ERROR: No default security group found in VPC"
        exit 1
```

```
    fi

    log "Default security group: $DEFAULT_SG"

    # Step 1: Create IAM execution role
    log "Step 1: Creating IAM execution role..."

    # Create trust policy document
    cat > "$TRUST_POLICY_FILE" << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
    CREATED_RESOURCES+=("FILE:$TRUST_POLICY_FILE")

    # Create the role
    aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "file://$TRUST_POLICY_FILE"
    CREATED_RESOURCES+=("IAM_ROLE:$ROLE_NAME")

    # Attach policy
    aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"

    log "IAM role created: $ROLE_NAME"

    # FIXED: Wait for IAM role propagation
    log "Waiting for IAM role propagation (15 seconds)..."
    sleep 15

    # Step 2: Create compute environment
    log "Step 2: Creating Fargate compute environment..."
```

```
aws batch create-compute-environment \
    --compute-environment-name "$COMPUTE_ENV_NAME" \
    --type MANAGED \
    --state ENABLED \
    --compute-resources "{
        \"type\": \"FARGATE\",
        \"maxvCpus\": 256,
        \"subnets\": [$SUBNET_ARRAY],
        \"securityGroupIds\": [\"$DEFAULT_SG\"]
    }"
CREATED_RESOURCES+=("COMPUTE_ENV:$COMPUTE_ENV_NAME")

# Wait for compute environment to be ready
wait_for_compute_env "$COMPUTE_ENV_NAME"

# Step 3: Create job queue
log "Step 3: Creating job queue..."

aws batch create-job-queue \
    --job-queue-name "$JOB_QUEUE_NAME" \
    --state ENABLED \
    --priority 900 \
    --compute-environment-order order=1,computeEnvironment="$COMPUTE_ENV_NAME"
CREATED_RESOURCES+=("JOB_QUEUE:$JOB_QUEUE_NAME")

# Wait for job queue to be ready
wait_for_job_queue "$JOB_QUEUE_NAME"

# Step 4: Create job definition
log "Step 4: Creating job definition..."

aws batch register-job-definition \
    --job-definition-name "$JOB_DEF_NAME" \
    --type container \
    --platform-capabilities FARGATE \
    --container-properties "{
        \"image\": \"busybox\",
        \"resourceRequirements\": [
            {\"type\": \"VCPU\", \"value\": \"0.25\"},
            {\"type\": \"MEMORY\", \"value\": \"512\"}
        ],
        \"command\": [\"echo\", \"hello world\"],
        \"networkConfiguration\": {
```

```
            \"assignPublicIp\": \"ENABLED\"
        },
        \"executionRoleArn\": \"arn:aws:iam::${ACCOUNT_ID}:role/${ROLE_NAME}\"
    }"

log "Job definition created: $JOB_DEF_NAME"

# Step 5: Submit job
log "Step 5: Submitting job..."

JOB_ID=$(aws batch submit-job \
    --job-name "$JOB_NAME" \
    --job-queue "$JOB_QUEUE_NAME" \
    --job-definition "$JOB_DEF_NAME" \
    --query 'jobId' \
    --output text)

log "Job submitted with ID: $JOB_ID"

# Step 6: Wait for job completion and view output
log "Step 6: Waiting for job completion..."
wait_for_job "$JOB_ID"

# Get log stream name
log "Getting job logs..."
LOG_STREAM=$(aws batch describe-jobs \
    --jobs "$JOB_ID" \
    --query 'jobs[0].attempts[0].taskProperties.containers[0].logStreamName' \
    --output text)

if [ "$LOG_STREAM" != "None" ] && [ "$LOG_STREAM" != "null" ]; then
    log "Log stream: $LOG_STREAM"
    log "Job output:"
    aws logs get-log-events \
        --log-group-name "/aws/batch/job" \
        --log-stream-name "$LOG_STREAM" \
        --query 'events[*].message' \
        --output text | tee -a "$LOG_FILE"
else
    log "No log stream available for job"
fi

log "Tutorial completed successfully!"
```

```
    # Show created resources
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up manually later."
        echo "To clean up manually, run the following commands:"
        echo "aws batch update-job-queue --job-queue $JOB_QUEUE_NAME --state
 DISABLED"
        echo "aws batch delete-job-queue --job-queue $JOB_QUEUE_NAME"
        echo "aws batch update-compute-environment --compute-environment
 $COMPUTE_ENV_NAME --state DISABLED"
        echo "aws batch delete-compute-environment --compute-environment
 $COMPUTE_ENV_NAME"
        echo "aws iam detach-role-policy --role-name $ROLE_NAME --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        echo "aws iam delete-role --role-name $ROLE_NAME"
    fi
}

# Run main function
main "$@"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateComputeEnvironment
  - CreateJobQueue

- DeleteComputeEnvironment
- DeleteJobQueue
- DescribeComputeEnvironments
- DescribeJobQueues
- DescribeJobs
- RegisterJobDefinition
- SubmitJob
- UpdateComputeEnvironment
- UpdateJobQueue

使用 Amazon CLI 及 Bash 脚本的 Amazon Cloud Map 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon Cloud Map 结合使用，来执行操作和实现常见场景。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 场景

场景

Cloud Map 自定义属性

以下代码示例展示了如何：

- 为基于 API 的服务发现创建 HTTP 命名空间
- 创建 DynamoDB 表并将其注册为具有自定义属性的数据服务
- 创建用于读取和写入数据的 Lambda 函数
- 将 Lambda 函数注册为具有自定义属性的服务实例，以进行基于操作的发现
- 构建使用自定义属性发现服务的客户端应用程序

- 清理所有资源，包括 Lambda 函数、DynamoDB 表和 Cloud Map 服务

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# AWS Cloud Map Tutorial Script
# This script demonstrates how to use AWS Cloud Map for service discovery with
 custom attributes

# Set up logging
LOG_FILE="cloudmap-tutorial.log"
echo "AWS Cloud Map Tutorial Script" > $LOG_FILE
echo "Started at $(date)" >> $LOG_FILE

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log commands and their output
log_cmd() {
  echo "$ $1" | tee -a $LOG_FILE
  eval "$1" | tee -a $LOG_FILE
}

# Function to handle errors
handle_error() {
  local LINE=$1
  echo "An error occurred at line $LINE" | tee -a $LOG_FILE
  echo "Resources created so far:" | tee -a $LOG_FILE
  for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource" | tee -a $LOG_FILE
  done
  echo "Attempting to clean up resources..." | tee -a $LOG_FILE
  cleanup
  exit 1
}
```

```
# Set up error handling
trap 'handle_error $LINENO' ERR

# Helper function to wait for Cloud Map operations to complete
wait_for_operation() {
  local OPERATION_ID=$1
  local TIMEOUT=300  # 5 minutes timeout
  local START_TIME=$(date +%s)

  while true; do
    local STATUS=$(aws servicediscovery get-operation --operation-id $OPERATION_ID
 --query 'Operation.Status' --output text)

    if [ "$STATUS" == "SUCCESS" ]; then
      echo "Operation completed successfully" | tee -a $LOG_FILE
      break
    elif [ "$STATUS" == "FAIL" ]; then
      echo "Operation failed" | tee -a $LOG_FILE
      return 1
    fi

    local CURRENT_TIME=$(date +%s)
    if [ $((CURRENT_TIME - START_TIME)) -gt $TIMEOUT ]; then
      echo "Operation timed out" | tee -a $LOG_FILE
      return 1
    fi

    sleep 5
  done

  return 0
}

# Function to clean up resources
cleanup() {
  echo "Cleaning up resources..." | tee -a $LOG_FILE

  # Reverse the order of created resources for proper deletion
  for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
    resource="${CREATED_RESOURCES[$i]}"
    echo "Deleting $resource..." | tee -a $LOG_FILE

    if [[ $resource == "instance:"* ]]; then
```

```
    # Extract service ID and instance ID
    SERVICE_ID=$(echo $resource | cut -d':' -f2)
    INSTANCE_ID=$(echo $resource | cut -d':' -f3)

    # Check if instance exists before trying to deregister
    INSTANCE_EXISTS=$(aws servicediscovery list-instances --service-id $SERVICE_ID
--query "Instances[?Id=='$INSTANCE_ID'].Id" --output text 2>/dev/null || echo "")
    if [[ -n "$INSTANCE_EXISTS" ]]; then
      OPERATION_ID=$(aws servicediscovery deregister-instance --service-id
$SERVICE_ID --instance-id $INSTANCE_ID --query 'OperationId' --output text)

      # Wait for deregistration to complete
      echo "Waiting for instance deregistration to complete..." | tee -a $LOG_FILE
      wait_for_operation $OPERATION_ID
    else
      echo "Instance $INSTANCE_ID already deregistered" | tee -a $LOG_FILE
    fi
  elif [[ $resource == "lambda:"* ]]; then
    # Extract function name
    FUNCTION_NAME=$(echo $resource | cut -d':' -f2)
    aws lambda delete-function --function-name $FUNCTION_NAME
  elif [[ $resource == "role:"* ]]; then
    # Extract role name
    ROLE_NAME=$(echo $resource | cut -d':' -f2)

    # Detach all policies first
    for POLICY_ARN in $(aws iam list-attached-role-policies --role-name $ROLE_NAME
--query 'AttachedPolicies[*].PolicyArn' --output text); do
      aws iam detach-role-policy --role-name $ROLE_NAME --policy-arn $POLICY_ARN
    done

    # Delete the role
    aws iam delete-role --role-name $ROLE_NAME
  elif [[ $resource == "dynamodb:"* ]]; then
    # Extract table name
    TABLE_NAME=$(echo $resource | cut -d':' -f2)
    aws dynamodb delete-table --table-name $TABLE_NAME

    # Wait for table deletion to complete
    echo "Waiting for DynamoDB table deletion to complete..." | tee -a $LOG_FILE
    aws dynamodb wait table-not-exists --table-name $TABLE_NAME
  fi
done
```

```
  # Handle services separately to ensure all instances are deregistered first
  for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
    resource="${CREATED_RESOURCES[$i]}"
    if [[ $resource == "service:"* ]]; then
      # Extract service ID
      SERVICE_ID=$(echo $resource | cut -d':' -f2)
      echo "Deleting service $SERVICE_ID..." | tee -a $LOG_FILE

      # Make sure all instances are deregistered
      INSTANCES=$(aws servicediscovery list-instances --service-id $SERVICE_ID --
query 'Instances[*].Id' --output text)
      if [[ -n "$INSTANCES" ]]; then
        echo "Service still has instances. Waiting before deletion..." | tee -a
 $LOG_FILE
        sleep 10
      fi

      # Try to delete the service
      aws servicediscovery delete-service --id $SERVICE_ID
      sleep 5
    fi
  done

  # Handle namespaces last to ensure all services are deleted first
  for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
    resource="${CREATED_RESOURCES[$i]}"
    if [[ $resource == "namespace:"* ]]; then
      # Extract namespace ID
      NAMESPACE_ID=$(echo $resource | cut -d':' -f2)
      echo "Deleting namespace $NAMESPACE_ID..." | tee -a $LOG_FILE

      # Check if namespace still has services
      SERVICES=$(aws servicediscovery list-services --filters
 "Name=NAMESPACE_ID,Values=$NAMESPACE_ID,Condition=EQ" --query 'Services[*].Id' --
output text)
      if [[ -n "$SERVICES" ]]; then
        echo "Namespace still has services. Deleting them first..." | tee -a
 $LOG_FILE
        for SERVICE_ID in $SERVICES; do
          echo "Deleting service $SERVICE_ID..." | tee -a $LOG_FILE
          aws servicediscovery delete-service --id $SERVICE_ID
        done
        sleep 5
      fi
```

```
      # Try to delete the namespace
      OPERATION_ID=$(aws servicediscovery delete-namespace --id $NAMESPACE_ID --
query 'OperationId' --output text 2>/dev/null || echo "")
      if [[ -n "$OPERATION_ID" ]]; then
        echo "Waiting for namespace deletion to complete..." | tee -a $LOG_FILE
        wait_for_operation $OPERATION_ID
      else
        echo "Failed to delete namespace or namespace already deleted" | tee -a
 $LOG_FILE
      fi
    fi
  done

  echo "Cleanup complete" | tee -a $LOG_FILE
}

# Step 1: Create an AWS Cloud Map namespace
echo "Step 1: Creating AWS Cloud Map namespace..." | tee -a $LOG_FILE

# Check if namespace already exists
NAMESPACE_ID=$(aws servicediscovery list-namespaces --query "Namespaces[?
Name=='cloudmap-tutorial'].Id" --output text)

if [[ -z "$NAMESPACE_ID" || "$NAMESPACE_ID" == "None" ]]; then
  log_cmd "aws servicediscovery create-http-namespace --name cloudmap-tutorial --
creator-request-id namespace-request"
  OPERATION_ID=$(aws servicediscovery create-http-namespace --name cloudmap-tutorial
 --creator-request-id namespace-request --query 'OperationId' --output text)

  # Wait for namespace creation to complete
  echo "Waiting for namespace creation to complete..." | tee -a $LOG_FILE
  wait_for_operation $OPERATION_ID

  # Get the namespace ID
  NAMESPACE_ID=$(aws servicediscovery list-namespaces --query "Namespaces[?
Name=='cloudmap-tutorial'].Id" --output text)
  echo "Namespace created with ID: $NAMESPACE_ID" | tee -a $LOG_FILE
else
  echo "Namespace cloudmap-tutorial already exists with ID: $NAMESPACE_ID" | tee -a
 $LOG_FILE
fi

CREATED_RESOURCES+=("namespace:$NAMESPACE_ID")
```

```
# Step 2: Create a DynamoDB table
echo "Step 2: Creating DynamoDB table..." | tee -a $LOG_FILE

# Check if table already exists
TABLE_EXISTS=$(aws dynamodb describe-table --table-name cloudmap 2>&1 || echo
 "NOT_EXISTS")

if [[ $TABLE_EXISTS == *"ResourceNotFoundException"* || $TABLE_EXISTS ==
 "NOT_EXISTS" ]]; then
  log_cmd "aws dynamodb create-table --table-name cloudmap --attribute-definitions
 AttributeName=id,AttributeType=S --key-schema AttributeName=id,KeyType=HASH --
billing-mode PAY_PER_REQUEST"

  # Wait for DynamoDB table to become active
  echo "Waiting for DynamoDB table to become active..." | tee -a $LOG_FILE
  aws dynamodb wait table-exists --table-name cloudmap
else
  echo "DynamoDB table cloudmap already exists" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("dynamodb:cloudmap")

# Step 3: Create an AWS Cloud Map data service
echo "Step 3: Creating AWS Cloud Map data service..." | tee -a $LOG_FILE

# Get all services in the namespace
echo "Listing all services in namespace $NAMESPACE_ID..." | tee -a $LOG_FILE
SERVICES=$(aws servicediscovery list-services --filters "Name=NAMESPACE_ID,Values=
$NAMESPACE_ID,Condition=EQ" --query 'Services[*].[Id,Name]' --output text)
echo "Services found: $SERVICES" | tee -a $LOG_FILE

# Check if data service already exists
DATA_SERVICE_ID=""
while read -r id name || [[ -n "$id" ]]; do
  echo "Checking service: ID=$id, Name=$name" | tee -a $LOG_FILE
  if [[ "$name" == "data-service" ]]; then
    DATA_SERVICE_ID="$id"
    break
  fi
done <<< "$SERVICES"

if [[ -z "$DATA_SERVICE_ID" ]]; then
  echo "Data service does not exist, creating it..." | tee -a $LOG_FILE
```

```
  # Create the service and capture the ID directly
  echo "$ aws servicediscovery create-service --name data-service --namespace-id
 $NAMESPACE_ID --creator-request-id data-service-request" | tee -a $LOG_FILE
  CREATE_OUTPUT=$(aws servicediscovery create-service --name data-service --
namespace-id $NAMESPACE_ID --creator-request-id data-service-request)
  echo "$CREATE_OUTPUT" | tee -a $LOG_FILE

  # Extract the service ID using AWS CLI query
  DATA_SERVICE_ID=$(aws servicediscovery list-services --filters
 "Name=NAMESPACE_ID,Values=$NAMESPACE_ID,Condition=EQ" --query "Services[?
Name=='data-service'].Id" --output text)
  echo "Data service created with ID: $DATA_SERVICE_ID" | tee -a $LOG_FILE
else
  echo "Data service already exists with ID: $DATA_SERVICE_ID" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("service:$DATA_SERVICE_ID")

# Register DynamoDB table as a service instance
echo "Registering DynamoDB table as a service instance..." | tee -a $LOG_FILE

# Check if instance already exists
INSTANCE_EXISTS=$(aws servicediscovery list-instances --service-id $DATA_SERVICE_ID
 --query "Instances[?Id=='data-instance'].Id" --output text)

if [[ -z "$INSTANCE_EXISTS" ]]; then
  log_cmd "aws servicediscovery register-instance --service-id $DATA_SERVICE_ID --
instance-id data-instance --attributes tablename=cloudmap,region=$(aws configure get
 region)"
  OPERATION_ID=$(aws servicediscovery register-instance --service-
id $DATA_SERVICE_ID --instance-id data-instance --attributes
 tablename=cloudmap,region=$(aws configure get region) --query 'OperationId' --
output text)

  # Wait for instance registration to complete
  echo "Waiting for instance registration to complete..." | tee -a $LOG_FILE
  wait_for_operation $OPERATION_ID
else
  echo "Instance data-instance already exists" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("instance:$DATA_SERVICE_ID:data-instance")

# Step 4: Create an IAM role for Lambda
```

```
echo "Step 4: Creating IAM role for Lambda..." | tee -a $LOG_FILE

# Create a trust policy for Lambda
cat > lambda-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Check if role already exists
echo "Checking if IAM role already exists..." | tee -a $LOG_FILE
ROLE_EXISTS=$(aws iam get-role --role-name cloudmap-tutorial-role 2>&1 || echo
 "NOT_EXISTS")

if [[ $ROLE_EXISTS == *"NoSuchEntity"* || $ROLE_EXISTS == "NOT_EXISTS" ]]; then
    log_cmd "aws iam create-role --role-name cloudmap-tutorial-role --assume-role-
policy-document file://lambda-trust-policy.json"
else
    echo "Role cloudmap-tutorial-role already exists, using existing role" | tee -a
 $LOG_FILE
fi

# FIXED: Create a custom policy with least privilege instead of using
 PowerUserAccess
cat > cloudmap-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
```

```
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:GetItem",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/cloudmap"
    },
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:DiscoverInstances"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Check if policy already exists
POLICY_ARN=$(aws iam list-policies --query "Policies[?
PolicyName=='CloudMapTutorialPolicy'].Arn" --output text)

if [[ -z "$POLICY_ARN" ]]; then
  echo "Creating CloudMapTutorialPolicy..." | tee -a $LOG_FILE
  echo "$ aws iam create-policy --policy-name CloudMapTutorialPolicy --policy-
document file://cloudmap-policy.json" | tee -a $LOG_FILE
  CREATE_OUTPUT=$(aws iam create-policy --policy-name CloudMapTutorialPolicy --
policy-document file://cloudmap-policy.json)
  echo "$CREATE_OUTPUT" | tee -a $LOG_FILE
  POLICY_ARN=$(aws iam list-policies --query "Policies[?
PolicyName=='CloudMapTutorialPolicy'].Arn" --output text)
else
  echo "Policy CloudMapTutorialPolicy already exists with ARN: $POLICY_ARN" | tee -a
 $LOG_FILE
fi

echo "$ aws iam attach-role-policy --role-name cloudmap-tutorial-role --policy-arn
 $POLICY_ARN" | tee -a $LOG_FILE
aws iam attach-role-policy --role-name cloudmap-tutorial-role --policy-arn
 $POLICY_ARN | tee -a $LOG_FILE
```

```
echo "$ aws iam attach-role-policy --role-name cloudmap-tutorial-role --policy-
arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole" | tee -a
 $LOG_FILE
aws iam attach-role-policy --role-name cloudmap-tutorial-role --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole | tee -a $LOG_FILE

# Wait for role to propagate
echo "Waiting for IAM role to propagate..." | tee -a $LOG_FILE
sleep 10

ROLE_ARN=$(aws iam get-role --role-name cloudmap-tutorial-role --query 'Role.Arn' --
output text)
CREATED_RESOURCES+=("role:cloudmap-tutorial-role")

# Step 5: Create an AWS Cloud Map app service
echo "Step 5: Creating AWS Cloud Map app service..." | tee -a $LOG_FILE

# Get all services in the namespace
SERVICES=$(aws servicediscovery list-services --filters "Name=NAMESPACE_ID,Values=
$NAMESPACE_ID,Condition=EQ" --query 'Services[*].[Id,Name]' --output text)

# Check if app service already exists
APP_SERVICE_ID=""
while read -r id name || [[ -n "$id" ]]; do
  if [[ "$name" == "app-service" ]]; then
    APP_SERVICE_ID="$id"
    break
  fi
done <<< "$SERVICES"

if [[ -z "$APP_SERVICE_ID" ]]; then
  echo "App service does not exist, creating it..." | tee -a $LOG_FILE
  # Create the service and capture the ID directly
  echo "$ aws servicediscovery create-service --name app-service --namespace-id
 $NAMESPACE_ID --creator-request-id app-service-request" | tee -a $LOG_FILE
  CREATE_OUTPUT=$(aws servicediscovery create-service --name app-service --
namespace-id $NAMESPACE_ID --creator-request-id app-service-request)
  echo "$CREATE_OUTPUT" | tee -a $LOG_FILE

  # Extract the service ID using AWS CLI query
  APP_SERVICE_ID=$(aws servicediscovery list-services --filters
 "Name=NAMESPACE_ID,Values=$NAMESPACE_ID,Condition=EQ" --query "Services[?
Name=='app-service'].Id" --output text)
```

```
  echo "App service created with ID: $APP_SERVICE_ID" | tee -a $LOG_FILE
else
  echo "App service already exists with ID: $APP_SERVICE_ID" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("service:$APP_SERVICE_ID")

# Step 6: Create a Lambda function to write data
echo "Step 6: Creating Lambda function to write data..." | tee -a $LOG_FILE

# Create Lambda function code
cat > writefunction.py << EOF
import boto3
import json
import random

def lambda_handler(event, context):
    # Use AWS Cloud Map to discover the DynamoDB table
    serviceclient = boto3.client('servicediscovery')

    # Discover the data service instance
    response = serviceclient.discover_instances(
        NamespaceName='cloudmap-tutorial',
        ServiceName='data-service'
    )

    # Extract table name and region from the instance attributes
    tablename = response['Instances'][0]['Attributes']['tablename']
    region = response['Instances'][0]['Attributes']['region']

    # Create DynamoDB client in the specified region
    dynamodb = boto3.resource('dynamodb', region_name=region)
    table = dynamodb.Table(tablename)

    # Write data to the table
    table.put_item(
        Item={
            'id': str(random.randint(1,100)),
            'todo': event
        }
    )

    return {
        'statusCode': 200,
```

```
        'body': json.dumps('Data written successfully!')
    }
EOF

# Zip the function code
log_cmd "zip writefunction.zip writefunction.py"

# Create the Lambda function
FUNCTION_EXISTS=$(aws lambda list-functions --query "Functions[?
FunctionName=='writefunction'].FunctionName" --output text)
if [[ -z "$FUNCTION_EXISTS" ]]; then
  log_cmd "aws lambda create-function --function-name writefunction --runtime
 python3.12 --role $ROLE_ARN --handler writefunction.lambda_handler --zip-file
 fileb://writefunction.zip --architectures x86_64"

  # Wait for the Lambda function to be active before updating
  echo "Waiting for Lambda function to become active..." | tee -a $LOG_FILE
  function_state="Pending"
  while [ "$function_state" == "Pending" ]; do
      sleep 5
      function_state=$(aws lambda get-function --function-name writefunction --query
 'Configuration.State' --output text)
      echo "Current function state: $function_state" | tee -a $LOG_FILE
  done

  # Update the function timeout
  log_cmd "aws lambda update-function-configuration --function-name writefunction --
timeout 5"
else
  echo "Lambda function writefunction already exists" | tee -a $LOG_FILE
fi
CREATED_RESOURCES+=("lambda:writefunction")

# Step 7: Register the Lambda write function as an AWS Cloud Map service instance
echo "Step 7: Registering Lambda write function as a service instance..." | tee -a
 $LOG_FILE

# Check if instance already exists
INSTANCE_EXISTS=$(aws servicediscovery list-instances --service-id $APP_SERVICE_ID
 --query "Instances[?Id=='write-instance'].Id" --output text)

if [[ -z "$INSTANCE_EXISTS" ]]; then
  log_cmd "aws servicediscovery register-instance --service-id $APP_SERVICE_ID --
instance-id write-instance --attributes action=write,functionname=writefunction"
```

```
  OPERATION_ID=$(aws servicediscovery register-instance --service-id $APP_SERVICE_ID
 --instance-id write-instance --attributes action=write,functionname=writefunction
 --query 'OperationId' --output text)

  # Wait for instance registration to complete
  echo "Waiting for write instance registration to complete..." | tee -a $LOG_FILE
  wait_for_operation $OPERATION_ID
else
  echo "Instance write-instance already exists" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("instance:$APP_SERVICE_ID:write-instance")

# Step 8: Create a Lambda function to read data
echo "Step 8: Creating Lambda function to read data..." | tee -a $LOG_FILE

# Create Lambda function code
cat > readfunction.py << EOF
import boto3
import json

def lambda_handler(event, context):
    # Use AWS Cloud Map to discover the DynamoDB table
    serviceclient = boto3.client('servicediscovery')

    # Discover the data service instance
    response = serviceclient.discover_instances(
        NamespaceName='cloudmap-tutorial',
        ServiceName='data-service'
    )

    # Extract table name and region from the instance attributes
    tablename = response['Instances'][0]['Attributes']['tablename']
    region = response['Instances'][0]['Attributes']['region']

    # Create DynamoDB client in the specified region
    dynamodb = boto3.resource('dynamodb', region_name=region)
    table = dynamodb.Table(tablename)

    # Read data from the table
    response = table.scan()

    return {
        'statusCode': 200,
```

```
        'body': json.dumps(response['Items'])
    }
EOF

# Zip the function code
log_cmd "zip readfunction.zip readfunction.py"

# Create the Lambda function
FUNCTION_EXISTS=$(aws lambda list-functions --query "Functions[?
FunctionName=='readfunction'].FunctionName" --output text)
if [[ -z "$FUNCTION_EXISTS" ]]; then
  log_cmd "aws lambda create-function --function-name readfunction --runtime
 python3.12 --role $ROLE_ARN --handler readfunction.lambda_handler --zip-file
 fileb://readfunction.zip --architectures x86_64"

  # Wait for the Lambda function to be active before updating
  echo "Waiting for Lambda function to become active..." | tee -a $LOG_FILE
  function_state="Pending"
  while [ "$function_state" == "Pending" ]; do
      sleep 5
      function_state=$(aws lambda get-function --function-name readfunction --query
 'Configuration.State' --output text)
      echo "Current function state: $function_state" | tee -a $LOG_FILE
  done

  # Update the function timeout
  log_cmd "aws lambda update-function-configuration --function-name readfunction --
timeout 5"
else
  echo "Lambda function readfunction already exists" | tee -a $LOG_FILE
fi
CREATED_RESOURCES+=("lambda:readfunction")

# Step 9: Register the Lambda read function as an AWS Cloud Map service instance
echo "Step 9: Registering Lambda read function as a service instance..." | tee -a
 $LOG_FILE

# Check if instance already exists
INSTANCE_EXISTS=$(aws servicediscovery list-instances --service-id $APP_SERVICE_ID
 --query "Instances[?Id=='read-instance'].Id" --output text)

if [[ -z "$INSTANCE_EXISTS" ]]; then
  log_cmd "aws servicediscovery register-instance --service-id $APP_SERVICE_ID --
instance-id read-instance --attributes action=read,functionname=readfunction"
```

```
  OPERATION_ID=$(aws servicediscovery register-instance --service-id $APP_SERVICE_ID
 --instance-id read-instance --attributes action=read,functionname=readfunction --
query 'OperationId' --output text)

  # Wait for read instance registration to complete
  echo "Waiting for read instance registration to complete..." | tee -a $LOG_FILE
  wait_for_operation $OPERATION_ID
else
  echo "Instance read-instance already exists" | tee -a $LOG_FILE
fi

CREATED_RESOURCES+=("instance:$APP_SERVICE_ID:read-instance")

# Step 10: Create Python clients to interact with the services
echo "Step 10: Creating Python clients..." | tee -a $LOG_FILE

cat > writeclient.py << EOF
import boto3

serviceclient = boto3.client('servicediscovery')

response = serviceclient.discover_instances(NamespaceName='cloudmap-tutorial',
 ServiceName='app-service', QueryParameters={ 'action': 'write' })

functionname = response["Instances"][0]["Attributes"]["functionname"]

lambdaclient = boto3.client('lambda')

resp = lambdaclient.invoke(FunctionName=functionname, Payload='"This is a test
 data"')

print(resp["Payload"].read())
EOF

cat > readclient.py << EOF
import boto3

serviceclient = boto3.client('servicediscovery')

response = serviceclient.discover_instances(NamespaceName='cloudmap-tutorial',
 ServiceName='app-service', QueryParameters={ 'action': 'read' })

functionname = response["Instances"][0]["Attributes"]["functionname"]
```

```
lambdaclient = boto3.client('lambda')

resp = lambdaclient.invoke(FunctionName=functionname,
 InvocationType='RequestResponse')

print(resp["Payload"].read())
EOF

echo "Running write client..." | tee -a $LOG_FILE
log_cmd "python3 writeclient.py"

echo "Running read client..." | tee -a $LOG_FILE
log_cmd "python3 readclient.py"

# Step 11: Clean up resources
echo "Resources created:" | tee -a $LOG_FILE
for resource in "${CREATED_RESOURCES[@]}"; do
  echo "- $resource" | tee -a $LOG_FILE
done

echo "" | tee -a $LOG_FILE
echo "==========================================" | tee -a $LOG_FILE
echo "CLEANUP CONFIRMATION" | tee -a $LOG_FILE
echo "==========================================" | tee -a $LOG_FILE
echo "Do you want to clean up all created resources? (y/n): " | tee -a $LOG_FILE
read -r CLEANUP_CONFIRM
if [[ $CLEANUP_CONFIRM == "y" || $CLEANUP_CONFIRM == "Y" ]]; then
  cleanup
else
  echo "Resources were not cleaned up. You can manually clean them up later." | tee
 -a $LOG_FILE
fi

echo "Script completed at $(date)" | tee -a $LOG_FILE
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateHttpNamespace
  - CreateService
  - DeleteNamespace
  - DeleteService

- DeregisterInstance
- :GetOperation
- ListNamespaces
- ListServices
- RegisterInstance

Cloud Map 服务发现

以下代码示例展示了如何：

- 使用 Route 53 托管区集成创建公有 DNS 命名空间
- 创建可通过 DNS 查询和 API 调用发现的服务
- 使用不同的发现配置注册服务实例
- 使用 DNS 查询和 Cloud Map API 发现服务
- 使用 dig 命令和 API 调用验证服务发现
- 按正确的顺序清理资源（实例、服务、命名空间）

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# AWS Cloud Map Private Namespace Tutorial Script
# This script demonstrates how to use AWS Cloud Map for service discovery
# with DNS queries and API calls

# Exit on error
set -e

# Configuration
REGION="us-east-2"
NAMESPACE_NAME="cloudmap-tutorial.com"
```

```
LOG_FILE="cloudmap-tutorial.log"
CREATOR_REQUEST_ID=$(date +%s)

# Function to log messages
log() {
    local message="$1"
    echo "$(date '+%Y-%m-%d %H:%M:%S') - $message" | tee -a "$LOG_FILE"
}

# Function to check operation status
check_operation() {
    local operation_id="$1"
    local status=""

    log "Checking operation status for $operation_id..."

    while [[ "$status" != "SUCCESS" ]]; do
        sleep 5
        status=$(aws servicediscovery get-operation \
            --operation-id "$operation_id" \
            --region "$REGION" \
            --query "Operation.Status" \
            --output text)

        log "Operation status: $status"

        if [[ "$status" == "FAIL" ]]; then
            log "Operation failed. Exiting."
            exit 1
        fi
    done

    log "Operation completed successfully."
}

# Function to clean up resources
cleanup() {
    log "Starting cleanup process..."

    if [[ -n "$FIRST_INSTANCE_ID" ]]; then
        log "Deregistering first service instance..."
        aws servicediscovery deregister-instance \
            --service-id "$PUBLIC_SERVICE_ID" \
            --instance-id "$FIRST_INSTANCE_ID" \
```

```
            --region "$REGION" || log "Failed to deregister first instance"
    fi

    if [[ -n "$SECOND_INSTANCE_ID" ]]; then
        log "Deregistering second service instance..."
        aws servicediscovery deregister-instance \
            --service-id "$BACKEND_SERVICE_ID" \
            --instance-id "$SECOND_INSTANCE_ID" \
            --region "$REGION" || log "Failed to deregister second instance"
    fi

    if [[ -n "$PUBLIC_SERVICE_ID" ]]; then
        log "Deleting public service..."
        aws servicediscovery delete-service \
            --id "$PUBLIC_SERVICE_ID" \
            --region "$REGION" || log "Failed to delete public service"
    fi

    if [[ -n "$BACKEND_SERVICE_ID" ]]; then
        log "Deleting backend service..."
        aws servicediscovery delete-service \
            --id "$BACKEND_SERVICE_ID" \
            --region "$REGION" || log "Failed to delete backend service"
    fi

    if [[ -n "$NAMESPACE_ID" ]]; then
        log "Deleting namespace..."
        aws servicediscovery delete-namespace \
            --id "$NAMESPACE_ID" \
            --region "$REGION" || log "Failed to delete namespace"
    fi

    log "Cleanup completed."
}

# Set up trap for cleanup on script exit
trap cleanup EXIT INT TERM

# Initialize log file
> "$LOG_FILE"
log "Starting AWS Cloud Map tutorial script"

# Step 1: Create an AWS Cloud Map namespace
log "Creating AWS Cloud Map namespace: $NAMESPACE_NAME"
```

```
OPERATION_RESULT=$(aws servicediscovery create-public-dns-namespace \
    --name "$NAMESPACE_NAME" \
    --creator-request-id "cloudmap-tutorial-$CREATOR_REQUEST_ID" \
    --region "$REGION")

OPERATION_ID=$(echo "$OPERATION_RESULT" | jq -r '.OperationId')
log "Namespace creation initiated. Operation ID: $OPERATION_ID"

# Check operation status
check_operation "$OPERATION_ID"

# Get the namespace ID
log "Getting namespace ID..."
NAMESPACE_ID=$(aws servicediscovery list-namespaces \
    --region "$REGION" \
    --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" \
    --output text)

log "Namespace ID: $NAMESPACE_ID"

# Get the hosted zone ID
log "Getting Route 53 hosted zone ID..."
HOSTED_ZONE_ID=$(aws route53 list-hosted-zones-by-name \
    --dns-name "$NAMESPACE_NAME" \
    --query "HostedZones[0].Id" \
    --output text | sed 's|/hostedzone/||')

log "Hosted Zone ID: $HOSTED_ZONE_ID"

# Step 2: Create the AWS Cloud Map services
log "Creating public service..."
PUBLIC_SERVICE_RESULT=$(aws servicediscovery create-service \
    --name "public-service" \
    --namespace-id "$NAMESPACE_ID" \
    --dns-config "RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=300}]" \
    --region "$REGION")

PUBLIC_SERVICE_ID=$(echo "$PUBLIC_SERVICE_RESULT" | jq -r '.Service.Id')
log "Public service created. Service ID: $PUBLIC_SERVICE_ID"

log "Creating backend service..."
BACKEND_SERVICE_RESULT=$(aws servicediscovery create-service \
    --name "backend-service" \
    --namespace-id "$NAMESPACE_ID" \
```

```
    --type "HTTP" \
    --region "$REGION")

BACKEND_SERVICE_ID=$(echo "$BACKEND_SERVICE_RESULT" | jq -r '.Service.Id')
log "Backend service created. Service ID: $BACKEND_SERVICE_ID"

# Step 3: Register the AWS Cloud Map service instances
log "Registering first service instance..."
FIRST_INSTANCE_RESULT=$(aws servicediscovery register-instance \
    --service-id "$PUBLIC_SERVICE_ID" \
    --instance-id "first" \
    --attributes "AWS_INSTANCE_IPV4=192.168.2.1" \
    --region "$REGION")

FIRST_INSTANCE_ID="first"
FIRST_OPERATION_ID=$(echo "$FIRST_INSTANCE_RESULT" | jq -r '.OperationId')
log "First instance registration initiated. Operation ID: $FIRST_OPERATION_ID"

# Check operation status
check_operation "$FIRST_OPERATION_ID"

log "Registering second service instance..."
SECOND_INSTANCE_RESULT=$(aws servicediscovery register-instance \
    --service-id "$BACKEND_SERVICE_ID" \
    --instance-id "second" \
    --attributes "service-name=backend" \
    --region "$REGION")

SECOND_INSTANCE_ID="second"
SECOND_OPERATION_ID=$(echo "$SECOND_INSTANCE_RESULT" | jq -r '.OperationId')
log "Second instance registration initiated. Operation ID: $SECOND_OPERATION_ID"

# Check operation status
check_operation "$SECOND_OPERATION_ID"

# Step 4: Discover the AWS Cloud Map service instances
log "Getting Route 53 name servers..."
NAME_SERVERS=$(aws route53 get-hosted-zone \
    --id "$HOSTED_ZONE_ID" \
    --query "DelegationSet.NameServers[0]" \
    --output text)

log "Name server: $NAME_SERVERS"
```

```
log "Using dig to query DNS records (this will be simulated)..."
log "Command: dig @$NAME_SERVERS public-service.$NAMESPACE_NAME"
log "Expected output would show: public-service.$NAMESPACE_NAME. 300 IN A
 192.168.2.1"

log "Using AWS CLI to discover backend service instances..."
DISCOVER_RESULT=$(aws servicediscovery discover-instances \
    --namespace-name "$NAMESPACE_NAME" \
    --service-name "backend-service" \
    --region "$REGION")

log "Discovery result: $(echo "$DISCOVER_RESULT" | jq -c '.')"

# Display created resources
log "Resources created:"
log "- Namespace: $NAMESPACE_NAME (ID: $NAMESPACE_ID)"
log "- Public Service: public-service (ID: $PUBLIC_SERVICE_ID)"
log "- Backend Service: backend-service (ID: $BACKEND_SERVICE_ID)"
log "- Service Instance: first (Service: public-service)"
log "- Service Instance: second (Service: backend-service)"

# Ask user if they want to clean up resources
read -p "Do you want to clean up all created resources? (y/n): " CLEANUP_RESPONSE

if [[ "$CLEANUP_RESPONSE" == "y" || "$CLEANUP_RESPONSE" == "Y" ]]; then
    log "User confirmed cleanup. Proceeding with resource deletion."
    # Cleanup function will be called automatically on exit
else
    log "User chose not to clean up resources. Exiting without cleanup."
    trap - EXIT
    exit 0
fi
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreatePublicDnsNamespace
  - CreateService
  - DeleteNamespace
  - DeleteService
  - DeregisterInstance
  - DiscoverInstances

- :GetOperation
- ListNamespaces
- RegisterInstance

使用 Amazon CLI 及 Bash 脚本的 CloudFront 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 CloudFront 结合使用，来执行操作和实现常见场景。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 场景

场景

CloudFront 入门

以下代码示例展示了如何：

- 创建 Amazon S3 存储桶来存储内容
- 将示例内容上传到 S3 存储桶
- 创建来源访问控制（OAC）来实现安全的 S3 访问
- 创建以 S3 为源的 CloudFront 分配
- 更新 S3 存储桶策略以支持 CloudFront 访问权限
- 等待分配部署并测试内容访问权限
- 清理资源，包括分配、OAC 和 S3 存储桶

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# CloudFront Getting Started Tutorial Script
# This script creates an S3 bucket, uploads sample content, creates a CloudFront
 distribution with OAC,
# and demonstrates how to access content through CloudFront.

# Set up logging
LOG_FILE="cloudfront-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting CloudFront Getting Started Tutorial at $(date)"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created before error:"
    if [ -n "$BUCKET_NAME" ]; then
        echo "- S3 Bucket: $BUCKET_NAME"
    fi
    if [ -n "$OAC_ID" ]; then
        echo "- CloudFront Origin Access Control: $OAC_ID"
    fi
    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "- CloudFront Distribution: $DISTRIBUTION_ID"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
```

```
    echo "Cleaning up resources..."

    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "Disabling CloudFront distribution $DISTRIBUTION_ID..."

        # Get the current configuration and ETag
        ETAG=$(aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" --
query 'ETag' --output text)
        if [ $? -ne 0 ]; then
            echo "Failed to get distribution config. Continuing with cleanup..."
        else
            # Create a modified configuration with Enabled=false
            aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" | \
            jq '.DistributionConfig.Enabled = false' > temp_disabled_config.json

            # Update the distribution to disable it
            aws cloudfront update-distribution \
                --id "$DISTRIBUTION_ID" \
                --distribution-config file://<(jq '.DistributionConfig'
 temp_disabled_config.json) \
                --if-match "$ETAG"

            if [ $? -ne 0 ]; then
                echo "Failed to disable distribution. Continuing with cleanup..."
            else
                echo "Waiting for distribution to be disabled (this may take several
 minutes)..."
                aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"

                # Delete the distribution
                ETAG=$(aws cloudfront get-distribution-config --id
 "$DISTRIBUTION_ID" --query 'ETag' --output text)
                aws cloudfront delete-distribution --id "$DISTRIBUTION_ID" --if-
match "$ETAG"
                if [ $? -ne 0 ]; then
                    echo "Failed to delete distribution. You may need to delete it
 manually."
                else
                    echo "CloudFront distribution deleted."
                fi
            fi
        fi
    fi
```

```
    if [ -n "$OAC_ID" ]; then
        echo "Deleting Origin Access Control $OAC_ID..."
        OAC_ETAG=$(aws cloudfront get-origin-access-control --id "$OAC_ID" --query
 'ETag' --output text 2>/dev/null)
        if [ $? -ne 0 ]; then
            echo "Failed to get Origin Access Control ETag. You may need to delete
 it manually."
        else
            aws cloudfront delete-origin-access-control --id "$OAC_ID" --if-match
 "$OAC_ETAG"
            if [ $? -ne 0 ]; then
                echo "Failed to delete Origin Access Control. You may need to delete
 it manually."
            else
                echo "Origin Access Control deleted."
            fi
        fi
    fi

    if [ -n "$BUCKET_NAME" ]; then
        echo "Deleting S3 bucket $BUCKET_NAME and its contents..."
        aws s3 rm "s3://$BUCKET_NAME" --recursive
        if [ $? -ne 0 ]; then
            echo "Failed to remove bucket contents. Continuing with bucket
 deletion..."
        fi

        aws s3 rb "s3://$BUCKET_NAME"
        if [ $? -ne 0 ]; then
            echo "Failed to delete bucket. You may need to delete it manually."
        else
            echo "S3 bucket deleted."
        fi
    fi

    # Clean up temporary files
    rm -f temp_disabled_config.json
    rm -rf temp_content
}

# Generate a random identifier for the bucket name
RANDOM_ID=$(openssl rand -hex 6)
BUCKET_NAME="cloudfront-${RANDOM_ID}"
echo "Using bucket name: $BUCKET_NAME"
```

```
# Create a temporary directory for content
TEMP_DIR="temp_content"
mkdir -p "$TEMP_DIR/css"
if [ $? -ne 0 ]; then
    handle_error "Failed to create temporary directory"
fi

# Step 1: Create an S3 bucket
echo "Creating S3 bucket: $BUCKET_NAME"
aws s3 mb "s3://$BUCKET_NAME"
if [ $? -ne 0 ]; then
    handle_error "Failed to create S3 bucket"
fi

# Step 2: Create sample content
echo "Creating sample content..."
cat > "$TEMP_DIR/index.html" << 'EOF'
<!DOCTYPE html>
<html>
<head>
    <title>Hello World</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">
</head>
<body>
    <h1>Hello world!</h1>
</body>
</html>
EOF

cat > "$TEMP_DIR/css/styles.css" << 'EOF'
body {
    font-family: Arial, sans-serif;
    margin: 40px;
    background-color: #f5f5f5;
}
h1 {
    color: #333;
    text-align: center;
}
EOF

# Step 3: Upload content to the S3 bucket
echo "Uploading content to S3 bucket..."
```

```
aws s3 cp "$TEMP_DIR/" "s3://$BUCKET_NAME/" --recursive
if [ $? -ne 0 ]; then
    handle_error "Failed to upload content to S3 bucket"
fi

# Step 4: Create Origin Access Control
echo "Creating Origin Access Control..."
OAC_RESPONSE=$(aws cloudfront create-origin-access-control \
    --origin-access-control-config Name="oac-for-
$BUCKET_NAME",SigningProtocol=sigv4,SigningBehavior=always,OriginAccessControlOriginType=s3)

if [ $? -ne 0 ]; then
    handle_error "Failed to create Origin Access Control"
fi

OAC_ID=$(echo "$OAC_RESPONSE" | jq -r '.OriginAccessControl.Id')
echo "Created Origin Access Control with ID: $OAC_ID"

# Step 5: Create CloudFront distribution
echo "Creating CloudFront distribution..."

# Get AWS account ID for bucket policy
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)
if [ $? -ne 0 ]; then
    handle_error "Failed to get AWS account ID"
fi

# Create distribution configuration
cat > distribution-config.json << EOF
{
    "CallerReference": "cli-tutorial-$(date +%s)",
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "S3-$BUCKET_NAME",
                "DomainName": "$BUCKET_NAME.s3.amazonaws.com",
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                },
                "OriginAccessControlId": "$OAC_ID"
            }
        ]
    },
```

```
    "DefaultCacheBehavior": {
        "TargetOriginId": "S3-$BUCKET_NAME",
        "ViewerProtocolPolicy": "redirect-to-https",
        "AllowedMethods": {
            "Quantity": 2,
            "Items": ["GET", "HEAD"],
            "CachedMethods": {
                "Quantity": 2,
                "Items": ["GET", "HEAD"]
            }
        },
        "DefaultTTL": 86400,
        "MinTTL": 0,
        "MaxTTL": 31536000,
        "Compress": true,
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            }
        }
    },
    "Comment": "CloudFront distribution for tutorial",
    "Enabled": true,
    "WebACLId": ""
}
EOF

DIST_RESPONSE=$(aws cloudfront create-distribution --distribution-config file://
distribution-config.json)
if [ $? -ne 0 ]; then
    handle_error "Failed to create CloudFront distribution"
fi

DISTRIBUTION_ID=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.Id')
DOMAIN_NAME=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.DomainName')

echo "Created CloudFront distribution with ID: $DISTRIBUTION_ID"
echo "CloudFront domain name: $DOMAIN_NAME"

# Step 6: Update S3 bucket policy
echo "Updating S3 bucket policy..."
cat > bucket-policy.json << EOF
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::$BUCKET_NAME/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::$ACCOUNT_ID:distribution/
$DISTRIBUTION_ID"
                }
            }
        }
    ]
}
EOF

aws s3api put-bucket-policy --bucket "$BUCKET_NAME" --policy file://bucket-
policy.json
if [ $? -ne 0 ]; then
    handle_error "Failed to update S3 bucket policy"
fi

# Step 7: Wait for distribution to deploy
echo "Waiting for CloudFront distribution to deploy (this may take 5-10 minutes)..."
aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"
if [ $? -ne 0 ]; then
    echo "Warning: Distribution deployment wait timed out. The distribution may
 still be deploying."
else
    echo "CloudFront distribution is now deployed."
fi

# Step 8: Display access information
echo ""
echo "===== CloudFront Distribution Setup Complete ====="
echo "You can access your content at: https://$DOMAIN_NAME/index.html"
echo ""
echo "Resources created:"
echo "- S3 Bucket: $BUCKET_NAME"
```

```
echo "- CloudFront Origin Access Control: $OAC_ID"
echo "- CloudFront Distribution: $DISTRIBUTION_ID"
echo ""

# Ask user if they want to clean up resources
read -p "Do you want to clean up all resources created by this script? (y/n): "
 CLEANUP_RESPONSE
if [[ "$CLEANUP_RESPONSE" =~ ^[Yy] ]]; then
    cleanup
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually delete them later."
    echo "To access your content, visit: https://$DOMAIN_NAME/index.html"
fi

echo "Tutorial completed at $(date)"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - :CreateDistribution
  - CreateOriginAccessControl
  - :DeleteDistribution
  - DeleteOriginAccessControl
  - GetDistribution
  - GetDistributionConfig
  - GetOriginAccessControl
  - :UpdateDistribution
  - WaitDistributionDeployed

将 Amazon CLI 与 Bash 脚本结合使用的 DynamoDB 示例

以下代码示例演示如何通过 DynamoDB 将 Amazon Command Line Interface 与 Bash 脚本结合使用，来执行操作和实现常见场景。

基本功能是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 基本功能
- 操作
- 场景

基本功能

了解基本功能

以下代码示例展示了如何：

- 创建可保存电影数据的表。
- 在表中加入单一电影，获取并更新此电影。
- 向 JSON 示例文件的表中写入电影数据。
- 查询在给定年份发行的电影。
- 扫描在年份范围内发行的电影。
- 删除表中的电影后再删除表。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

DynamoDB 入门场景。

```
###############################################################################
# function dynamodb_getting_started_movies
#
```

```
# Scenario to create an Amazon DynamoDB table and perform a series of operations on
 the table.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function dynamodb_getting_started_movies() {

  source ./dynamodb_operations.sh

  key_schema_json_file="dynamodb_key_schema.json"
  attribute_definitions_json_file="dynamodb_attr_def.json"
  item_json_file="movie_item.json"
  key_json_file="movie_key.json"
  batch_json_file="batch.json"
  attribute_names_json_file="attribute_names.json"
  attributes_values_json_file="attribute_values.json"

  echo_repeat "*" 88
  echo
  echo "Welcome to the Amazon DynamoDB getting started demo."
  echo
  echo_repeat "*" 88
  echo

  local table_name
  echo -n "Enter a name for a new DynamoDB table: "
  get_input
  table_name=$get_input_result

  echo '[
  {"AttributeName": "year", "KeyType": "HASH"},
   {"AttributeName": "title", "KeyType": "RANGE"}
  ]' >"$key_schema_json_file"

  echo '[
  {"AttributeName": "year", "AttributeType": "N"},
   {"AttributeName": "title", "AttributeType": "S"}
  ]' >"$attribute_definitions_json_file"

  if dynamodb_create_table -n "$table_name" -a "$attribute_definitions_json_file" \
    -k "$key_schema_json_file" 1>/dev/null; then
    echo "Created a DynamoDB table named $table_name"
```

```
else
  errecho "The table failed to create. This demo will exit."
  clean_up
  return 1
fi

echo "Waiting for the table to become active...."

if dynamodb_wait_table_active -n "$table_name"; then
  echo "The table is now active."
else
  errecho "The table failed to become active. This demo will exit."
  cleanup "$table_name"
  return 1
fi

echo
echo_repeat "*" 88
echo

echo -n "Enter the title of a movie you want to add to the table: "
get_input
local added_title
added_title=$get_input_result

local added_year
get_int_input "What year was it released? "
added_year=$get_input_result

local rating
get_float_input "On a scale of 1 - 10, how do you rate it? " "1" "10"
rating=$get_input_result

local plot
echo -n "Summarize the plot for me: "
get_input
plot=$get_input_result

echo '{
  "year": {"N" :"'"$added_year"'"},
  "title": {"S" :  "'"$added_title"'"},
  "info": {"M" : {"plot": {"S" : "'"$plot"'"}, "rating": {"N" :"'"$rating"'"} } }
 }' >"$item_json_file"
```

```
 if dynamodb_put_item -n "$table_name" -i "$item_json_file"; then
   echo "The movie '$added_title' was successfully added to the table
'$table_name'."
 else
   errecho "Put item failed. This demo will exit."
   clean_up "$table_name"
   return 1
 fi

 echo
 echo_repeat "*" 88
 echo

 echo "Let's update your movie '$added_title'."
 get_float_input "You rated it $rating, what new rating would you give it? " "1"
"10"
 rating=$get_input_result

 echo -n "You summarized the plot as '$plot'."
 echo "What would you say now? "
 get_input
 plot=$get_input_result

 echo '{
   "year": {"N" :"'"$added_year"'"},
   "title": {"S" :  "'"$added_title"'"}
   }' >"$key_json_file"

 echo '{
   ":r": {"N" :"'"$rating"'"},
   ":p": {"S" : "'"$plot"'"}
  }' >"$item_json_file"

 local update_expression="SET info.rating = :r, info.plot = :p"

 if dynamodb_update_item -n "$table_name" -k "$key_json_file" -e
"$update_expression" -v "$item_json_file"; then
   echo "Updated '$added_title' with new attributes."
 else
   errecho "Update item failed. This demo will exit."
   clean_up "$table_name"
   return 1
 fi
```

```
  echo
  echo_repeat "*" 88
  echo

  echo "We will now use batch write to upload 150 movie entries into the table."

  local batch_json
  for batch_json in movie_files/movies_*.json; do
    echo "{ \"$table_name\" : $(<"$batch_json") }" >"$batch_json_file"
    if dynamodb_batch_write_item -i "$batch_json_file" 1>/dev/null; then
      echo "Entries in $batch_json added to table."
    else
      errecho "Batch write failed. This demo will exit."
      clean_up "$table_name"
      return 1
    fi
  done

  local title="The Lord of the Rings: The Fellowship of the Ring"
  local year="2001"

  if get_yes_no_input "Let's move on...do you want to get info about '$title'? (y/n)
"; then
    echo '{
  "year": {"N" :"'"$year"'"},
  "title": {"S" :  "'"$title"'"}
  }' >"$key_json_file"
    local info
    info=$(dynamodb_get_item -n "$table_name" -k "$key_json_file")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
      errecho "Get item failed. This demo will exit."
      clean_up "$table_name"
      return 1
    fi

    echo "Here is what I found:"
    echo "$info"
  fi

  local ask_for_year=true
  while [[ "$ask_for_year" == true ]]; do
    echo "Let's get a list of movies released in a given year."
```

```
    get_int_input "Enter a year between 1972 and 2018: " "1972" "2018"
    year=$get_input_result
    echo '{
    "#n": "year"
    }' >"$attribute_names_json_file"

    echo '{
    ":v": {"N" :"'"$year"'"}
    }' >"$attributes_values_json_file"

    response=$(dynamodb_query -n "$table_name" -k "#n=:v" -a
"$attribute_names_json_file" -v "$attributes_values_json_file")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
      errecho "Query table failed. This demo will exit."
      clean_up "$table_name"
      return 1
    fi

    echo "Here is what I found:"
    echo "$response"

    if ! get_yes_no_input "Try another year? (y/n) "; then
      ask_for_year=false
    fi
  done

  echo "Now let's scan for movies released in a range of years. Enter a year: "
  get_int_input "Enter a year between 1972 and 2018: " "1972" "2018"
  local start=$get_input_result

  get_int_input "Enter another year: " "1972" "2018"
  local end=$get_input_result

  echo '{
    "#n": "year"
    }' >"$attribute_names_json_file"

  echo '{
    ":v1": {"N" : "'"$start"'"},
    ":v2": {"N" : "'"$end"'"}
    }' >"$attributes_values_json_file"
```

```
  response=$(dynamodb_scan -n "$table_name" -f "#n BETWEEN :v1 AND :v2" -a
 "$attribute_names_json_file" -v "$attributes_values_json_file")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "Scan table failed. This demo will exit."
    clean_up "$table_name"
    return 1
  fi

  echo "Here is what I found:"
  echo "$response"

  echo
  echo_repeat "*" 88
  echo

  echo "Let's remove your movie '$added_title' from the table."

  if get_yes_no_input "Do you want to remove '$added_title'? (y/n) "; then
    echo '{
  "year": {"N" :"'"$added_year"'"},
  "title": {"S" :  "'"$added_title"'"}
  }' >"$key_json_file"

    if ! dynamodb_delete_item -n "$table_name" -k "$key_json_file"; then
      errecho "Delete item failed. This demo will exit."
      clean_up "$table_name"
      return 1
    fi
  fi

  if get_yes_no_input "Do you want to delete the table '$table_name'? (y/n) "; then
    if ! clean_up "$table_name"; then
      return 1
    fi
  else
    if ! clean_up; then
      return 1
    fi
  fi

  return 0
}
```

此场景中使用的 DynamoDB 函数。

```
###############################################################################
# function dynamodb_create_table
#
# This function creates an Amazon DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table to create.
#       -a attribute_definitions -- JSON file path of a list of attributes and their
 types.
#       -k key_schema -- JSON file path of a list of attributes and their key types.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_create_table() {
  local table_name attribute_definitions key_schema response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_create_table"
    echo "Creates an Amazon DynamoDB table with on-demand billing."
    echo " -n table_name  -- The name of the table to create."
    echo " -a attribute_definitions -- JSON file path of a list of attributes and
 their types."
    echo " -k key_schema -- JSON file path of a list of attributes and their key
 types."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:a:k:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      a) attribute_definitions="${OPTARG}" ;;
      k) key_schema="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$attribute_definitions" ]]; then
  errecho "ERROR: You must provide an attribute definitions json file path the -a
parameter."
  usage
  return 1
fi

if [[ -z "$key_schema" ]]; then
  errecho "ERROR: You must provide a key schema json file path the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    table_name:   $table_name"
iecho "    attribute_definitions:   $attribute_definitions"
iecho "    key_schema:   $key_schema"
iecho ""

response=$(aws dynamodb create-table \
  --table-name "$table_name" \
  --attribute-definitions file://"$attribute_definitions" \
  --billing-mode PAY_PER_REQUEST \
  --key-schema file://"$key_schema" )
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-table operation failed.$response"
    return 1
  fi

  return 0
}

###############################################################################
# function dynamodb_describe_table
#
# This function returns the status of a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#
#  Response:
#       - TableStatus:
#     And:
#       0 - Table is active.
#       1 - If it fails.
###############################################################################
function dynamodb_describe_table {
  local table_name
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_describe_table"
    echo "Describe the status of a DynamoDB table."
    echo "  -n table_name  -- The name of the table."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      h)
```

```
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  local table_status
    table_status=$(
      aws dynamodb describe-table \
        --table-name "$table_name" \
        --output text \
        --query 'Table.TableStatus'
    )

   local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log "$error_code"
      errecho "ERROR: AWS reports describe-table operation failed.$table_status"
      return 1
    fi

  echo "$table_status"

  return 0
}

###############################################################################
# function dynamodb_put_item
#
# This function puts an item into a DynamoDB table.
#
```

```
# Parameters:
#       -n table_name  -- The name of the table.
#       -i item  -- Path to json file containing the item values.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_put_item() {
  local table_name item response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_put_item"
    echo "Put an item into a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -i item  -- Path to json file containing the item values."
    echo ""
  }

  while getopts "n:i:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      i) item="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    item:   $item"
  iecho ""
  iecho ""

  response=$(aws dynamodb put-item \
    --table-name "$table_name" \
    --item file://"$item")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports put-item operation failed.$response"
    return 1
  fi

  return 0

}

################################################################################
# function dynamodb_update_item
#
# This function updates an item in a DynamoDB table.
#
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 update.
#       -e update expression  -- An expression that defines one or more attributes
 to be updated.
#       -v values  -- Path to json file containing the update values.
#
```

```
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_update_item() {
  local table_name keys update_expression values response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_update_item"
    echo "Update an item in a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k keys  -- Path to json file containing the keys that identify the item
 to update."
    echo " -e update expression  -- An expression that defines one or more
 attributes to be updated."
    echo " -v values  -- Path to json file containing the update values."
    echo ""
  }

  while getopts "n:k:e:v:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      k) keys="${OPTARG}" ;;
      e) update_expression="${OPTARG}" ;;
      v) values="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
```

```
    usage
    return 1
  fi

  if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
  fi
  if [[ -z "$update_expression" ]]; then
    errecho "ERROR: You must provide an update expression with the -e parameter."
    usage
    return 1
  fi

  if [[ -z "$values" ]]; then
    errecho "ERROR: You must provide a values json file path the -v parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    keys:   $keys"
  iecho "    update_expression:   $update_expression"
  iecho "    values:   $values"

  response=$(aws dynamodb update-item \
    --table-name "$table_name" \
    --key file://"$keys" \
    --update-expression "$update_expression" \
    --expression-attribute-values file://"$values")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports update-item operation failed.$response"
    return 1
  fi

  return 0

}
```

```
###############################################################################
# function dynamodb_batch_write_item
#
# This function writes a batch of items into a DynamoDB table.
#
# Parameters:
#       -i item  -- Path to json file containing the items to write.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_batch_write_item() {
  local item response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_batch_write_item"
    echo "Write a batch of items into a DynamoDB table."
    echo " -i item  -- Path to json file containing the items to write."
    echo ""
  }
  while getopts "i:h" option; do
    case "${option}" in
      i) item="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
```

```
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    item:   $item"
  iecho ""

  response=$(aws dynamodb batch-write-item \
    --request-items file://"$item")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-write-item operation failed.$response"
    return 1
  fi

  return 0
}

###############################################################################
# function dynamodb_get_item
#
# This function gets an item from a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 get.
#       [-q query]  -- Optional JMESPath query expression.
#
#  Returns:
#       The item as text output.
#  And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_get_item() {
  local table_name keys query response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # ######################################
 # Function usage explanation
 ######################################
 function usage() {
   echo "function dynamodb_get_item"
   echo "Get an item from a DynamoDB table."
   echo " -n table_name  -- The name of the table."
   echo " -k keys  -- Path to json file containing the keys that identify the item
to get."
   echo " [-q query]  -- Optional JMESPath query expression."
   echo ""
 }
 query=""
 while getopts "n:k:q:h" option; do
   case "${option}" in
     n) table_name="${OPTARG}" ;;
     k) keys="${OPTARG}" ;;
     q) query="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$table_name" ]]; then
   errecho "ERROR: You must provide a table name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$keys" ]]; then
   errecho "ERROR: You must provide a keys json file path the -k parameter."
   usage
   return 1
 fi

 if [[ -n "$query" ]]; then
```

```
    response=$(aws dynamodb get-item \
      --table-name "$table_name" \
      --key file://"$keys" \
      --output text \
      --query "$query")
  else
    response=$(
      aws dynamodb get-item \
        --table-name "$table_name" \
        --key file://"$keys" \
        --output text
    )
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports get-item operation failed.$response"
    return 1
  fi

  if [[ -n "$query" ]]; then
    echo "$response" | sed "/^\t/s/\t//1" # Remove initial tab that the JMSEPath
 query inserts on some strings.
  else
    echo "$response"
  fi

  return 0
}

###############################################################################
# function dynamodb_query
#
# This function queries a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k key_condition_expression -- The key condition expression.
#       -a attribute_names -- Path to JSON file containing the attribute names.
#       -v attribute_values -- Path to JSON file containing the attribute values.
#       [-p projection_expression]  -- Optional projection expression.
#
```

```
#  Returns:
#       The items as json output.
#  And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_query() {
  local table_name key_condition_expression attribute_names attribute_values
 projection_expression response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_query"
    echo "Query a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k key_condition_expression -- The key condition expression."
    echo " -a attribute_names -- Path to JSON file containing the attribute names."
    echo " -v attribute_values -- Path to JSON file containing the attribute
 values."
    echo " [-p projection_expression]  -- Optional projection expression."
    echo ""
  }

  while getopts "n:k:a:v:p:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      k) key_condition_expression="${OPTARG}" ;;
      a) attribute_names="${OPTARG}" ;;
      v) attribute_values="${OPTARG}" ;;
      p) projection_expression="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$key_condition_expression" ]]; then
    errecho "ERROR: You must provide a key condition expression with the -k
parameter."
    usage
    return 1
  fi

  if [[ -z "$attribute_names" ]]; then
    errecho "ERROR: You must provide a attribute names with the -a parameter."
    usage
    return 1
  fi

  if [[ -z "$attribute_values" ]]; then
    errecho "ERROR: You must provide a attribute values with the -v parameter."
    usage
    return 1
  fi

  if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb query \
      --table-name "$table_name" \
      --key-condition-expression "$key_condition_expression" \
      --expression-attribute-names file://"$attribute_names" \
      --expression-attribute-values file://"$attribute_values")
  else
    response=$(aws dynamodb query \
      --table-name "$table_name" \
      --key-condition-expression "$key_condition_expression" \
      --expression-attribute-names file://"$attribute_names" \
      --expression-attribute-values file://"$attribute_values" \
      --projection-expression "$projection_expression")
  fi

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports query operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function dynamodb_scan
#
# This function scans a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -f filter_expression  -- The filter expression.
#       -a expression_attribute_names -- Path to JSON file containing the expression
 attribute names.
#       -v expression_attribute_values -- Path to JSON file containing the
 expression attribute values.
#       [-p projection_expression]  -- Optional projection expression.
#
#  Returns:
#       The items as json output.
#  And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_scan() {
  local table_name filter_expression expression_attribute_names
 expression_attribute_values projection_expression response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_scan"
    echo "Scan a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -f filter_expression  -- The filter expression."
```

```
   echo " -a expression_attribute_names -- Path to JSON file containing the
expression attribute names."
   echo " -v expression_attribute_values -- Path to JSON file containing the
expression attribute values."
   echo " [-p projection_expression]  -- Optional projection expression."
   echo ""
 }

 while getopts "n:f:a:v:p:h" option; do
   case "${option}" in
     n) table_name="${OPTARG}" ;;
     f) filter_expression="${OPTARG}" ;;
     a) expression_attribute_names="${OPTARG}" ;;
     v) expression_attribute_values="${OPTARG}" ;;
     p) projection_expression="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$table_name" ]]; then
   errecho "ERROR: You must provide a table name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$filter_expression" ]]; then
   errecho "ERROR: You must provide a filter expression with the -f parameter."
   usage
   return 1
 fi

 if [[ -z "$expression_attribute_names" ]]; then
   errecho "ERROR: You must provide expression attribute names with the -a
parameter."
   usage
```

```
    return 1
  fi

  if [[ -z "$expression_attribute_values" ]]; then
    errecho "ERROR: You must provide expression attribute values with the -v
 parameter."
    usage
    return 1
  fi

  if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb scan \
      --table-name "$table_name" \
      --filter-expression "$filter_expression" \
      --expression-attribute-names file://"$expression_attribute_names" \
      --expression-attribute-values file://"$expression_attribute_values")
  else
    response=$(aws dynamodb scan \
      --table-name "$table_name" \
      --filter-expression "$filter_expression" \
      --expression-attribute-names file://"$expression_attribute_names" \
      --expression-attribute-values file://"$expression_attribute_values" \
      --projection-expression "$projection_expression")
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports scan operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function dynamodb_delete_item
#
# This function deletes an item from a DynamoDB table.
#
# Parameters:
```

```
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 delete.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_delete_item() {
  local table_name keys response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_delete_item"
    echo "Delete an item from a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k keys  -- Path to json file containing the keys that identify the item
 to delete."
    echo ""
  }
  while getopts "n:k:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      k) keys="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    keys:   $keys"
  iecho ""

  response=$(aws dynamodb delete-item \
    --table-name "$table_name" \
    --key file://"$keys")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-item operation failed.$response"
    return 1
  fi

  return 0

}

###############################################################################
# function dynamodb_delete_table
#
# This function deletes a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table to delete.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_delete_table() {
  local table_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function dynamodb_delete_table"
  echo "Deletes an Amazon DynamoDB table."
  echo " -n table_name  -- The name of the table to delete."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) table_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    table_name:   $table_name"
iecho ""

response=$(aws dynamodb delete-table \
  --table-name "$table_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-table operation failed.$response"
```

```
    return 1
  fi

  return 0
}
```

此场景中使用的实用程序函数。

```
################################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
################################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
```

```
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - BatchWriteItem
  - CreateTable
  - DeleteItem
  - DeleteTable
  - DescribeTable
  - GetItem
  - PutItem
  - Query
  - Scan
  - UpdateItem

操作

`BatchGetItem`

以下代码示例演示了如何使用 `BatchGetItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_batch_get_item
#
# This function gets a batch of items from a DynamoDB table.
#
# Parameters:
#       -i item  -- Path to json file containing the keys of the items to get.
#
#  Returns:
#       The items as json output.
#  And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_batch_get_item() {
  local item response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_batch_get_item"
    echo "Get a batch of items from a DynamoDB table."
    echo " -i item  -- Path to json file containing the keys of the items to get."
    echo ""
  }
```

```
  while getopts "i:h" option; do
    case "${option}" in
      i) item="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
  fi

  response=$(aws dynamodb batch-get-item \
    --request-items file://"$item")
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-get-item operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
#########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchGetItem。

`BatchWriteItem`

以下代码示例演示了如何使用 `BatchWriteItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_batch_write_item
#
# This function writes a batch of items into a DynamoDB table.
#
# Parameters:
#       -i item  -- Path to json file containing the items to write.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_batch_write_item() {
  local item response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_batch_write_item"
    echo "Write a batch of items into a DynamoDB table."
    echo " -i item  -- Path to json file containing the items to write."
    echo ""
  }
  while getopts "i:h" option; do
    case "${option}" in
      i) item="${OPTARG}" ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    item:   $item"
  iecho ""

  response=$(aws dynamodb batch-write-item \
    --request-items file://"$item")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-write-item operation failed.$response"
    return 1
  fi

  return 0
}
```

本示例中使用的实用程序函数。

```
########################################################################
# function iecho
```

```
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
```

```
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 BatchWriteItem。

`CreateTable`

以下代码示例演示了如何使用 `CreateTable`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_create_table
#
# This function creates an Amazon DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table to create.
#       -a attribute_definitions -- JSON file path of a list of attributes and their
 types.
#       -k key_schema -- JSON file path of a list of attributes and their key types.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function dynamodb_create_table() {
  local table_name attribute_definitions key_schema response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_create_table"
    echo "Creates an Amazon DynamoDB table with on-demand billing."
    echo " -n table_name  -- The name of the table to create."
    echo " -a attribute_definitions -- JSON file path of a list of attributes and
 their types."
    echo " -k key_schema -- JSON file path of a list of attributes and their key
 types."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:a:k:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      a) attribute_definitions="${OPTARG}" ;;
      k) key_schema="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$attribute_definitions" ]]; then
```

```
    errecho "ERROR: You must provide an attribute definitions json file path the -a
 parameter."
    usage
    return 1
  fi

  if [[ -z "$key_schema" ]]; then
    errecho "ERROR: You must provide a key schema json file path the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    attribute_definitions:   $attribute_definitions"
  iecho "    key_schema:   $key_schema"
  iecho ""

  response=$(aws dynamodb create-table \
    --table-name "$table_name" \
    --attribute-definitions file://"$attribute_definitions" \
    --billing-mode PAY_PER_REQUEST \
    --key-schema file://"$key_schema" )

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-table operation failed.$response"
    return 1
  fi

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
```

```
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
```

```
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateTable。

`DeleteItem`

以下代码示例演示了如何使用 `DeleteItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_delete_item
#
# This function deletes an item from a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 delete.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_delete_item() {
  local table_name keys response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # ######################################
 # Function usage explanation
 ######################################
 function usage() {
   echo "function dynamodb_delete_item"
   echo "Delete an item from a DynamoDB table."
   echo " -n table_name  -- The name of the table."
   echo " -k keys  -- Path to json file containing the keys that identify the item
to delete."
   echo ""
 }
 while getopts "n:k:h" option; do
   case "${option}" in
     n) table_name="${OPTARG}" ;;
     k) keys="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$table_name" ]]; then
   errecho "ERROR: You must provide a table name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$keys" ]]; then
   errecho "ERROR: You must provide a keys json file path the -k parameter."
   usage
   return 1
 fi

 iecho "Parameters:\n"
 iecho "    table_name:   $table_name"
 iecho "    keys:   $keys"
 iecho ""
```

```
  response=$(aws dynamodb delete-item \
    --table-name "$table_name" \
    --key file://"$keys")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-item operation failed.$response"
    return 1
  fi

  return 0

}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
```

```
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteItem。

DeleteTable

以下代码示例演示了如何使用 DeleteTable。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_delete_table
#
# This function deletes a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table to delete.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_delete_table() {
  local table_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function dynamodb_delete_table"
    echo "Deletes an Amazon DynamoDB table."
    echo " -n table_name  -- The name of the table to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho ""

  response=$(aws dynamodb delete-table \
    --table-name "$table_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-table operation failed.$response"
    return 1
  fi

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
```

```
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
```

```
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteTable。

`DescribeTable`

以下代码示例演示了如何使用 `DescribeTable`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function dynamodb_describe_table
#
# This function returns the status of a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#
#  Response:
#       - TableStatus:
#     And:
#       0 - Table is active.
#       1 - If it fails.
##############################################################################
function dynamodb_describe_table {
  local table_name
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
  function usage() {
```

```
  echo "function dynamodb_describe_table"
  echo "Describe the status of a DynamoDB table."
  echo "  -n table_name  -- The name of the table."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) table_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

local table_status
  table_status=$(
    aws dynamodb describe-table \
      --table-name "$table_name" \
      --output text \
      --query 'Table.TableStatus'
  )

 local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log "$error_code"
    errecho "ERROR: AWS reports describe-table operation failed.$table_status"
    return 1
  fi
```

```
  echo "$table_status"

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
```

```
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeTable。

`GetItem`

以下代码示例演示了如何使用 `GetItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_get_item
#
# This function gets an item from a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 get.
#       [-q query]  -- Optional JMESPath query expression.
#
#  Returns:
#       The item as text output.
#  And:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function dynamodb_get_item() {
  local table_name keys query response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_get_item"
    echo "Get an item from a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k keys  -- Path to json file containing the keys that identify the item
 to get."
    echo " [-q query]  -- Optional JMESPath query expression."
    echo ""
  }
  query=""
  while getopts "n:k:q:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      k) keys="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$keys" ]]; then
```

```
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
  fi

  if [[ -n "$query" ]]; then
    response=$(aws dynamodb get-item \
      --table-name "$table_name" \
      --key file://"$keys" \
      --output text \
      --query "$query")
  else
    response=$(
      aws dynamodb get-item \
        --table-name "$table_name" \
        --key file://"$keys" \
        --output text
    )
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports get-item operation failed.$response"
    return 1
  fi

  if [[ -n "$query" ]]; then
    echo "$response" | sed "/^\t/s/\t//1" # Remove initial tab that the JMSEPath
 query inserts on some strings.
  else
    echo "$response"
  fi

  return 0
}
```

本示例中使用的实用程序函数。

```
########################################################################
# function errecho
```

```
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetItem。

`ListTables`

以下代码示例演示了如何使用 `ListTables`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_list_tables
#
# This function lists all the tables in a DynamoDB.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_list_tables() {
  response=$(aws dynamodb list-tables \
    --output text \
    --query "TableNames")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-write-item operation failed.$response"
    return 1
  fi

  echo "$response" | tr -s "[:space:]" "\n"

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
```

```
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListTables。

`PutItem`

以下代码示例演示了如何使用 `PutItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_put_item
#
# This function puts an item into a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -i item  -- Path to json file containing the item values.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_put_item() {
  local table_name item response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
  ######################################
```

```
function usage() {
  echo "function dynamodb_put_item"
  echo "Put an item into a DynamoDB table."
  echo " -n table_name  -- The name of the table."
  echo " -i item  -- Path to json file containing the item values."
  echo ""
}

while getopts "n:i:h" option; do
  case "${option}" in
    n) table_name="${OPTARG}" ;;
    i) item="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$item" ]]; then
  errecho "ERROR: You must provide an item with the -i parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    table_name:   $table_name"
iecho "    item:   $item"
iecho ""
iecho ""

response=$(aws dynamodb put-item \
```

```
    --table-name "$table_name" \
    --item file://"$item")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports put-item operation failed.$response"
    return 1
  fi

  return 0

}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
```

```
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutItem。

Query

下面的代码示例演示如何使用 Query。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function dynamodb_query
#
# This function queries a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k key_condition_expression -- The key condition expression.
#       -a attribute_names -- Path to JSON file containing the attribute names.
#       -v attribute_values -- Path to JSON file containing the attribute values.
#       [-p projection_expression]  -- Optional projection expression.
#
#  Returns:
#       The items as json output.
#  And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function dynamodb_query() {
  local table_name key_condition_expression attribute_names attribute_values
 projection_expression response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_query"
    echo "Query a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k key_condition_expression -- The key condition expression."
    echo " -a attribute_names -- Path to JSON file containing the attribute names."
    echo " -v attribute_values -- Path to JSON file containing the attribute
 values."
```

```
  echo " [-p projection_expression]  -- Optional projection expression."
  echo ""
}

while getopts "n:k:a:v:p:h" option; do
  case "${option}" in
    n) table_name="${OPTARG}" ;;
    k) key_condition_expression="${OPTARG}" ;;
    a) attribute_names="${OPTARG}" ;;
    v) attribute_values="${OPTARG}" ;;
    p) projection_expression="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$key_condition_expression" ]]; then
  errecho "ERROR: You must provide a key condition expression with the -k
parameter."
  usage
  return 1
fi

if [[ -z "$attribute_names" ]]; then
  errecho "ERROR: You must provide a attribute names with the -a parameter."
  usage
  return 1
fi

if [[ -z "$attribute_values" ]]; then
```

```
    errecho "ERROR: You must provide a attribute values with the -v parameter."
    usage
    return 1
  fi

  if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb query \
      --table-name "$table_name" \
      --key-condition-expression "$key_condition_expression" \
      --expression-attribute-names file://"$attribute_names" \
      --expression-attribute-values file://"$attribute_values")
  else
    response=$(aws dynamodb query \
      --table-name "$table_name" \
      --key-condition-expression "$key_condition_expression" \
      --expression-attribute-names file://"$attribute_names" \
      --expression-attribute-values file://"$attribute_values" \
      --projection-expression "$projection_expression")
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports query operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
```

```
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Query。

Scan

以下代码示例演示了如何使用 Scan。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function dynamodb_scan
#
# This function scans a DynamoDB table.
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -f filter_expression  -- The filter expression.
#       -a expression_attribute_names -- Path to JSON file containing the expression
 attribute names.
#       -v expression_attribute_values -- Path to JSON file containing the
 expression attribute values.
#       [-p projection_expression]  -- Optional projection expression.
#
#  Returns:
#       The items as json output.
#  And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function dynamodb_scan() {
  local table_name filter_expression expression_attribute_names
 expression_attribute_values projection_expression response
  local option OPTARG # Required to use getopts command in a function.

  # ######################################
  # Function usage explanation
  ######################################
  function usage() {
    echo "function dynamodb_scan"
    echo "Scan a DynamoDB table."
```

```
   echo " -n table_name  -- The name of the table."
   echo " -f filter_expression  -- The filter expression."
   echo " -a expression_attribute_names -- Path to JSON file containing the
expression attribute names."
   echo " -v expression_attribute_values -- Path to JSON file containing the
expression attribute values."
   echo " [-p projection_expression]  -- Optional projection expression."
   echo ""
 }

 while getopts "n:f:a:v:p:h" option; do
   case "${option}" in
     n) table_name="${OPTARG}" ;;
     f) filter_expression="${OPTARG}" ;;
     a) expression_attribute_names="${OPTARG}" ;;
     v) expression_attribute_values="${OPTARG}" ;;
     p) projection_expression="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$table_name" ]]; then
   errecho "ERROR: You must provide a table name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$filter_expression" ]]; then
   errecho "ERROR: You must provide a filter expression with the -f parameter."
   usage
   return 1
 fi

 if [[ -z "$expression_attribute_names" ]]; then
```

```
    errecho "ERROR: You must provide expression attribute names with the -a
 parameter."
    usage
    return 1
  fi

  if [[ -z "$expression_attribute_values" ]]; then
    errecho "ERROR: You must provide expression attribute values with the -v
 parameter."
    usage
    return 1
  fi

  if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb scan \
      --table-name "$table_name" \
      --filter-expression "$filter_expression" \
      --expression-attribute-names file://"$expression_attribute_names" \
      --expression-attribute-values file://"$expression_attribute_values")
  else
    response=$(aws dynamodb scan \
      --table-name "$table_name" \
      --filter-expression "$filter_expression" \
      --expression-attribute-names file://"$expression_attribute_names" \
      --expression-attribute-values file://"$expression_attribute_values" \
      --projection-expression "$projection_expression")
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports scan operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi
```

```
  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 Scan。

`UpdateItem`

以下代码示例演示了如何使用 `UpdateItem`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function dynamodb_update_item
#
# This function updates an item in a DynamoDB table.
#
#
# Parameters:
#       -n table_name  -- The name of the table.
#       -k keys  -- Path to json file containing the keys that identify the item to
 update.
#       -e update expression  -- An expression that defines one or more attributes
 to be updated.
#       -v values  -- Path to json file containing the update values.
#
#  Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function dynamodb_update_item() {
  local table_name keys update_expression values response
  local option OPTARG # Required to use getopts command in a function.

  ######################################
  # Function usage explanation
```

```
 ######################################
 function usage() {
   echo "function dynamodb_update_item"
   echo "Update an item in a DynamoDB table."
   echo " -n table_name  -- The name of the table."
   echo " -k keys  -- Path to json file containing the keys that identify the item
to update."
   echo " -e update expression  -- An expression that defines one or more
attributes to be updated."
   echo " -v values  -- Path to json file containing the update values."
   echo ""
 }

 while getopts "n:k:e:v:h" option; do
   case "${option}" in
     n) table_name="${OPTARG}" ;;
     k) keys="${OPTARG}" ;;
     e) update_expression="${OPTARG}" ;;
     v) values="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$table_name" ]]; then
   errecho "ERROR: You must provide a table name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$keys" ]]; then
   errecho "ERROR: You must provide a keys json file path the -k parameter."
   usage
   return 1
 fi
 if [[ -z "$update_expression" ]]; then
```

```
    errecho "ERROR: You must provide an update expression with the -e parameter."
    usage
    return 1
  fi

  if [[ -z "$values" ]]; then
    errecho "ERROR: You must provide a values json file path the -v parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    table_name:   $table_name"
  iecho "    keys:   $keys"
  iecho "    update_expression:   $update_expression"
  iecho "    values:   $values"

  response=$(aws dynamodb update-item \
    --table-name "$table_name" \
    --key file://"$keys" \
    --update-expression "$update_expression" \
    --expression-attribute-values file://"$values")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports update-item operation failed.$response"
    return 1
  fi

  return 0

}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
```

```
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
```

```
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateItem。

场景

高级全局二级索引方案

以下代码示例演示了如何使用高级全局二级索引配置。

- 创建具有多个 GSI 的表。
- 创建具有按需容量和 GSI 的表。
- 将项目放入具有多个 GSI 的表中。
- 使用不同的条件查询多个 GSI。

Amazon CLI 及 Bash 脚本

创建具有多个 GSI 的表。

```
# Create a table with multiple GSIs
aws dynamodb create-table \
    --table-name MusicLibrary \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
        AttributeName=AlbumTitle,AttributeType=S \
        AttributeName=Genre,AttributeType=S \
        AttributeName=Year,AttributeType=N \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST \
    --global-secondary-indexes \
```

```
        "[
            {
                \"IndexName\": \"AlbumIndex\",
                \"KeySchema\": [{\"AttributeName\":\"AlbumTitle\",\"KeyType\":\"HASH
\"}],
                \"Projection\": {\"ProjectionType\":\"ALL\"}
            },
            {
                \"IndexName\": \"GenreYearIndex\",
                \"KeySchema\": [
                    {\"AttributeName\":\"Genre\",\"KeyType\":\"HASH\"},
                    {\"AttributeName\":\"Year\",\"KeyType\":\"RANGE\"}
                ],
                \"Projection\": {\"ProjectionType\":\"INCLUDE\",\"NonKeyAttributes
\":[\"Artist\",\"SongTitle\"]}
            }
        ]"
```

创建具有按需容量和 GSI 的表。

```
# Create a table with on-demand capacity and GSI
aws dynamodb create-table \
    --table-name MusicOnDemand \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
        AttributeName=Genre,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST \
    --global-secondary-indexes \
        "[
            {
                \"IndexName\": \"GenreIndex\",
                \"KeySchema\": [{\"AttributeName\":\"Genre\",\"KeyType\":\"HASH\"}],
                \"Projection\": {\"ProjectionType\":\"ALL\"}
            }
        ]"
```

将项目放入具有多个 GSI 的表中。

```
# Add items to MusicLibrary table
aws dynamodb put-item \
    --table-name MusicLibrary \
    --item '{
        "Artist": {"S": "The Beatles"},
        "SongTitle": {"S": "Hey Jude"},
        "AlbumTitle": {"S": "Past Masters"},
        "Genre": {"S": "Rock"},
        "Year": {"N": "1968"}
    }'

aws dynamodb put-item \
    --table-name MusicLibrary \
    --item '{
        "Artist": {"S": "Miles Davis"},
        "SongTitle": {"S": "So What"},
        "AlbumTitle": {"S": "Kind of Blue"},
        "Genre": {"S": "Jazz"},
        "Year": {"N": "1959"}
    }'
```

从具有多个 GSI 的表中查询项目。

```
# Query the AlbumIndex GSI
echo "Querying AlbumIndex GSI:"
aws dynamodb query \
    --table-name MusicLibrary \
    --index-name AlbumIndex \
    --key-condition-expression "AlbumTitle = :album" \
    --expression-attribute-values '{":album":{"S":"Kind of Blue"}}'

# Query the GenreYearIndex GSI with a range condition
echo "Querying GenreYearIndex GSI with range condition:"
aws dynamodb query \
    --table-name MusicLibrary \
    --index-name GenreYearIndex \
    --key-condition-expression "Genre = :genre AND #yr > :year" \
    --expression-attribute-names '{"#yr": "Year"}' \
    --expression-attribute-values '{":genre":{"S":"Rock"},":year":{"N":"1965"}}'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。

- CreateTable
- PutItem
- Query

创建和管理 MRSC 全局表

以下代码示例显示如何使用多区域强一致性（MRSC）创建和管理 DynamoDB 全局表。

- 创建具有多区域强一致性的表。
- 验证 MRSC 配置和副本状态。
- 通过即时读取来测试各区域间的强一致性。
- 使用 MRSC 保证执行条件写入。
- 清理 MRSC 全局表资源。

Amazon CLI 及 Bash 脚本

创建具有多区域强一致性的表。

```
# Step 1: Create a new table in us-east-2 (primary region for MRSC)
# Note: Table must be empty when enabling MRSC
aws dynamodb create-table \
    --table-name MusicTable \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST \
    --region us-east-2

# Wait for table to become active
aws dynamodb wait table-exists --table-name MusicTable --region us-east-2

# Step 2: Add replica and witness with Multi-Region Strong Consistency
# MRSC requires exactly three replicas in supported regions
aws dynamodb update-table \
    --table-name MusicTable \
    --replica-updates '[{"Create": {"RegionName": "us-east-1"}}]' \
```

```
    --global-table-witness-updates '[{"Create": {"RegionName": "us-west-2"}}]' \
    --multi-region-consistency STRONG \
    --region us-east-2
```

验证 MRSC 配置和副本状态。

```
# Verify the global table configuration and MRSC setting
aws dynamodb describe-table \
    --table-name MusicTable \
    --region us-east-2 \
    --query 'Table.
{TableName:TableName,TableStatus:TableStatus,MultiRegionConsistency:MultiRegionConsistency,R
{Region:RegionName,Status:ReplicaStatus}}'
```

通过跨区域即时读取来测试强一致性。

```
# Write an item to the primary region
aws dynamodb put-item \
    --table-name MusicTable \
    --item '{"Artist": {"S":"The Beatles"},"SongTitle": {"S":"Hey Jude"},"Album":
 {"S":"The Beatles 1967-1970"},"Year": {"N":"1968"}}' \
    --region us-east-2

# Read the item from replica region to verify strong consistency (cannot read or
 write to witness)
# No wait time needed - MRSC provides immediate consistency
echo "Reading from us-east-1 (immediate consistency):"
aws dynamodb get-item \
    --table-name MusicTable \
    --key '{"Artist": {"S":"The Beatles"},"SongTitle": {"S":"Hey Jude"}}' \
    --consistent-read \
    --region us-east-1
```

使用 MRSC 保证执行条件写入。

```
# Perform a conditional update from a different region
# This demonstrates that conditions work consistently across all regions
aws dynamodb update-item \
    --table-name MusicTable \
```

```
    --key '{"Artist": {"S":"The Beatles"},"SongTitle": {"S":"Hey Jude"}}' \
    --update-expression "SET #rating = :rating" \
    --condition-expression "attribute_exists(Artist)" \
    --expression-attribute-names '{"#rating": "Rating"}' \
    --expression-attribute-values '{":rating": {"N":"5"}}' \
    --region us-east-1
```

清理 MRSC 全局表资源。

```
# Remove replica tables (must be done before deleting the primary table)
aws dynamodb update-table \
    --table-name MusicTable \
    --replica-updates '[{"Delete": {"RegionName": "us-east-1"}}]' \
    --global-table-witness-updates '[{"Delete": {"RegionName": "us-west-2"}}]' \
    --region us-east-2

# Wait for replicas to be deleted
echo "Waiting for replicas to be deleted..."
sleep 30

# Delete the primary table
aws dynamodb delete-table \
    --table-name MusicTable \
    --region us-east-2
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - DeleteTable
  - DescribeTable
  - GetItem
  - PutItem
  - UpdateItem
  - UpdateTable

管理全局二级索引

以下代码示例演示了如何管理全局二级索引的完整生命周期。

- 创建一个具有全局二级索引的表。
- 向现有表添加新的 GSI。
- 更新（提高）GSI 温吞吐量。
- 使用 GSI 查询数据。
- 删除 GSI。

Amazon CLI 及 Bash 脚本

创建一个具有全局二级索引的表。

```
# Create a table with a GSI
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
        AttributeName=AlbumTitle,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST \
    --global-secondary-indexes \
        "IndexName=AlbumIndex,\
        KeySchema=[{AttributeName=AlbumTitle,KeyType=HASH}],\
        Projection={ProjectionType=ALL}"
```

向现有表中添加新的（按需）GSI。

```
# Add a new GSI to an existing table
aws dynamodb update-table \
    --table-name MusicCollection \
    --attribute-definitions \
        AttributeName=Genre,AttributeType=S \
    --global-secondary-index-updates \
        "[{\"Create\":{\"IndexName\":\"GenreIndex\",\
        \"KeySchema\":[{\"AttributeName\":\"Genre\",\"KeyType\":\"HASH\"}],\
        \"Projection\":{\"ProjectionType\":\"ALL\"}}}]"
```

更新（提高）GSI 温吞吐量。

```
# Increase the warm throughput of a GSI (default values are 12k reads, 4k writes)
aws dynamodb update-table \
    --table-name MusicCollection \
    --global-secondary-index-updates \
        "[{\"Update\":{\"IndexName\":\"AlbumIndex\",\
        \"WarmThroughput\":{\"ReadUnitsPerSecond\":15000,\"WriteUnitsPerSecond
\":6000}}}]"
```

使用 GSI 查询数据。

```
# Query the AlbumIndex GSI
aws dynamodb query \
    --table-name MusicCollection \
    --index-name AlbumIndex \
    --key-condition-expression "AlbumTitle = :album" \
    --expression-attribute-values '{":album":{"S":"Let It Be"}}'

# Query the GenreIndex GSI
aws dynamodb query \
    --table-name MusicCollection \
    --index-name GenreIndex \
    --key-condition-expression "Genre = :genre" \
    --expression-attribute-values '{":genre":{"S":"Jazz"}}'
```

删除 GSI。

```
# Delete a GSI from a table
aws dynamodb update-table \
    --table-name MusicCollection \
    --global-secondary-index-updates \
        "[{\"Delete\":{\"IndexName\":\"GenreIndex\"}}]"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - DeleteTable
  - 查询：

- UpdateTable

管理基于资源的策略

以下代码示例演示了如何管理 DynamoDB 表的基于资源的策略的完整生命周期。

- 使用资源策略创建表。
- 获取资源策略。
- 更新资源策略。
- 删除资源策略。

Amazon CLI 及 Bash 脚本

使用资源策略创建表。

```
# Step 1: Create a DynamoDB table
aws dynamodb create-table \
    --table-name MusicCollection \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST

# Step 2: Create a resource-based policy document
cat > policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/DynamoDBReadOnly"
      },
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
```

```
      ],
      "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection"
    }
  ]
}
EOF

# Step 3: Attach the resource-based policy to the table
aws dynamodb put-resource-policy \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --policy file://policy.json
```

获取资源策略。

```
# Get the resource-based policy attached to a table
aws dynamodb get-resource-policy \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection
```

更新资源策略。

```
# Step 1: Create an updated policy document
cat > updated-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:role/DynamoDBReadOnly",
          "arn:aws:iam::123456789012:role/DynamoDBAnalytics"
        ]
      },
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection"
    }
```

```
  ]
}
EOF

# Step 2: Update the resource-based policy on the table
aws dynamodb put-resource-policy \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
    --policy file://updated-policy.json
```

删除资源策略。

```
# Delete the resource-based policy from a table
aws dynamodb delete-resource-policy \
    --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - DeleteResourcePolicy
  - GetResourcePolicy
  - PutResourcePolicy

设置基于属性的访问控制

以下代码示例展示了如何为 DynamoDB 实现基于属性的访问权限控制（ABAC）。

- 创建用于实现 ABAC 的 IAM 策略。
- 为不同部门创建带有标签的表。
- 根据标签列出并筛选表。

Amazon CLI 及 Bash 脚本

创建用于实现 ABAC 的 IAM 策略。

```
# Step 1: Create a policy document for ABAC
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

为不同部门创建带有标签的表。

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
        Key=Environment,Value=Development

# Create another table with different tags
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

根据标签列出并筛选表。

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text | xargs
 -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn" --output
 text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
```

```
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreatePolicy
  - CreateTable
  - ListTables

处理本地二级索引

以下代码示例显示了如何使用本地二级索引创建和查询表。

- 创建具有本地二级索引（LSI）的表。
- 创建包含多个具有不同投影类型的 LSI 的表。
- 使用 LSI 查询数据。

Amazon CLI 及 Bash 脚本

创建具有本地二级索引的表。

```
# Create a table with a Local Secondary Index
aws dynamodb create-table \
    --table-name CustomerOrders \
    --attribute-definitions \
        AttributeName=CustomerID,AttributeType=S \
        AttributeName=OrderID,AttributeType=S \
        AttributeName=OrderDate,AttributeType=S \
    --key-schema \
        AttributeName=CustomerID,KeyType=HASH \
```

```
        AttributeName=OrderID,KeyType=RANGE \
    --local-secondary-indexes \
        "IndexName=OrderDateIndex,\
        KeySchema=[{AttributeName=CustomerID,KeyType=HASH},
{AttributeName=OrderDate,KeyType=RANGE}],\
        Projection={ProjectionType=ALL}" \
    --billing-mode PAY_PER_REQUEST
```

创建具有多个 LSI 的表。

```
# Create a table with multiple Local Secondary Indexes
aws dynamodb create-table \
    --table-name CustomerDetails \
    --attribute-definitions \
        AttributeName=CustomerID,AttributeType=S \
        AttributeName=Name,AttributeType=S \
        AttributeName=Email,AttributeType=S \
        AttributeName=RegistrationDate,AttributeType=S \
    --key-schema \
        AttributeName=CustomerID,KeyType=HASH \
        AttributeName=Name,KeyType=RANGE \
    --local-secondary-indexes \
        "[
            {
                \"IndexName\": \"EmailIndex\",
                \"KeySchema\": [
                    {\"AttributeName\":\"CustomerID\",\"KeyType\":\"HASH\"},
                    {\"AttributeName\":\"Email\",\"KeyType\":\"RANGE\"}
                ],
                \"Projection\": {\"ProjectionType\":\"INCLUDE\",\"NonKeyAttributes
\":[\"Address\",\"Phone\"]}
            },
            {
                \"IndexName\": \"RegistrationIndex\",
                \"KeySchema\": [
                    {\"AttributeName\":\"CustomerID\",\"KeyType\":\"HASH\"},
                    {\"AttributeName\":\"RegistrationDate\",\"KeyType\":\"RANGE\"}
                ],
                \"Projection\": {\"ProjectionType\":\"KEYS_ONLY\"}
            }
        ]" \
    --billing-mode PAY_PER_REQUEST
```

使用 LSI 查询数据。

```
# Query the OrderDateIndex LSI
aws dynamodb query \
    --table-name CustomerOrders \
    --index-name OrderDateIndex \
    --key-condition-expression "CustomerID = :custId AND OrderDate BETWEEN :date1
 AND :date2" \
    --expression-attribute-values '{
        ":custId": {"S": "C1"},
        ":date1": {"S": "2023-01-01"},
        ":date2": {"S": "2023-02-01"}
    }'

# Query with a filter expression
aws dynamodb query \
    --table-name CustomerOrders \
    --index-name OrderDateIndex \
    --key-condition-expression "CustomerID = :custId" \
    --filter-expression "Amount > :amount" \
    --expression-attribute-values '{
        ":custId": {"S": "C1"},
        ":amount": {"N": "150"}
    }'
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - 查询：

处理 Streams 和生存时间

以下代码示例演示了如何管理 DynamoDB Streams 和生存时间功能。

- 创建一个启用了 Streams 的表。
- 描述 Streams。
- 创建用于处理 Streams 的 Lambda 函数。
- 在表上启用 TTL。

- 添加具有 TTL 属性的项目。
- 描述 TTL 设置。

Amazon CLI 及 Bash 脚本

创建一个启用了 Streams 的表。

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

描述 Streams。

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"

# Describe the stream
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

为 Streams 创建 Lambda 函数。

```
# Step 1: Create an IAM role for the Lambda function
```

```
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

在表上启用 TTL。

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
```

```
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo

# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

添加具有 TTL 属性的项目。

```
# Calculate expiration time (current time + 1 day in seconds)
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

描述 TTL 设置。

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
    --table-name TTLDemo
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AttachRolePolicy
  - CreateRole
  - CreateTable
  - DescribeTable
  - DescribeTimeToLive
  - PutItem
  - UpdateTimeToLive

使用全局表和多区域复制最终一致性（MREC）

以下代码示例显示如何使用具有最终一致性的多区域复制（MREC）来管理 DynamoDB 全局表。

- 使用多区域复制（MREC）创建表。
- 从副本表中放置和获取项目。
- 逐一移除副本。
- 通过删除表来进行清除。

Amazon CLI 及 Bash 脚本

使用多区域复制来创建表。

```
# Step 1: Create a new table (MusicTable) in US East (Ohio), with DynamoDB Streams
 enabled (NEW_AND_OLD_IMAGES)
aws dynamodb create-table \
    --table-name MusicTable \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES \
    --region us-east-2
```

```
# Step 2: Create an identical MusicTable table in US East (N. Virginia)
aws dynamodb update-table --table-name MusicTable --cli-input-json \
'{
  "ReplicaUpdates":
  [
    {
      "Create": {
        "RegionName": "us-east-1"
      }
    }
  ]
}' \
--region us-east-2

# Step 3: Create a table in Europe (Ireland)
aws dynamodb update-table --table-name MusicTable --cli-input-json \
'{
  "ReplicaUpdates":
  [
    {
      "Create": {
        "RegionName": "eu-west-1"
      }
    }
  ]
}' \
--region us-east-2
```

描述多区域表。

```
# Step 4: View the list of replicas created using describe-table
aws dynamodb describe-table \
    --table-name MusicTable \
    --region us-east-2 \
    --query 'Table.
{TableName:TableName,TableStatus:TableStatus,MultiRegionConsistency:MultiRegionConsistency,R
{Region:RegionName,Status:ReplicaStatus}}'
```

将项目放在副本表中。

```
# Step 5: To verify that replication is working, add a new item to the Music table
 in US East (Ohio)
aws dynamodb put-item \
    --table-name MusicTable \
    --item '{"Artist": {"S":"item_1"},"SongTitle": {"S":"Song Value 1"}}' \
    --region us-east-2
```

从副本表中获取项目。

```
# Step 6: Wait for a few seconds, and then check to see whether the item has been
# successfully replicated to US East (N. Virginia) and Europe (Ireland)
aws dynamodb get-item \
    --table-name MusicTable \
    --key '{"Artist": {"S":"item_1"},"SongTitle": {"S":"Song Value 1"}}' \
    --region us-east-1

aws dynamodb get-item \
    --table-name MusicTable \
    --key '{"Artist": {"S":"item_1"},"SongTitle": {"S":"Song Value 1"}}' \
    --region eu-west-1
```

移除副本。

```
# Step 7: Delete the replica table in Europe (Ireland) Region
aws dynamodb update-table --table-name MusicTable --cli-input-json \
'{
  "ReplicaUpdates":
  [
    {
      "Delete": {
        "RegionName": "eu-west-1"
      }
    }
  ]
}' \
--region us-east-2

# Delete the replica table in US East (N. Virginia) Region
aws dynamodb update-table --table-name MusicTable --cli-input-json \
'{
```

```
  "ReplicaUpdates":
  [
    {
      "Delete": {
        "RegionName": "us-east-1"
      }
    }
  ]
}' \
--region us-east-2
```

通过删除表来进行清除。

```
# Clean up: Delete the primary table
aws dynamodb delete-table --table-name MusicTable --region us-east-2

echo "Global table demonstration complete."
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - DeleteTable
  - DescribeTable
  - GetItem
  - PutItem
  - UpdateTable

使用资源标记

以下代码示例显示了如何管理 DynamoDB 资源的标签。

- 使用标签创建表。
- 列出资源的标签。
- 向资源添加标签。
- 从资源中删除标签。
- 按标签筛选表。

Amazon CLI 及 Bash 脚本

使用标签创建表。

```
# Create a table with tags
aws dynamodb create-table \
    --table-name TaggedTable \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Environment,Value=Production \
        Key=Project,Value=Analytics \
        Key=Owner,Value=DataTeam
```

列出资源的标签。

```
# Get the table ARN
TABLE_ARN=$(aws dynamodb describe-table \
    --table-name TaggedTable \
    --query "Table.TableArn" \
    --output text)

# List tags for the table
aws dynamodb list-tags-of-resource \
    --resource-arn $TABLE_ARN
```

向资源添加标签。

```
# Add tags to an existing table
aws dynamodb tag-resource \
    --resource-arn $TABLE_ARN \
    --tags \
        Key=CostCenter,Value=12345 \
        Key=BackupSchedule,Value=Daily
```

从资源中删除标签。

```
# Remove tags from a table
aws dynamodb untag-resource \
    --resource-arn $TABLE_ARN \
    --tag-keys Owner BackupSchedule
```

按标签筛选表。

```
# Create another table with different tags
aws dynamodb create-table \
    --table-name AnotherTaggedTable \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Environment,Value=Development \
        Key=Project,Value=Testing

# Wait for table to become active
aws dynamodb wait table-exists --table-name AnotherTaggedTable

# List all tables
echo "All tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nFiltering tables by Environment=Production tag:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text | xargs
 -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn" --output
 text)

# Find tables with specific tag
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateTable
  - ListTagsOfResource
  - TagResource
  - UntagResource

处理表加密

以下代码示例演示了如何管理 DynamoDB 表的加密选项。

- 创建具有默认加密的表。
- 创建具有客户托管式 CMK 的表。
- 更新表加密设置。
- 描述表加密。

Amazon CLI 及 Bash 脚本

创建具有默认加密的表。

```
# Create a table with default encryption (AWS owned key)
aws dynamodb create-table \
    --table-name CustomerData \
    --attribute-definitions \
        AttributeName=CustomerID,AttributeType=S \
    --key-schema \
        AttributeName=CustomerID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS
```

创建具有客户托管式 CMK 的表。

```
# Step 1: Create a customer managed key in KMS
aws kms create-key \
    --description "Key for DynamoDB table encryption" \
```

```
    --key-usage ENCRYPT_DECRYPT \
    --customer-master-key-spec SYMMETRIC_DEFAULT

# Store the key ID for later use
KEY_ID=$(aws kms list-keys --query "Keys[?contains(KeyArn, 'Key for
 DynamoDB')].KeyId" --output text)

# Step 2: Create a table with the customer managed key
aws dynamodb create-table \
    --table-name SensitiveData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

更新表加密。

```
# Update a table to use a different KMS key
aws dynamodb update-table \
    --table-name CustomerData \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

描述表加密。

```
# Describe the table to see encryption settings
aws dynamodb describe-table \
    --table-name CustomerData \
    --query "Table.SSEDescription"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateKey
  - CreateTable
  - DescribeTable
  - UpdateTable

使用 Amazon CLI 及 Bash 脚本的 Amazon EC2 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon EC2 结合使用，来执行操作和实现常见场景。

基本功能是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 基本功能
- 操作
- 场景

基本功能

了解基本功能

以下代码示例展示了如何：

- 创建密钥对和安全组。
- 选择 Amazon 机器映像 (AMI) 和兼容的实例类型，然后创建实例。
- 停止实例，然后再重启。
- 将弹性 IP 地址与您的实例相关联。
- 使用 SSH 连接到您的实例，然后清理资源。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

在命令提示符中运行交互式场景。

```
###############################################################################
# function get_started_with_ec2_instances
#
# Runs an interactive scenario that shows how to get started using EC2 instances.
#
#     "EC2 access" permissions are needed to run this code.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function get_started_with_ec2_instances() {
  # Requires version 4 for mapfile.
  local required_version=4.0

  # Get the current Bash version
  # Check if BASH_VERSION is set
  local current_version
  if [[ -n "$BASH_VERSION" ]]; then
    # Convert BASH_VERSION to a number for comparison
    current_version=$BASH_VERSION
  else
    # Get the current Bash version using the bash command
    current_version=$(bash --version | head -n 1 | awk '{ print $4 }')
  fi

  # Convert version strings to numbers for comparison
  local required_version_num current_version_num
  required_version_num=$(echo "$required_version" | awk -F. '{ print ($1 * 10000) +
($2 * 100) + $3 }')
  current_version_num=$(echo "$current_version" | awk -F. '{ print ($1 * 10000) +
($2 * 100) + $3 }')
```

```
  # Compare versions
  if ((current_version_num < required_version_num)); then
    echo "Error: This script requires Bash version $required_version or higher."
    echo "Your current Bash version is number is $current_version."
    exit 1
  fi

  {
    if [ "$EC2_OPERATIONS_SOURCED" != "True" ]; then

      source ./ec2_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started with
instances demo."
  echo_repeat "*" 88
  echo

  echo "Let's create an RSA key pair that you can be use to securely connect to "
  echo "your EC2 instance."

  echo -n "Enter a unique name for your key: "
  get_input
  local key_name
  key_name=$get_input_result

  local temp_dir
  temp_dir=$(mktemp -d)
  local key_file_name="$temp_dir/${key_name}.pem"

  if ec2_create_keypair -n "${key_name}" -f "${key_file_name}"; then
    echo "Created a key pair $key_name and saved the private key to $key_file_name"
    echo
  else
    errecho "The key pair failed to create. This demo will exit."
    return 1
  fi

  chmod 400 "${key_file_name}"

  if yes_no_input "Do you want to list some of your key pairs? (y/n) "; then
    local keys_and_fingerprints
```

```
    keys_and_fingerprints="$(ec2_describe_key_pairs)" && {
      local image_name_and_id
      while IFS=$'\n' read -r image_name_and_id; do
        local entries
        IFS=$'\t' read -ra entries <<<"$image_name_and_id"
        echo "Found rsa key ${entries[0]} with fingerprint:"
        echo "     ${entries[1]}"
      done <<<"$keys_and_fingerprints"

    }
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's create a security group to manage access to your instance."
  echo -n "Enter a unique name for your security group: "
  get_input
  local security_group_name
  security_group_name=$get_input_result
  local security_group_id
  security_group_id=$(ec2_create_security_group -n "$security_group_name" \
    -d "Security group for EC2 instance") || {
    errecho "The security failed to create. This demo will exit."
    clean_up "$key_name" "$key_file_name"
    return 1
  }

  echo "Security group created with ID $security_group_id"
  echo

  local public_ip
  public_ip=$(curl -s http://checkip.amazonaws.com)

  echo "Let's add a rule to allow SSH only from your current IP address."
  echo "Your public IP address is $public_ip"
  echo -n "press return to add this rule to your security group."
  get_input

  if ! ec2_authorize_security_group_ingress -g "$security_group_id" -i "$public_ip"
-p tcp -f 22 -t 22; then
    errecho "The security group rules failed to update. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
```

```
  fi

  echo "Security group rules updated"

  local security_group_description
  security_group_description="$(ec2_describe_security_groups -g
 "${security_group_id}")" || {
    errecho "Failed to describe security groups. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  mapfile -t parameters <<<"$security_group_description"
  IFS=$'\t' read -ra entries <<<"${parameters[0]}"
  echo "Security group: ${entries[0]}"
  echo "    ID: ${entries[1]}"
  echo "    VPC: ${entries[2]}"
  echo "Inbound permissions:"
  IFS=$'\t' read -ra entries <<<"${parameters[1]}"
  echo "    IpProtocol: ${entries[0]}"
  echo "    FromPort: ${entries[1]}"
  echo "    ToPort: ${entries[2]}"
  echo "    CidrIp: ${parameters[2]}"

  local parameters
  parameters="$(ssm_get_parameters_by_path -p "/aws/service/ami-amazon-linux-
latest")" || {
    errecho "Failed to get parameters. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  local image_ids=""
  mapfile -t parameters <<<"$parameters"
  for image_name_and_id in "${parameters[@]}"; do
    IFS=$'\t' read -ra values <<<"$image_name_and_id"
    if [[ "${values[0]}" == *"amzn2"* ]]; then
      image_ids+="${values[1]} "
    fi
  done

  local images
  images="$(ec2_describe_images -i "$image_ids")" || {
```

```
    errecho "Failed to describe images. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  new_line_and_tab_to_list "$images"
  local images=("${list_result[@]}")

  # Get the size of the array
  local images_count=${#images[@]}

  if ((images_count == 0)); then
    errecho "No images found. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's create an instance from an Amazon Linux 2 AMI. Here are some options:"
  for ((i = 0; i < images_count; i += 3)); do
    echo "$(((i / 3) + 1)) - ${images[$i]}"
  done

  integer_input "Please enter the number of the AMI you want to use: " 1
"$((images_count / 3))"
  local choice=$get_input_result
  choice=$(((choice - 1) * 3))

  echo "Great choice."
  echo

  local architecture=${images[$((choice + 1))]}
  local image_id=${images[$((choice + 2))]}
  echo "Here are some instance types that support the ${architecture} architecture
of the image:"
  response="$(ec2_describe_instance_types -a "${architecture}" -t
"*.micro,*.small")" || {
    errecho "Failed to describe instance types. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }
```

```
  local instance_types
  mapfile -t instance_types <<<"$response"

  # Get the size of the array
  local instance_types_count=${#instance_types[@]}

  echo "Here are some options:"
  for ((i = 0; i < instance_types_count; i++)); do
    echo "$((i + 1)) - ${instance_types[$i]}"
  done

  integer_input "Which one do you want to use? " 1 "${#instance_types[@]}
"
  choice=$get_input_result
  local instance_type=${instance_types[$((choice - 1))]}
  echo "Another great choice."
  echo

  echo "Creating your instance and waiting for it to start..."
  local instance_id
  instance_id=$(ec2_run_instances -i "$image_id" -t "$instance_type" -k "$key_name"
 -s "$security_group_id") || {
    errecho "Failed to run instance. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  ec2_wait_for_instance_running -i "$instance_id"
  echo "Your instance is ready:"
  echo

  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  echo
  print_instance_details "${instance_details}"

  local public_ip
  public_ip=$(echo "${instance_details}" | awk '{print $6}')
  echo
  echo "You can use SSH to connect to your instance"
  echo "If the connection attempt times out, you might have to manually update the
 SSH ingress rule"
```

```
  echo "for your IP address in the AWS Management Console."
  connect_to_instance "$key_file_name" "$public_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's stop and start your instance to see what changes."
  echo "Stopping your instance and waiting until it's stopped..."
  ec2_stop_instances -i "$instance_id"
  ec2_wait_for_instance_stopped -i "$instance_id"

  echo "Your instance is stopped. Restarting..."

  ec2_start_instances -i "$instance_id"
  ec2_wait_for_instance_running -i "$instance_id"

  echo "Your instance is running again."
  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  print_instance_details "${instance_details}"

  public_ip=$(echo "${instance_details}" | awk '{print $6}')

  echo "Every time your instance is restarted, its public IP address changes"
  connect_to_instance "$key_file_name" "$public_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "You can allocate an Elastic IP address and associate it with your instance"
  echo "to keep a consistent IP address even when your instance restarts."

  local result
  result=$(ec2_allocate_address -d vpc) || {
    errecho "Failed to allocate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
    return 1
```

```
  }

  local elastic_ip allocation_id
  elastic_ip=$(echo "$result" | awk '{print $1}')
  allocation_id=$(echo "$result" | awk '{print $2}')

  echo "Allocated static Elastic IP address: $elastic_ip"

  local association_id
  association_id=$(ec2_associate_address -i "$instance_id" -a "$allocation_id") || {
    errecho "Failed to associate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
"$allocation_id"
    return 1
  }

  echo "Associated your Elastic IP with your instance."
  echo "You can now use SSH to connect to your instance by using the Elastic IP."
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's stop and start your instance to see what changes."
  echo "Stopping your instance and waiting until it's stopped..."
  ec2_stop_instances -i "$instance_id"
  ec2_wait_for_instance_stopped -i "$instance_id"

  echo "Your instance is stopped. Restarting..."

  ec2_start_instances -i "$instance_id"
  ec2_wait_for_instance_running -i "$instance_id"

  echo "Your instance is running again."
  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  print_instance_details "${instance_details}"

  echo "Because you have associated an Elastic IP with your instance, you can"
  echo "connect by using a consistent IP address after the instance restarts."
```

```
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  if yes_no_input "Do you want to delete the resources created in this demo: (y/n)
 "; then
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id" \
      "$allocation_id" "$association_id"
  else
    echo "The following resources were not deleted."
    echo "Key pair: $key_name"
    echo "Key file: $key_file_name"
    echo "Security group: $security_group_id"
    echo "Instance: $instance_id"
    echo "Elastic IP address: $elastic_ip"
  fi
}

###############################################################################
# function clean_up
#
# This function cleans up the created resources.
#     $1 - The name of the ec2 key pair to delete.
#     $2 - The name of the key file to delete.
#     $3 - The ID of the security group to delete.
#     $4 - The ID of the instance to terminate.
#     $5 - The ID of the elastic IP address to release.
#     $6 - The ID of the elastic IP address to disassociate.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function clean_up() {
  local result=0
  local key_pair_name=$1
  local key_file_name=$2
  local security_group_id=$3
  local instance_id=$4
  local allocation_id=$5
```

```
  local association_id=$6

  if [ -n "$association_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_disassociate_address -a "$association_id"); then
      echo "Disassociated elastic IP address with ID $association_id"
    else
      errecho "The elastic IP address disassociation failed."
      result=1
    fi
  fi

  if [ -n "$allocation_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_release_address -a "$allocation_id"); then
      echo "Released elastic IP address with ID $allocation_id"
    else
      errecho "The elastic IP address release failed."
      result=1
    fi
  fi

  if [ -n "$instance_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_terminate_instances -i "$instance_id"); then
      echo "Started terminating instance with ID $instance_id"

      ec2_wait_for_instance_terminated -i "$instance_id"
    else
      errecho "The instance terminate failed."
      result=1
    fi
  fi

  if [ -n "$security_group_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_security_group -i "$security_group_id"); then
      echo "Deleted security group with ID $security_group_id"
    else
      errecho "The security group delete failed."
      result=1
    fi
  fi
```

```
  if [ -n "$key_pair_name" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_keypair -n "$key_pair_name"); then
      echo "Deleted key pair named $key_pair_name"
    else
      errecho "The key pair delete failed."
      result=1
    fi
  fi

  if [ -n "$key_file_name" ]; then
    rm -f "$key_file_name"
  fi

  return $result
}

###############################################################################
# function ssm_get_parameters_by_path
#
# This function retrieves one or more parameters from the AWS Systems Manager
 Parameter Store
# by specifying a parameter path.
#
# Parameters:
#       -p parameter_path - The path of the parameter(s) to retrieve.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ssm_get_parameters_by_path() {
  local parameter_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ssm_get_parameters_by_path"
    echo "Retrieves one or more parameters from the AWS Systems Manager Parameter
 Store by specifying a parameter path."
    echo "  -p parameter_path - The path of the parameter(s) to retrieve."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "p:h" option; do
    case "${option}" in
      p) parameter_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$parameter_path" ]]; then
    errecho "ERROR: You must provide a parameter path with the -p parameter."
    usage
    return 1
  fi

  response=$(aws ssm get-parameters-by-path \
    --path "$parameter_path" \
    --query "Parameters[*].[Name, Value]" \
    --output text) || {
    aws_cli_error_log $?
    errecho "ERROR: AWS reports get-parameters-by-path operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

#############################################################################
# function print_instance_details
#
# This function prints the details of an Amazon Elastic Compute Cloud (Amazon EC2)
 instance.
#
# Parameters:
```

```
#       instance_details - The instance details in the format "InstanceId ImageId
 InstanceType KeyName VpcId PublicIpAddress State.Name".
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function print_instance_details() {
  local instance_details="$1"

  if [[ -z "${instance_details}" ]]; then
    echo "Error: Missing required instance details argument."
    return 1
  fi

  local instance_id image_id instance_type key_name vpc_id public_ip state
  instance_id=$(echo "${instance_details}" | awk '{print $1}')
  image_id=$(echo "${instance_details}" | awk '{print $2}')
  instance_type=$(echo "${instance_details}" | awk '{print $3}')
  key_name=$(echo "${instance_details}" | awk '{print $4}')
  vpc_id=$(echo "${instance_details}" | awk '{print $5}')
  public_ip=$(echo "${instance_details}" | awk '{print $6}')
  state=$(echo "${instance_details}" | awk '{print $7}')

  echo "    ID: ${instance_id}"
  echo "    Image ID: ${image_id}"
  echo "    Instance type: ${instance_type}"
  echo "    Key name: ${key_name}"
  echo "    VPC ID: ${vpc_id}"
  echo "    Public IP: ${public_ip}"
  echo "    State: ${state}"

  return 0
}

###############################################################################
# function connect_to_instance
#
# This function displays the public IP address of an Amazon Elastic Compute Cloud
 (Amazon EC2) instance and prompts the user to connect to the instance via SSH.
#
# Parameters:
#       $1 - The name of the key file used to connect to the instance.
#       $2 - The public IP address of the instance.
```

```
#
# Returns:
#       None
###############################################################################
function connect_to_instance() {
  local key_file_name="$1"
  local public_ip="$2"

  # Validate the input parameters
  if [[ -z "$key_file_name" ]]; then
    echo "ERROR: You must provide a key file name as the first argument." >&2
    return 1
  fi

  if [[ -z "$public_ip" ]]; then
    echo "ERROR: You must provide a public IP address as the second argument." >&2
    return 1
  fi

  # Display the public IP address and connection command
  echo "To connect, run the following command:"
  echo "    ssh -i ${key_file_name} ec2-user@${public_ip}"

  # Prompt the user to connect to the instance
  if yes_no_input "Do you want to connect now? (y/n) "; then
    echo "After you have connected, you can return to this example by typing 'exit'"
    ssh -i "${key_file_name}" ec2-user@"${public_ip}"
  fi
}

###############################################################################
# function get_input
#
# This function gets user input from the command line.
#
# Outputs:
#   User input to stdout.
#
# Returns:
#       0
###############################################################################
function get_input() {

  if [ -z "${mock_input+x}" ]; then
```

```
    read -r get_input_result
  else

    if [ "$mock_input_array_index" -lt ${#mock_input_array[@]} ]; then
      get_input_result="${mock_input_array[$mock_input_array_index]}"
      # bashsupport disable=BP2001
      # shellcheck disable=SC2206
      ((mock_input_array_index++))
      echo -n "$get_input_result"
    else
      echo "MOCK_INPUT_ARRAY has no more elements" 1>&2
      return 1
    fi
  fi

  return 0
}

################################################################################
# function yes_no_input
#
# This function requests a yes/no answer from the user, following to a prompt.
#
# Parameters:
#       $1 - The prompt.
#
# Returns:
#       0 - If yes.
#       1 - If no.
################################################################################
function yes_no_input() {
  if [ -z "$1" ]; then
    echo "Internal error yes_no_input"
    return 1
  fi

  local index=0
  local response="N"
  while [[ $index -lt 10 ]]; do
    index=$((index + 1))
    echo -n "$1"
    if ! get_input; then
      return 1
    fi
```

```
    response=$(echo "$get_input_result" | tr '[:upper:]' '[:lower:]')
    if [ "$response" = "y" ] || [ "$response" = "n" ]; then
      break
    else
      echo -e "\nPlease enter or 'y' or 'n'."
    fi
  done

  echo

  if [ "$response" = "y" ]; then
    return 0
  else
    return 1
  fi
}

################################################################################
# function integer_input
#
# This function prompts the user to enter an integer within a specified range
# and validates the input.
#
# Parameters:
#       $1 - The prompt message to display to the user.
#       $2 - The minimum value of the accepted range.
#       $3 - The maximum value of the accepted range.
#
# Returns:
#       The valid integer input from the user.
#       If the input is invalid or out of range, the function will continue
#       prompting the user until a valid input is provided.
################################################################################
function integer_input() {
  local prompt="$1"
  local min_value="$2"
  local max_value="$3"
  local input=""

  while true; do
    # Display the prompt message and wait for user input
    echo -n "$prompt"

    if ! get_input; then
```

```
      return 1
    fi

    input="$get_input_result"

    # Check if the input is a valid integer
    if [[ "$input" =~ ^-?[0-9]+$ ]]; then
      # Check if the input is within the specified range
      if ((input >= min_value && input <= max_value)); then
        return 0
      else
        echo "Error: Input, $input, must be between $min_value and $max_value."
      fi
    else
      echo "Error: Invalid input- $input. Please enter an integer."
    fi
  done
}
################################################################################
# function new_line_and_tab_to_list
#
# This function takes a string input containing newlines and tabs, and
# converts it into a list (array) of elements.
#
# Parameters:
#       $1 - The input string containing newlines and tabs.
#
# Returns:
#       The resulting list (array) is stored in the global variable
#       'list_result'.
################################################################################
function new_line_and_tab_to_list() {
  local input=$1
  export list_result

  list_result=()
  mapfile -t lines <<<"$input"
  local line
  for line in "${lines[@]}"; do
    IFS=$'\t' read -ra parameters <<<"$line"
    list_result+=("${parameters[@]}")
  done
}
```

```
###############################################################################
# function echo_repeat
#
# This function prints a string 'n' times to stdout.
#
# Parameters:
#       $1 - The string.
#       $2 - Number of times to print the string.
#
# Outputs:
#   String 'n' times to stdout.
#
# Returns:
#       0
###############################################################################
function echo_repeat() {
  local end=$2
  for ((i = 0; i < end; i++)); do
    echo -n "$1"
  done
  echo
}
```

此场景中使用的 DynamoDB 函数。

```
###############################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_create_keypair() {
  local key_pair_name file_path response
```

```
 local option OPTARG # Required to use getopts command in a function.

 # bashsupport disable=BP5008
 function usage() {
   echo "function ec2_create_keypair"
   echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-bit
RSA key pair"
   echo " and writes it to a file."
   echo "  -n key_pair_name - A key pair name."
   echo "  -f file_path - File to store the key pair."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:f:h" option; do
   case "${option}" in
     n) key_pair_name="${OPTARG}" ;;
     f) file_path="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$key_pair_name" ]]; then
   errecho "ERROR: You must provide a key name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$file_path" ]]; then
   errecho "ERROR: You must provide a file path with the -f parameter."
   usage
   return 1
 fi

 response=$(aws ec2 create-key-pair \
```

```
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
}

################################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
```

```
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
```

```
function ec2_create_security_group() {
  local security_group_name security_group_description response

  # Function to display usage information
  function usage() {
    echo "function ec2_create_security_group"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -n security_group_name - The name of the security group."
    echo "  -d security_group_description - The description of the security group."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "n:d:h" option; do
    case "${option}" in
      n) security_group_name="${OPTARG}" ;;
      d) security_group_description="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$security_group_name" ]]; then
    errecho "ERROR: You must provide a security group name with the -n parameter."
    return 1
  fi

  if [[ -z "$security_group_description" ]]; then
    errecho "ERROR: You must provide a security group description with the -d
 parameter."
    return 1
  fi

  # Create the security group
  response=$(aws ec2 create-security-group \
```

```
    --group-name "$security_group_name" \
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

###############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) security
 groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:h" option; do
    case "${option}" in
```

```
      g) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id" --
query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
#
# Parameters:
```

```
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      i) ip_address="${OPTARG}" ;;
      p) protocol="${OPTARG}" ;;
      f) from_port="${OPTARG}" ;;
      t) to_port="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
  fi

  if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
 parameter."
    usage
    return 1
  fi

  if [[ -z "$protocol" ]]; then
    errecho "ERROR: You must provide a protocol with the -p parameter."
    usage
    return 1
  fi

  if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports authorize-security-group-ingress operation failed.
$response"
```

```
    return 1
  }

  return 0
}

###############################################################################
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

################################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g., x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
################################################################################
function ec2_describe_instance_types() {
  local architecture=""
  local instance_types=""
```

```
  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-t|--
type INSTANCE_TYPE] [-h|--help]"
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
 (e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance types
 (e.g., t2.micro)"
    echo "  -h, --help                       Show this help message"
  }

  while [[ $# -gt 0 ]]; do
    case "$1" in
      -a | --architecture)
        architecture="$2"
        shift 2
        ;;
      -t | --type)
        instance_types="$2"
        shift 2
        ;;
      -h | --help)
        usage
        return 0
        ;;
      *)
        echo "Unknown argument: $1"
        return 1
        ;;
    esac
  done

  if [[ -z "$architecture" ]]; then
    errecho "Error: Architecture not specified."
    usage
    return 1
  fi

  if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
  fi
```

```
local tmp_json_file="temp_ec2.json"
echo -n '[
  {
    "Name": "processor-info.supported-architecture",
    "Values": [' >"$tmp_json_file"

local items
IFS=',' read -ra items <<<"$architecture"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
  echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
  if [[ $i -lt $((array_size - 1)) ]]; then
    echo -n ',' >>"$tmp_json_file"
  fi
done
echo -n ']},
  {
  "Name": "instance-type",
    "Values": [' >>"$tmp_json_file"
IFS=',' read -ra items <<<"$instance_types"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
  echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
  if [[ $i -lt $((array_size - 1)) ]]; then
    echo -n ',' >>"$tmp_json_file"
  fi
done

echo -n ']}]' >>"$tmp_json_file"

local response
response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
  --query 'InstanceTypes[*].[InstanceType]' --output text)

local error_code=$?

rm "$tmp_json_file"

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  echo "ERROR: AWS reports describe-instance-types operation failed."
```

```
    return 1
  fi

  echo "$response"
  return 0
}

###############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:t:k:s:c:h" option; do
```

```
    case "${option}" in
      i) image_id="${OPTARG}" ;;
      t) instance_type="${OPTARG}" ;;
      k) key_pair_name="${OPTARG}" ;;
      s) security_group_id="${OPTARG}" ;;
      c) count="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$image_id" ]]; then
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
    usage
    return 1
  fi

  if [[ -z "$instance_type" ]]; then
    errecho "ERROR: You must provide an instance type with the -t parameter."
    usage
    return 1
  fi

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -k parameter."
    usage
    return 1
  fi

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$count" ]]; then
    count=1
  fi

  response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

################################################################################
# function ec2_describe_instances
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
```

```
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
```

```
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
  }

  return 0
}

###############################################################################
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
  }
```

```
  return 0
}

################################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic Compute
 Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
################################################################################
function ec2_allocate_address() {
  local domain response

  # Function to display usage information
  function usage() {
    echo "function ec2_allocate_address"
    echo "Allocates an Elastic IP address for use with Amazon Elastic Compute Cloud
 (Amazon EC2) instances in a specific AWS Region."
    echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard')."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "d:h" option; do
    case "${option}" in
      d) domain="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$domain" ]]; then
    errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc' or
 'standard')."
    return 1
  fi

  if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
    errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
    return 1
  fi

  # Allocate the Elastic IP address
  response=$(aws ec2 allocate-address \
    --domain "$domain" \
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

###############################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to associate.
```

```
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_associate_address"
    echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 associate."
    echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:i:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      i) instance_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
```

```
    return 1
  fi

  if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
  fi

  # Associate the Elastic IP address
  response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

################################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
################################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
```

```
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a association_id - The association ID that represents the association
 of the Elastic IP address with an instance."
    echo ""
 }

 # Parse the command-line arguments
 while getopts "a:h" option; do
   case "${option}" in
     a) association_id="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$association_id" ]]; then
   errecho "ERROR: You must provide an association ID with the -a parameter."
   return 1
 fi

 response=$(aws ec2 disassociate-address \
   --association-id "$association_id") || {
   aws_cli_error_log ${?}
   errecho "ERROR: AWS reports disassociate-address operation failed."
   errecho "$response"
   return 1
 }

 return 0
}

################################################################################
# function ec2_release_address
```

```
#
# This function releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
```

```
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}

###############################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
```

```
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --output
 text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
```

```
  }

  return 0
}

###############################################################################
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
}
```

此场景中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
```

```
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - :CreateKeyPair
  - CreateSecurityGroup
  - :DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress

- RunInstances：
- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

操作

`AllocateAddress`

以下代码示例演示了如何使用 `AllocateAddress`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic Compute
 Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_allocate_address() {
  local domain response
```

```
 # Function to display usage information
 function usage() {
   echo "function ec2_allocate_address"
   echo "Allocates an Elastic IP address for use with Amazon Elastic Compute Cloud
(Amazon EC2) instances in a specific AWS Region."
   echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
'standard')."
   echo ""
 }

 # Parse the command-line arguments
 while getopts "d:h" option; do
   case "${option}" in
     d) domain="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$domain" ]]; then
   errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc' or
'standard')."
   return 1
 fi

 if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
   errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
   return 1
 fi

 # Allocate the Elastic IP address
 response=$(aws ec2 allocate-address \
   --domain "$domain" \
   --query "[PublicIp,AllocationId]" \
```

```
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

本示例中使用的实用程序函数。

```
################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
```

```
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AllocateAddress。

`AssociateAddress`

以下代码示例演示了如何使用 `AssociateAddress`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to associate.
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
#
###############################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_associate_address"
    echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 associate."
    echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:i:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      i) instance_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
```

```
    return 1
  fi

  # Associate the Elastic IP address
  response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
```

```
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssociateAddress。

`AuthorizeSecurityGroupIngress`

以下代码示例演示了如何使用 `AuthorizeSecurityGroupIngress`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
```

```
#
# Parameters:
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      i) ip_address="${OPTARG}" ;;
      p) protocol="${OPTARG}" ;;
      f) from_port="${OPTARG}" ;;
      t) to_port="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
  fi

  if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
    usage
    return 1
  fi

  if [[ -z "$protocol" ]]; then
    errecho "ERROR: You must provide a protocol with the -p parameter."
    usage
    return 1
  fi

  if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
```

```
    errecho "ERROR: AWS reports authorize-security-group-ingress operation failed.
$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AuthorizeSecurityGroupIngress。

`CreateKeyPair`

以下代码示例演示了如何使用 `CreateKeyPair`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_create_keypair() {
```

```
 local key_pair_name file_path response
 local option OPTARG # Required to use getopts command in a function.

 # bashsupport disable=BP5008
 function usage() {
   echo "function ec2_create_keypair"
   echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-bit
RSA key pair"
   echo " and writes it to a file."
   echo "  -n key_pair_name - A key pair name."
   echo "  -f file_path - File to store the key pair."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:f:h" option; do
   case "${option}" in
     n) key_pair_name="${OPTARG}" ;;
     f) file_path="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$key_pair_name" ]]; then
   errecho "ERROR: You must provide a key name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$file_path" ]]; then
   errecho "ERROR: You must provide a file path with the -f parameter."
   usage
   return 1
 fi
```

```
  response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
```

```
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateKeyPair。

`CreateSecurityGroup`

以下代码示例演示了如何使用 `CreateSecurityGroup`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
```

```
#
# Returns:
#       The ID of the created security group, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_create_security_group() {
  local security_group_name security_group_description response

  # Function to display usage information
  function usage() {
    echo "function ec2_create_security_group"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -n security_group_name - The name of the security group."
    echo "  -d security_group_description - The description of the security group."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "n:d:h" option; do
    case "${option}" in
      n) security_group_name="${OPTARG}" ;;
      d) security_group_description="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$security_group_name" ]]; then
    errecho "ERROR: You must provide a security group name with the -n parameter."
    return 1
  fi
```

```
  if [[ -z "$security_group_description" ]]; then
    errecho "ERROR: You must provide a security group description with the -d
 parameter."
    return 1
  fi

  # Create the security group
  response=$(aws ec2 create-security-group \
    --group-name "$security_group_name" \
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

本示例中使用的实用程序函数。

```
################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
```

```
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateSecurityGroup。

`DeleteKeyPair`

以下代码示例演示了如何使用 `DeleteKeyPair`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
```

```
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
```

```
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
```

```
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteKeyPair。

`DeleteSecurityGroup`

以下代码示例演示了如何使用 `DeleteSecurityGroup`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --output
 text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi
```

```
  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteSecurityGroup。

`DescribeImages`

以下代码示例演示了如何使用 `DescribeImages`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
```

```
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeImages。

`DescribeInstanceTypes`

以下代码示例演示了如何使用 `DescribeInstanceTypes`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g., x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
###############################################################################
function ec2_describe_instance_types() {
  local architecture=""
  local instance_types=""

  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-t|--
type INSTANCE_TYPE] [-h|--help]"
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
 (e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance types
 (e.g., t2.micro)"
```

```
  echo "  -h, --help                        Show this help message"
}

while [[ $# -gt 0 ]]; do
  case "$1" in
    -a | --architecture)
      architecture="$2"
      shift 2
      ;;
    -t | --type)
      instance_types="$2"
      shift 2
      ;;
    -h | --help)
      usage
      return 0
      ;;
    *)
      echo "Unknown argument: $1"
      return 1
      ;;
  esac
done

if [[ -z "$architecture" ]]; then
  errecho "Error: Architecture not specified."
  usage
  return 1
fi

if [[ -z "$instance_types" ]]; then
  errecho "Error: Instance type not specified."
  usage
  return 1
fi

local tmp_json_file="temp_ec2.json"
echo -n '[
  {
    "Name": "processor-info.supported-architecture",
    "Values": [' >"$tmp_json_file"

local items
IFS=',' read -ra items <<<"$architecture"
```

```
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done
  echo -n ']},
    {
    "Name": "instance-type",
      "Values": [' >>"$tmp_json_file"
  IFS=',' read -ra items <<<"$instance_types"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done

  echo -n ']}]' >>"$tmp_json_file"

  local response
  response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
    --query 'InstanceTypes[*].[InstanceType]' --output text)

  local error_code=$?

  rm "$tmp_json_file"

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    echo "ERROR: AWS reports describe-instance-types operation failed."
    return 1
  fi

  echo "$response"
  return 0
}
```

本示例中使用的实用程序函数。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstanceTypes。

`DescribeInstances`

以下代码示例演示了如何使用 `DescribeInstances`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_describe_instances
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
```

```
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
```

```
    return 1
  }

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeInstances。

`DescribeKeyPairs`

以下代码示例演示了如何使用 `DescribeKeyPairs`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.
```

```
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeKeyPairs。

`DescribeSecurityGroups`

以下代码示例演示了如何使用 `DescribeSecurityGroups`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) security
 groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "g:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id" --
query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
#############################################################################
```

```
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeSecurityGroups。

DisassociateAddress

以下代码示例演示了如何使用 DisassociateAddress。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a association_id - The association ID that represents the association
 of the Elastic IP address with an instance."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
```

```
      a) association_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$association_id" ]]; then
    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DisassociateAddress。

`ReleaseAddress`

以下代码示例演示了如何使用 `ReleaseAddress`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
```

```
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ReleaseAddress。

RunInstances

以下代码示例演示了如何使用 RunInstances。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
    echo "  -h - Display help."
    echo ""
  }
```

```
 # Retrieve the calling parameters.
 while getopts "i:t:k:s:c:h" option; do
   case "${option}" in
     i) image_id="${OPTARG}" ;;
     t) instance_type="${OPTARG}" ;;
     k) key_pair_name="${OPTARG}" ;;
     s) security_group_id="${OPTARG}" ;;
     c) count="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$image_id" ]]; then
   errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
   usage
   return 1
 fi

 if [[ -z "$instance_type" ]]; then
   errecho "ERROR: You must provide an instance type with the -t parameter."
   usage
   return 1
 fi

 if [[ -z "$key_pair_name" ]]; then
   errecho "ERROR: You must provide a key pair name with the -k parameter."
   usage
   return 1
 fi

 if [[ -z "$security_group_id" ]]; then
   errecho "ERROR: You must provide a security group ID with the -s parameter."
   usage
```

```
    return 1
  fi

  if [[ -z "$count" ]]; then
    count=1
  fi

  response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

本示例中使用的实用程序函数。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
```

```
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RunInstances。

`StartInstances`

以下代码示例演示了如何使用 `StartInstances`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
```

```
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StartInstances。

`StopInstances`

以下代码示例演示了如何使用 `StopInstances`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
```

```
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
```

```
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
```

```
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 StopInstances。

`TerminateInstances`

以下代码示例演示了如何使用 `TerminateInstances`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
```

```
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}
```

本示例中使用的实用程序函数。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
```

```
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 TerminateInstances。

场景

创建具有私有子网和 NAT 网关的 VPC

以下代码示例展示了如何：

- 使用 CLI 创建具有私有子网和 NAT 网关的 VPC。
- 设置必要的组件，包括 VPC、子网、路由表和 NAT 网关。
- 配置安全组和 IAM 角色以实现适当的访问权限和安全性。
- 使用 CLI 命令自动创建和配置这些资源。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# VPC with Private Subnets and NAT Gateways (IMDSv2 Compliant Version)
```

```
# This script creates a VPC with public and private subnets in two Availability
 Zones,
# NAT gateways, an internet gateway, route tables, a VPC endpoint for S3,
# security groups, a launch template, an Auto Scaling group, and an Application Load
 Balancer.

# Set up logging
LOG_FILE="vpc-private-subnets-nat.log"
exec > >(tee -a "$LOG_FILE") 2>&1

# Cleanup function to delete all created resources
cleanup_resources() {
  echo "Cleaning up resources..."

  # Delete Auto Scaling group if it exists
  if [ -n "${ASG_NAME:-}" ]; then
    echo "Deleting Auto Scaling group: $ASG_NAME"
    aws autoscaling delete-auto-scaling-group --auto-scaling-group-name "$ASG_NAME"
 --force-delete
    echo "Waiting for Auto Scaling group to be deleted..."
    aws autoscaling wait auto-scaling-groups-deleted --auto-scaling-group-names
 "$ASG_NAME"
  fi

  # Delete load balancer if it exists
  if [ -n "${LB_ARN:-}" ]; then
    echo "Deleting load balancer: $LB_ARN"
    aws elbv2 delete-load-balancer --load-balancer-arn "$LB_ARN"
    # Wait for load balancer to be deleted
    sleep 30
  fi

  # Delete target group if it exists
  if [ -n "${TARGET_GROUP_ARN:-}" ]; then
    echo "Deleting target group: $TARGET_GROUP_ARN"
    aws elbv2 delete-target-group --target-group-arn "$TARGET_GROUP_ARN"
  fi

  # Delete launch template if it exists
  if [ -n "${LAUNCH_TEMPLATE_NAME:-}" ]; then
    echo "Deleting launch template: $LAUNCH_TEMPLATE_NAME"
    aws ec2 delete-launch-template --launch-template-name "$LAUNCH_TEMPLATE_NAME"
  fi
```

```
# Delete NAT Gateways if they exist
if [ -n "${NAT_GW1_ID:-}" ]; then
  echo "Deleting NAT Gateway 1: $NAT_GW1_ID"
  aws ec2 delete-nat-gateway --nat-gateway-id "$NAT_GW1_ID"
fi

if [ -n "${NAT_GW2_ID:-}" ]; then
  echo "Deleting NAT Gateway 2: $NAT_GW2_ID"
  aws ec2 delete-nat-gateway --nat-gateway-id "$NAT_GW2_ID"
fi

# Wait for NAT Gateways to be deleted
if [ -n "${NAT_GW1_ID:-}" ] || [ -n "${NAT_GW2_ID:-}" ]; then
  echo "Waiting for NAT Gateways to be deleted..."
  sleep 60
fi

# Release Elastic IPs if they exist
if [ -n "${EIP1_ALLOC_ID:-}" ]; then
  echo "Releasing Elastic IP 1: $EIP1_ALLOC_ID"
  aws ec2 release-address --allocation-id "$EIP1_ALLOC_ID"
fi

if [ -n "${EIP2_ALLOC_ID:-}" ]; then
  echo "Releasing Elastic IP 2: $EIP2_ALLOC_ID"
  aws ec2 release-address --allocation-id "$EIP2_ALLOC_ID"
fi

# Delete VPC endpoint if it exists
if [ -n "${VPC_ENDPOINT_ID:-}" ]; then
  echo "Deleting VPC endpoint: $VPC_ENDPOINT_ID"
  aws ec2 delete-vpc-endpoints --vpc-endpoint-ids "$VPC_ENDPOINT_ID"
fi

# Delete security groups if they exist
if [ -n "${APP_SG_ID:-}" ]; then
  echo "Deleting application security group: $APP_SG_ID"
  aws ec2 delete-security-group --group-id "$APP_SG_ID"
fi

if [ -n "${LB_SG_ID:-}" ]; then
  echo "Deleting load balancer security group: $LB_SG_ID"
  aws ec2 delete-security-group --group-id "$LB_SG_ID"
fi
```

```
 # Detach and delete Internet Gateway if it exists
 if [ -n "${IGW_ID:-}" ] && [ -n "${VPC_ID:-}" ]; then
   echo "Detaching Internet Gateway: $IGW_ID from VPC: $VPC_ID"
   aws ec2 detach-internet-gateway --internet-gateway-id "$IGW_ID" --vpc-id
"$VPC_ID"
   echo "Deleting Internet Gateway: $IGW_ID"
   aws ec2 delete-internet-gateway --internet-gateway-id "$IGW_ID"
 fi

 # Delete route table associations and route tables if they exist
 if [ -n "${PUBLIC_RT_ASSOC1_ID:-}" ]; then
   echo "Disassociating public route table from subnet 1: $PUBLIC_RT_ASSOC1_ID"
   aws ec2 disassociate-route-table --association-id "$PUBLIC_RT_ASSOC1_ID"
 fi

 if [ -n "${PUBLIC_RT_ASSOC2_ID:-}" ]; then
   echo "Disassociating public route table from subnet 2: $PUBLIC_RT_ASSOC2_ID"
   aws ec2 disassociate-route-table --association-id "$PUBLIC_RT_ASSOC2_ID"
 fi

 if [ -n "${PRIVATE_RT1_ASSOC_ID:-}" ]; then
   echo "Disassociating private route table 1: $PRIVATE_RT1_ASSOC_ID"
   aws ec2 disassociate-route-table --association-id "$PRIVATE_RT1_ASSOC_ID"
 fi

 if [ -n "${PRIVATE_RT2_ASSOC_ID:-}" ]; then
   echo "Disassociating private route table 2: $PRIVATE_RT2_ASSOC_ID"
   aws ec2 disassociate-route-table --association-id "$PRIVATE_RT2_ASSOC_ID"
 fi

 if [ -n "${PUBLIC_RT_ID:-}" ]; then
   echo "Deleting public route table: $PUBLIC_RT_ID"
   aws ec2 delete-route-table --route-table-id "$PUBLIC_RT_ID"
 fi

 if [ -n "${PRIVATE_RT1_ID:-}" ]; then
   echo "Deleting private route table 1: $PRIVATE_RT1_ID"
   aws ec2 delete-route-table --route-table-id "$PRIVATE_RT1_ID"
 fi

 if [ -n "${PRIVATE_RT2_ID:-}" ]; then
   echo "Deleting private route table 2: $PRIVATE_RT2_ID"
   aws ec2 delete-route-table --route-table-id "$PRIVATE_RT2_ID"
```

```
  fi

  # Delete subnets if they exist
  if [ -n "${PUBLIC_SUBNET1_ID:-}" ]; then
    echo "Deleting public subnet 1: $PUBLIC_SUBNET1_ID"
    aws ec2 delete-subnet --subnet-id "$PUBLIC_SUBNET1_ID"
  fi

  if [ -n "${PUBLIC_SUBNET2_ID:-}" ]; then
    echo "Deleting public subnet 2: $PUBLIC_SUBNET2_ID"
    aws ec2 delete-subnet --subnet-id "$PUBLIC_SUBNET2_ID"
  fi

  if [ -n "${PRIVATE_SUBNET1_ID:-}" ]; then
    echo "Deleting private subnet 1: $PRIVATE_SUBNET1_ID"
    aws ec2 delete-subnet --subnet-id "$PRIVATE_SUBNET1_ID"
  fi

  if [ -n "${PRIVATE_SUBNET2_ID:-}" ]; then
    echo "Deleting private subnet 2: $PRIVATE_SUBNET2_ID"
    aws ec2 delete-subnet --subnet-id "$PRIVATE_SUBNET2_ID"
  fi

  # Delete VPC if it exists
  if [ -n "${VPC_ID:-}" ]; then
    echo "Deleting VPC: $VPC_ID"
    aws ec2 delete-vpc --vpc-id "$VPC_ID"
  fi

  echo "Cleanup completed."
}

# Error handling function
handle_error() {
  echo "ERROR: $1"
  echo "Attempting to clean up resources..."
  cleanup_resources
  exit 1
}

# Function to check command success
check_command() {
  if [ $? -ne 0 ]; then
    handle_error "$1"
```

```
  fi
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Create VPC
echo "Creating VPC..."
VPC_RESULT=$(aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications
 "ResourceType=vpc,Tags=[{Key=Name,Value=ProductionVPC-$RANDOM_ID}]")
check_command "Failed to create VPC"

VPC_ID=$(echo "$VPC_RESULT" | jq -r '.Vpc.VpcId')
echo "VPC created with ID: $VPC_ID"

# Get Availability Zones
echo "Getting Availability Zones..."
AZ_RESULT=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0:2].ZoneName' --output text)
check_command "Failed to get Availability Zones"

# Convert space-separated output to array
read -r -a AZS <<< "$AZ_RESULT"
AZ1=${AZS[0]}
AZ2=${AZS[1]}
echo "Using Availability Zones: $AZ1 and $AZ2"

# Create subnets
echo "Creating subnets..."
PUBLIC_SUBNET1_RESULT=$(aws ec2 create-subnet --vpc-id "$VPC_ID" --
cidr-block 10.0.0.0/24 --availability-zone "$AZ1" --tag-specifications
 "ResourceType=subnet,Tags=[{Key=Name,Value=PublicSubnet1-$RANDOM_ID}]")
check_command "Failed to create public subnet 1"
PUBLIC_SUBNET1_ID=$(echo "$PUBLIC_SUBNET1_RESULT" | jq -r '.Subnet.SubnetId')

PRIVATE_SUBNET1_RESULT=$(aws ec2 create-subnet --vpc-id "$VPC_ID" --
cidr-block 10.0.1.0/24 --availability-zone "$AZ1" --tag-specifications
 "ResourceType=subnet,Tags=[{Key=Name,Value=PrivateSubnet1-$RANDOM_ID}]")
check_command "Failed to create private subnet 1"
PRIVATE_SUBNET1_ID=$(echo "$PRIVATE_SUBNET1_RESULT" | jq -r '.Subnet.SubnetId')
```

```
PUBLIC_SUBNET2_RESULT=$(aws ec2 create-subnet --vpc-id "$VPC_ID" --
cidr-block 10.0.2.0/24 --availability-zone "$AZ2" --tag-specifications
 "ResourceType=subnet,Tags=[{Key=Name,Value=PublicSubnet2-$RANDOM_ID}]")
check_command "Failed to create public subnet 2"
PUBLIC_SUBNET2_ID=$(echo "$PUBLIC_SUBNET2_RESULT" | jq -r '.Subnet.SubnetId')

PRIVATE_SUBNET2_RESULT=$(aws ec2 create-subnet --vpc-id "$VPC_ID" --
cidr-block 10.0.3.0/24 --availability-zone "$AZ2" --tag-specifications
 "ResourceType=subnet,Tags=[{Key=Name,Value=PrivateSubnet2-$RANDOM_ID}]")
check_command "Failed to create private subnet 2"
PRIVATE_SUBNET2_ID=$(echo "$PRIVATE_SUBNET2_RESULT" | jq -r '.Subnet.SubnetId')

echo "Subnets created with IDs:"
echo "Public Subnet 1: $PUBLIC_SUBNET1_ID"
echo "Private Subnet 1: $PRIVATE_SUBNET1_ID"
echo "Public Subnet 2: $PUBLIC_SUBNET2_ID"
echo "Private Subnet 2: $PRIVATE_SUBNET2_ID"

# Create Internet Gateway
echo "Creating Internet Gateway..."
IGW_RESULT=$(aws ec2 create-internet-gateway --tag-specifications
 "ResourceType=internet-gateway,Tags=[{Key=Name,Value=ProductionIGW-$RANDOM_ID}]")
check_command "Failed to create Internet Gateway"
IGW_ID=$(echo "$IGW_RESULT" | jq -r '.InternetGateway.InternetGatewayId')
echo "Internet Gateway created with ID: $IGW_ID"

# Attach Internet Gateway to VPC
echo "Attaching Internet Gateway to VPC..."
aws ec2 attach-internet-gateway --internet-gateway-id "$IGW_ID" --vpc-id "$VPC_ID"
check_command "Failed to attach Internet Gateway to VPC"

# Create route tables
echo "Creating route tables..."
PUBLIC_RT_RESULT=$(aws ec2 create-route-table --vpc-id "$VPC_ID" --tag-
specifications "ResourceType=route-table,Tags=[{Key=Name,Value=PublicRouteTable-
$RANDOM_ID}]")
check_command "Failed to create public route table"
PUBLIC_RT_ID=$(echo "$PUBLIC_RT_RESULT" | jq -r '.RouteTable.RouteTableId')

PRIVATE_RT1_RESULT=$(aws ec2 create-route-table --vpc-id "$VPC_ID" --tag-
specifications "ResourceType=route-table,Tags=[{Key=Name,Value=PrivateRouteTable1-
$RANDOM_ID}]")
check_command "Failed to create private route table 1"
PRIVATE_RT1_ID=$(echo "$PRIVATE_RT1_RESULT" | jq -r '.RouteTable.RouteTableId')
```

```
PRIVATE_RT2_RESULT=$(aws ec2 create-route-table --vpc-id "$VPC_ID" --tag-
specifications "ResourceType=route-table,Tags=[{Key=Name,Value=PrivateRouteTable2-
$RANDOM_ID}]")
check_command "Failed to create private route table 2"
PRIVATE_RT2_ID=$(echo "$PRIVATE_RT2_RESULT" | jq -r '.RouteTable.RouteTableId')

echo "Route tables created with IDs:"
echo "Public Route Table: $PUBLIC_RT_ID"
echo "Private Route Table 1: $PRIVATE_RT1_ID"
echo "Private Route Table 2: $PRIVATE_RT2_ID"

# Add route to Internet Gateway in public route table
echo "Adding route to Internet Gateway in public route table..."
aws ec2 create-route --route-table-id "$PUBLIC_RT_ID" --destination-cidr-block
 0.0.0.0/0 --gateway-id "$IGW_ID"
check_command "Failed to add route to Internet Gateway"

# Associate subnets with route tables
echo "Associating subnets with route tables..."
PUBLIC_RT_ASSOC1_RESULT=$(aws ec2 associate-route-table --route-table-id
 "$PUBLIC_RT_ID" --subnet-id "$PUBLIC_SUBNET1_ID")
check_command "Failed to associate public subnet 1 with route table"
PUBLIC_RT_ASSOC1_ID=$(echo "$PUBLIC_RT_ASSOC1_RESULT" | jq -r '.AssociationId')

PUBLIC_RT_ASSOC2_RESULT=$(aws ec2 associate-route-table --route-table-id
 "$PUBLIC_RT_ID" --subnet-id "$PUBLIC_SUBNET2_ID")
check_command "Failed to associate public subnet 2 with route table"
PUBLIC_RT_ASSOC2_ID=$(echo "$PUBLIC_RT_ASSOC2_RESULT" | jq -r '.AssociationId')

PRIVATE_RT1_ASSOC_RESULT=$(aws ec2 associate-route-table --route-table-id
 "$PRIVATE_RT1_ID" --subnet-id "$PRIVATE_SUBNET1_ID")
check_command "Failed to associate private subnet 1 with route table"
PRIVATE_RT1_ASSOC_ID=$(echo "$PRIVATE_RT1_ASSOC_RESULT" | jq -r '.AssociationId')

PRIVATE_RT2_ASSOC_RESULT=$(aws ec2 associate-route-table --route-table-id
 "$PRIVATE_RT2_ID" --subnet-id "$PRIVATE_SUBNET2_ID")
check_command "Failed to associate private subnet 2 with route table"
PRIVATE_RT2_ASSOC_ID=$(echo "$PRIVATE_RT2_ASSOC_RESULT" | jq -r '.AssociationId')

echo "Route table associations created with IDs:"
echo "Public Subnet 1 Association: $PUBLIC_RT_ASSOC1_ID"
echo "Public Subnet 2 Association: $PUBLIC_RT_ASSOC2_ID"
echo "Private Subnet 1 Association: $PRIVATE_RT1_ASSOC_ID"
```

```
echo "Private Subnet 2 Association: $PRIVATE_RT2_ASSOC_ID"

# Create NAT Gateways
echo "Creating NAT Gateways..."

# Allocate Elastic IPs for NAT Gateways
echo "Allocating Elastic IPs for NAT Gateways..."
EIP1_RESULT=$(aws ec2 allocate-address --domain vpc --tag-specifications
 "ResourceType=elastic-ip,Tags=[{Key=Name,Value=NAT1-EIP-$RANDOM_ID}]")
check_command "Failed to allocate Elastic IP 1"
EIP1_ALLOC_ID=$(echo "$EIP1_RESULT" | jq -r '.AllocationId')

EIP2_RESULT=$(aws ec2 allocate-address --domain vpc --tag-specifications
 "ResourceType=elastic-ip,Tags=[{Key=Name,Value=NAT2-EIP-$RANDOM_ID}]")
check_command "Failed to allocate Elastic IP 2"
EIP2_ALLOC_ID=$(echo "$EIP2_RESULT" | jq -r '.AllocationId')

echo "Elastic IPs allocated with IDs:"
echo "EIP 1 Allocation ID: $EIP1_ALLOC_ID"
echo "EIP 2 Allocation ID: $EIP2_ALLOC_ID"

# Create NAT Gateways
echo "Creating NAT Gateway in public subnet 1..."
NAT_GW1_RESULT=$(aws ec2 create-nat-gateway --subnet-id
 "$PUBLIC_SUBNET1_ID" --allocation-id "$EIP1_ALLOC_ID" --tag-specifications
 "ResourceType=natgateway,Tags=[{Key=Name,Value=NAT-Gateway1-$RANDOM_ID}]")
check_command "Failed to create NAT Gateway 1"
NAT_GW1_ID=$(echo "$NAT_GW1_RESULT" | jq -r '.NatGateway.NatGatewayId')

echo "Creating NAT Gateway in public subnet 2..."
NAT_GW2_RESULT=$(aws ec2 create-nat-gateway --subnet-id
 "$PUBLIC_SUBNET2_ID" --allocation-id "$EIP2_ALLOC_ID" --tag-specifications
 "ResourceType=natgateway,Tags=[{Key=Name,Value=NAT-Gateway2-$RANDOM_ID}]")
check_command "Failed to create NAT Gateway 2"
NAT_GW2_ID=$(echo "$NAT_GW2_RESULT" | jq -r '.NatGateway.NatGatewayId')

echo "NAT Gateways created with IDs:"
echo "NAT Gateway 1: $NAT_GW1_ID"
echo "NAT Gateway 2: $NAT_GW2_ID"

# Wait for NAT Gateways to be available
echo "Waiting for NAT Gateways to be available..."
aws ec2 wait nat-gateway-available --nat-gateway-ids "$NAT_GW1_ID"
check_command "NAT Gateway 1 did not become available"
```

```
aws ec2 wait nat-gateway-available --nat-gateway-ids "$NAT_GW2_ID"
check_command "NAT Gateway 2 did not become available"
echo "NAT Gateways are now available"

# Add routes to NAT Gateways in private route tables
echo "Adding routes to NAT Gateways in private route tables..."
aws ec2 create-route --route-table-id "$PRIVATE_RT1_ID" --destination-cidr-block
 0.0.0.0/0 --nat-gateway-id "$NAT_GW1_ID"
check_command "Failed to add route to NAT Gateway 1"

aws ec2 create-route --route-table-id "$PRIVATE_RT2_ID" --destination-cidr-block
 0.0.0.0/0 --nat-gateway-id "$NAT_GW2_ID"
check_command "Failed to add route to NAT Gateway 2"

# Create VPC Endpoint for S3
echo "Creating VPC Endpoint for S3..."
S3_PREFIX_LIST_ID=$(aws ec2 describe-prefix-lists --filters "Name=prefix-
list-name,Values=com.amazonaws.$(aws configure get region).s3" --query
 'PrefixLists[0].PrefixListId' --output text)
check_command "Failed to get S3 prefix list ID"

VPC_ENDPOINT_RESULT=$(aws ec2 create-vpc-endpoint --vpc-id "$VPC_ID" --
service-name "com.amazonaws.$(aws configure get region).s3" --route-table-ids
 "$PRIVATE_RT1_ID" "$PRIVATE_RT2_ID" --tag-specifications "ResourceType=vpc-
endpoint,Tags=[{Key=Name,Value=S3-Endpoint-$RANDOM_ID}]")
check_command "Failed to create VPC endpoint for S3"
VPC_ENDPOINT_ID=$(echo "$VPC_ENDPOINT_RESULT" | jq -r '.VpcEndpoint.VpcEndpointId')
echo "VPC Endpoint created with ID: $VPC_ENDPOINT_ID"

# Create security groups
echo "Creating security groups..."
LB_SG_RESULT=$(aws ec2 create-security-group --group-name "LoadBalancerSG-
$RANDOM_ID" --description "Security group for the load balancer"
 --vpc-id "$VPC_ID" --tag-specifications "ResourceType=security-
group,Tags=[{Key=Name,Value=LoadBalancerSG-$RANDOM_ID}]")
check_command "Failed to create load balancer security group"
LB_SG_ID=$(echo "$LB_SG_RESULT" | jq -r '.GroupId')

# Allow inbound HTTP traffic from anywhere to the load balancer
aws ec2 authorize-security-group-ingress --group-id "$LB_SG_ID" --protocol tcp --
port 80 --cidr 0.0.0.0/0
check_command "Failed to authorize ingress to load balancer security group"
```

```
APP_SG_RESULT=$(aws ec2 create-security-group --group-name "AppServerSG-$RANDOM_ID"
 --description "Security group for the application servers" --vpc-id "$VPC_ID" --
tag-specifications "ResourceType=security-group,Tags=[{Key=Name,Value=AppServerSG-
$RANDOM_ID}]")
check_command "Failed to create application server security group"
APP_SG_ID=$(echo "$APP_SG_RESULT" | jq -r '.GroupId')

# Allow inbound HTTP traffic from the load balancer security group to the
 application servers
aws ec2 authorize-security-group-ingress --group-id "$APP_SG_ID" --protocol tcp --
port 80 --source-group "$LB_SG_ID"
check_command "Failed to authorize ingress to application server security group"

echo "Security groups created with IDs:"
echo "Load Balancer Security Group: $LB_SG_ID"
echo "Application Server Security Group: $APP_SG_ID"

# Create a launch template
echo "Creating launch template..."

# Create user data script with IMDSv2 support
cat > user-data.sh << 'EOF'
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd

# Use IMDSv2 with session token
TOKEN=$(curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600")
AZ=$(curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/latest/
meta-data/placement/availability-zone)
HOSTNAME=$(hostname -f)

echo "<h1>Hello from $HOSTNAME in $AZ</h1>" > /var/www/html/index.html
EOF

# Encode user data
USER_DATA=$(base64 -w 0 user-data.sh)

# Get latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI..."
```

```
AMI_ID=$(aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-
ami-hvm-*-x86_64-gp2" "Name=state,Values=available" --query 'sort_by(Images,
 &CreationDate)[-1].ImageId' --output text)
check_command "Failed to get latest Amazon Linux 2 AMI"
echo "Using AMI: $AMI_ID"

# Create launch template with IMDSv2 required
LAUNCH_TEMPLATE_NAME="AppServerTemplate-$RANDOM_ID"
echo "Creating launch template: $LAUNCH_TEMPLATE_NAME"

aws ec2 create-launch-template \
  --launch-template-name "$LAUNCH_TEMPLATE_NAME" \
  --version-description "Initial version" \
  --tag-specifications "ResourceType=launch-template,Tags=[{Key=Name,Value=
$LAUNCH_TEMPLATE_NAME}]" \
  --launch-template-data "{
    \"NetworkInterfaces\": [{
      \"DeviceIndex\": 0,
      \"Groups\": [\"$APP_SG_ID\"],
      \"DeleteOnTermination\": true
    }],
    \"ImageId\": \"$AMI_ID\",
    \"InstanceType\": \"t3.micro\",
    \"UserData\": \"$USER_DATA\",
    \"MetadataOptions\": {
      \"HttpTokens\": \"required\",
      \"HttpEndpoint\": \"enabled\"
    },
    \"TagSpecifications\": [{
      \"ResourceType\": \"instance\",
      \"Tags\": [{
        \"Key\": \"Name\",
        \"Value\": \"AppServer-$RANDOM_ID\"
      }]
    }]
  }"
check_command "Failed to create launch template"

# Create target group
echo "Creating target group..."
TARGET_GROUP_NAME="AppTargetGroup-$RANDOM_ID"
TARGET_GROUP_RESULT=$(aws elbv2 create-target-group \
  --name "$TARGET_GROUP_NAME" \
  --protocol HTTP \
```

```
  --port 80 \
  --vpc-id "$VPC_ID" \
  --target-type instance \
  --health-check-protocol HTTP \
  --health-check-path "/" \
  --health-check-port traffic-port)
check_command "Failed to create target group"
TARGET_GROUP_ARN=$(echo "$TARGET_GROUP_RESULT" | jq -r
 '.TargetGroups[0].TargetGroupArn')
echo "Target group created with ARN: $TARGET_GROUP_ARN"

# Create load balancer
echo "Creating load balancer..."
LB_NAME="AppLoadBalancer-$RANDOM_ID"
LB_RESULT=$(aws elbv2 create-load-balancer \
  --name "$LB_NAME" \
  --subnets "$PUBLIC_SUBNET1_ID" "$PUBLIC_SUBNET2_ID" \
  --security-groups "$LB_SG_ID" \
  --tags "Key=Name,Value=$LB_NAME")
check_command "Failed to create load balancer"
LB_ARN=$(echo "$LB_RESULT" | jq -r '.LoadBalancers[0].LoadBalancerArn')
echo "Load balancer created with ARN: $LB_ARN"

# Wait for load balancer to be active
echo "Waiting for load balancer to be active..."
aws elbv2 wait load-balancer-available --load-balancer-arns "$LB_ARN"
check_command "Load balancer did not become available"

# Create listener
echo "Creating listener..."
LISTENER_RESULT=$(aws elbv2 create-listener \
  --load-balancer-arn "$LB_ARN" \
  --protocol HTTP \
  --port 80 \
  --default-actions "Type=forward,TargetGroupArn=$TARGET_GROUP_ARN")
check_command "Failed to create listener"
LISTENER_ARN=$(echo "$LISTENER_RESULT" | jq -r '.Listeners[0].ListenerArn')
echo "Listener created with ARN: $LISTENER_ARN"

# Create Auto Scaling group
echo "Creating Auto Scaling group..."
ASG_NAME="AppAutoScalingGroup-$RANDOM_ID"
aws autoscaling create-auto-scaling-group \
  --auto-scaling-group-name "$ASG_NAME" \
```

```
  --launch-template "LaunchTemplateName=$LAUNCH_TEMPLATE_NAME,Version=\$Latest" \
  --min-size 2 \
  --max-size 4 \
  --desired-capacity 2 \
  --vpc-zone-identifier "$PRIVATE_SUBNET1_ID,$PRIVATE_SUBNET2_ID" \
  --target-group-arns "$TARGET_GROUP_ARN" \
  --health-check-type ELB \
  --health-check-grace-period 300 \
  --tags "Key=Name,Value=AppServer-$RANDOM_ID,PropagateAtLaunch=true"
check_command "Failed to create Auto Scaling group"
echo "Auto Scaling group created with name: $ASG_NAME"

# Get load balancer DNS name
LB_DNS_NAME=$(aws elbv2 describe-load-balancers --load-balancer-arns "$LB_ARN" --
query 'LoadBalancers[0].DNSName' --output text)
check_command "Failed to get load balancer DNS name"

echo ""
echo "========================================="
echo "DEPLOYMENT COMPLETE"
echo "========================================="
echo "VPC ID: $VPC_ID"
echo "Public Subnet 1: $PUBLIC_SUBNET1_ID (AZ: $AZ1)"
echo "Private Subnet 1: $PRIVATE_SUBNET1_ID (AZ: $AZ1)"
echo "Public Subnet 2: $PUBLIC_SUBNET2_ID (AZ: $AZ2)"
echo "Private Subnet 2: $PRIVATE_SUBNET2_ID (AZ: $AZ2)"
echo "NAT Gateway 1: $NAT_GW1_ID"
echo "NAT Gateway 2: $NAT_GW2_ID"
echo "Load Balancer: $LB_NAME"
echo "Auto Scaling Group: $ASG_NAME"
echo ""
echo "Your application will be available at: http://$LB_DNS_NAME"
echo "It may take a few minutes for the instances to launch and pass health checks."
echo ""

# Add health check monitoring
echo "========================================="
echo "MONITORING INSTANCE HEALTH AND LOAD BALANCER"
echo "========================================="
echo "Waiting for instances to launch and pass health checks..."
echo "This may take 3-5 minutes. Checking every 30 seconds..."

# Monitor instance health and load balancer accessibility
MAX_ATTEMPTS=10
```

```
ATTEMPT=1
HEALTHY_INSTANCES=0

while [ $ATTEMPT -le $MAX_ATTEMPTS ] && [ $HEALTHY_INSTANCES -lt 2 ]; do
  echo "Check attempt $ATTEMPT of $MAX_ATTEMPTS..."

  # Check Auto Scaling group instances
  echo "Checking Auto Scaling group instances..."
  ASG_INSTANCES=$(aws autoscaling describe-auto-scaling-groups --auto-
scaling-group-names "$ASG_NAME" --query 'AutoScalingGroups[0].Instances[*].
[InstanceId,HealthStatus]' --output json)
  echo "ASG Instances status:"
  echo "$ASG_INSTANCES" | jq -r '.[] | "Instance: \(.[0]), Health: \(.[1])"'

  # Check target group health
  echo "Checking target group health..."
  TARGET_HEALTH=$(aws elbv2 describe-target-health --target-group-arn
 "$TARGET_GROUP_ARN" --output json)
  echo "Target health status:"
  echo "$TARGET_HEALTH" | jq -r '.TargetHealthDescriptions[] | "Instance:
 \(.Target.Id), State: \(.TargetHealth.State), Reason: \(.TargetHealth.Reason // "N/
A"), Description: \(.TargetHealth.Description // "N/A")"'

  # Count healthy instances
  HEALTHY_INSTANCES=$(echo "$TARGET_HEALTH" | jq -r '[.TargetHealthDescriptions[] |
 select(.TargetHealth.State=="healthy")] | length')
  echo "Number of healthy instances: $HEALTHY_INSTANCES of 2 expected"

  # Check if we have healthy instances
  if [ $HEALTHY_INSTANCES -ge 2 ]; then
    echo "All instances are healthy!"

    # Test load balancer accessibility
    echo "Testing load balancer accessibility..."
    HTTP_STATUS=$(curl -s -o /dev/null -w "%{http_code}" "http://$LB_DNS_NAME")

    if [ "$HTTP_STATUS" = "200" ]; then
      echo "Load balancer is accessible! HTTP Status: $HTTP_STATUS"
      echo "You can access your application at: http://$LB_DNS_NAME"

      # Try to get the content to verify IMDSv2 is working
      echo "Fetching content to verify IMDSv2 functionality..."
      CONTENT=$(curl -s "http://$LB_DNS_NAME")
      echo "Response from server:"
```

```
      echo "$CONTENT"

      # Check if the content contains the expected pattern
      if [[ "$CONTENT" == *"Hello from"* && "$CONTENT" == *"in"* ]]; then
        echo "IMDSv2 is working correctly! The instance was able to access metadata
 using the token-based approach."
      else
        echo "Warning: Content doesn't match expected pattern. IMDSv2 functionality
 could not be verified."
      fi

      break
    else
      echo "Load balancer returned HTTP status: $HTTP_STATUS"
      echo "Will try again in 30 seconds..."
    fi
  else
    echo "Waiting for instances to become healthy..."
    echo "Will check again in 30 seconds..."
  fi

  ATTEMPT=$((ATTEMPT+1))

  if [ $ATTEMPT -le $MAX_ATTEMPTS ]; then
    sleep 30
  fi
done

if [ $HEALTHY_INSTANCES -lt 2 ]; then
  echo "Warning: Not all instances are healthy after maximum attempts."
  echo "You may need to wait longer or check for configuration issues."
fi

echo "To test your application, run:"
echo "curl http://$LB_DNS_NAME"
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
  cleanup_resources
```

```
  echo "All resources have been deleted."
else
  echo "Resources will not be deleted. You can manually delete them later."
  echo "To delete resources, run this script again and choose to clean up."
fi
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AllocateAddress
  - AssociateRouteTable
  - AttachInternetGateway
  - AuthorizeSecurityGroupIngress
  - CreateInternetGateway
  - CreateLaunchTemplate
  - CreateNatGateway
  - CreateRoute
  - CreateRouteTable
  - CreateSecurityGroup
  - CreateSubnet
  - CreateVpc
  - CreateVpcEndpoint
  - DeleteAutoScalingGroup
  - DeleteInternetGateway
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteNatGateway
  - DeleteRouteTable
  - DeleteSecurityGroup
  - DeleteSubnet
  - DeleteTargetGroup
  - DeleteVpc
  - DeleteVpcEndpoints
  - DescribeAvailabilityZones

- DescribeImages
- DescribePrefixLists
- DetachInternetGateway
- ReleaseAddress

开始使用 Amazon VPC

以下代码示例展示了如何：

- 设置账户
- 创建和配置 VPC
- 配置网络
- 配置安全性
- 部署资源
- 测试和验证
- 清理资源
- 考虑生产影响
- 考虑安全影响

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# VPC Creation Script
# This script creates a VPC with public and private subnets, internet gateway, NAT
 gateway, and security groups

# Set up logging
LOG_FILE="vpc_creation.log"
```

```
exec > >(tee -a "$LOG_FILE") 2>&1

# Function to handle errors
handle_error() {
  echo "ERROR: $1"
  echo "Resources created before error:"
  for resource in "${CREATED_RESOURCES[@]}"
  do
    echo "- $resource"
  done

  echo "Attempting to clean up resources..."
  cleanup_resources
  exit 1
}

# Function to clean up resources
cleanup_resources() {
  echo "Cleaning up resources in reverse order..."

  # Reverse the array to delete in reverse order of creation
  for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--))
  do
    resource="${CREATED_RESOURCES[$i]}"
    resource_type=$(echo "$resource" | cut -d':' -f1)
    resource_id=$(echo "$resource" | cut -d':' -f2)

    case "$resource_type" in
      "INSTANCE")
        echo "Terminating EC2 instance: $resource_id"
        aws ec2 terminate-instances --instance-ids "$resource_id" || echo "Failed to
 terminate instance: $resource_id"
        # Wait for instance to terminate
        echo "Waiting for instance to terminate..."
        aws ec2 wait instance-terminated --instance-ids "$resource_id" || echo
 "Failed to wait for instance termination: $resource_id"
        ;;
      "KEY_PAIR")
        echo "Deleting key pair: $resource_id"
        aws ec2 delete-key-pair --key-name "$resource_id" || echo "Failed to delete
 key pair: $resource_id"
        # Remove the .pem file if it exists
        if [ -f "${resource_id}.pem" ]; then
          rm -f "${resource_id}.pem"
```

```
        fi
        ;;
      "NAT_GATEWAY")
        echo "Deleting NAT Gateway: $resource_id"
        aws ec2 delete-nat-gateway --nat-gateway-id "$resource_id" || echo "Failed
 to delete NAT Gateway: $resource_id"
        # NAT Gateway deletion takes time, wait for it to complete
        echo "Waiting for NAT Gateway to be deleted..."
        aws ec2 wait nat-gateway-deleted --nat-gateway-ids "$resource_id" || echo
 "Failed to wait for NAT Gateway deletion: $resource_id"
        ;;
      "EIP")
        echo "Releasing Elastic IP: $resource_id"
        aws ec2 release-address --allocation-id "$resource_id" || echo "Failed to
 release Elastic IP: $resource_id"
        ;;
      "ROUTE_TABLE_ASSOCIATION")
        echo "Disassociating Route Table: $resource_id"
        aws ec2 disassociate-route-table --association-id "$resource_id" || echo
 "Failed to disassociate Route Table: $resource_id"
        ;;
      "ROUTE_TABLE")
        echo "Deleting Route Table: $resource_id"
        aws ec2 delete-route-table --route-table-id "$resource_id" || echo "Failed
 to delete Route Table: $resource_id"
        ;;
      "INTERNET_GATEWAY")
        echo "Detaching Internet Gateway: $resource_id from VPC: $VPC_ID"
        aws ec2 detach-internet-gateway --internet-gateway-id "$resource_id" --vpc-
id "$VPC_ID" || echo "Failed to detach Internet Gateway: $resource_id"
        echo "Deleting Internet Gateway: $resource_id"
        aws ec2 delete-internet-gateway --internet-gateway-id "$resource_id" || echo
 "Failed to delete Internet Gateway: $resource_id"
        ;;
      "SECURITY_GROUP")
        echo "Deleting Security Group: $resource_id"
        aws ec2 delete-security-group --group-id "$resource_id" || echo "Failed to
 delete Security Group: $resource_id"
        ;;
      "SUBNET")
        echo "Deleting Subnet: $resource_id"
        aws ec2 delete-subnet --subnet-id "$resource_id" || echo "Failed to delete
 Subnet: $resource_id"
        ;;
```

```
      "VPC")
        echo "Deleting VPC: $resource_id"
        aws ec2 delete-vpc --vpc-id "$resource_id" || echo "Failed to delete VPC:
 $resource_id"
        ;;
    esac
  done
}

# Initialize array to track created resources
CREATED_RESOURCES=()

echo "Starting VPC creation script at $(date)"

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
aws configure list || handle_error "AWS CLI is not properly configured"

# Verify identity and permissions
echo "Verifying identity and permissions..."
if ! aws sts get-caller-identity; then
  echo "ERROR: Unable to verify AWS identity. This could be due to:"
  echo "  - Expired credentials"
  echo "  - Missing or invalid AWS credentials"
  echo "  - Insufficient permissions"
  echo ""
  echo "Please run 'aws configure' to update your credentials or check your IAM
 permissions."
  exit 1
fi

# Create VPC
echo "Creating VPC with CIDR block 10.0.0.0/16..."
VPC_ID=$(aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications
 'ResourceType=vpc,Tags=[{Key=Name,Value=MyVPC}]' --query 'Vpc.VpcId' --output text)

if [ -z "$VPC_ID" ]; then
  handle_error "Failed to create VPC"
fi

CREATED_RESOURCES+=("VPC:$VPC_ID")
echo "VPC created with ID: $VPC_ID"

# Enable DNS support and hostnames
```

```
echo "Enabling DNS support and hostnames for VPC..."
aws ec2 modify-vpc-attribute --vpc-id "$VPC_ID" --enable-dns-support || handle_error
 "Failed to enable DNS support"
aws ec2 modify-vpc-attribute --vpc-id "$VPC_ID" --enable-dns-hostnames ||
 handle_error "Failed to enable DNS hostnames"

# Get available Availability Zones
echo "Getting available Availability Zones..."
AZ1=$(aws ec2 describe-availability-zones --query 'AvailabilityZones[0].ZoneName' --
output text)
AZ2=$(aws ec2 describe-availability-zones --query 'AvailabilityZones[1].ZoneName' --
output text)

if [ -z "$AZ1" ] || [ -z "$AZ2" ]; then
  handle_error "Failed to get Availability Zones"
fi

echo "Using Availability Zones: $AZ1 and $AZ2"

# Create public subnets
echo "Creating public subnet in $AZ1..."
PUBLIC_SUBNET_AZ1=$(aws ec2 create-subnet \
  --vpc-id "$VPC_ID" \
  --cidr-block 10.0.0.0/24 \
  --availability-zone "$AZ1" \
  --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=Public-Subnet-
AZ1}]' \
  --query 'Subnet.SubnetId' \
  --output text)

if [ -z "$PUBLIC_SUBNET_AZ1" ]; then
  handle_error "Failed to create public subnet in AZ1"
fi

CREATED_RESOURCES+=("SUBNET:$PUBLIC_SUBNET_AZ1")
echo "Public subnet created in $AZ1 with ID: $PUBLIC_SUBNET_AZ1"

echo "Creating public subnet in $AZ2..."
PUBLIC_SUBNET_AZ2=$(aws ec2 create-subnet \
  --vpc-id "$VPC_ID" \
  --cidr-block 10.0.1.0/24 \
  --availability-zone "$AZ2" \
  --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=Public-Subnet-
AZ2}]' \
```

```
  --query 'Subnet.SubnetId' \
  --output text)

if [ -z "$PUBLIC_SUBNET_AZ2" ]; then
  handle_error "Failed to create public subnet in AZ2"
fi

CREATED_RESOURCES+=("SUBNET:$PUBLIC_SUBNET_AZ2")
echo "Public subnet created in $AZ2 with ID: $PUBLIC_SUBNET_AZ2"

# Create private subnets
echo "Creating private subnet in $AZ1..."
PRIVATE_SUBNET_AZ1=$(aws ec2 create-subnet \
  --vpc-id "$VPC_ID" \
  --cidr-block 10.0.2.0/24 \
  --availability-zone "$AZ1" \
  --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=Private-Subnet-
AZ1}]' \
  --query 'Subnet.SubnetId' \
  --output text)

if [ -z "$PRIVATE_SUBNET_AZ1" ]; then
  handle_error "Failed to create private subnet in AZ1"
fi

CREATED_RESOURCES+=("SUBNET:$PRIVATE_SUBNET_AZ1")
echo "Private subnet created in $AZ1 with ID: $PRIVATE_SUBNET_AZ1"

echo "Creating private subnet in $AZ2..."
PRIVATE_SUBNET_AZ2=$(aws ec2 create-subnet \
  --vpc-id "$VPC_ID" \
  --cidr-block 10.0.3.0/24 \
  --availability-zone "$AZ2" \
  --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=Private-Subnet-
AZ2}]' \
  --query 'Subnet.SubnetId' \
  --output text)

if [ -z "$PRIVATE_SUBNET_AZ2" ]; then
  handle_error "Failed to create private subnet in AZ2"
fi

CREATED_RESOURCES+=("SUBNET:$PRIVATE_SUBNET_AZ2")
echo "Private subnet created in $AZ2 with ID: $PRIVATE_SUBNET_AZ2"
```

```
# Create Internet Gateway
echo "Creating Internet Gateway..."
IGW_ID=$(aws ec2 create-internet-gateway \
  --tag-specifications 'ResourceType=internet-gateway,Tags=[{Key=Name,Value=MyIGW}]'
 \
  --query 'InternetGateway.InternetGatewayId' \
  --output text)

if [ -z "$IGW_ID" ]; then
  handle_error "Failed to create Internet Gateway"
fi

CREATED_RESOURCES+=("INTERNET_GATEWAY:$IGW_ID")
echo "Internet Gateway created with ID: $IGW_ID"

# Attach Internet Gateway to VPC
echo "Attaching Internet Gateway to VPC..."
aws ec2 attach-internet-gateway --internet-gateway-id "$IGW_ID" --vpc-id "$VPC_ID"
 || handle_error "Failed to attach Internet Gateway to VPC"

# Create public route table
echo "Creating public route table..."
PUBLIC_RT=$(aws ec2 create-route-table \
  --vpc-id "$VPC_ID" \
  --tag-specifications 'ResourceType=route-table,Tags=[{Key=Name,Value=Public-RT}]'
 \
  --query 'RouteTable.RouteTableId' \
  --output text)

if [ -z "$PUBLIC_RT" ]; then
  handle_error "Failed to create public route table"
fi

CREATED_RESOURCES+=("ROUTE_TABLE:$PUBLIC_RT")
echo "Public route table created with ID: $PUBLIC_RT"

# Add route to Internet Gateway
echo "Adding route to Internet Gateway in public route table..."
aws ec2 create-route --route-table-id "$PUBLIC_RT" --destination-cidr-block
 0.0.0.0/0 --gateway-id "$IGW_ID" || handle_error "Failed to add route to Internet
 Gateway"

# Associate public subnets with public route table
```

```
echo "Associating public subnet in $AZ1 with public route table..."
PUBLIC_RT_ASSOC_1=$(aws ec2 associate-route-table --route-table-id "$PUBLIC_RT" --
subnet-id "$PUBLIC_SUBNET_AZ1" --query 'AssociationId' --output text)

if [ -z "$PUBLIC_RT_ASSOC_1" ]; then
  handle_error "Failed to associate public subnet in AZ1 with public route table"
fi

CREATED_RESOURCES+=("ROUTE_TABLE_ASSOCIATION:$PUBLIC_RT_ASSOC_1")

echo "Associating public subnet in $AZ2 with public route table..."
PUBLIC_RT_ASSOC_2=$(aws ec2 associate-route-table --route-table-id "$PUBLIC_RT" --
subnet-id "$PUBLIC_SUBNET_AZ2" --query 'AssociationId' --output text)

if [ -z "$PUBLIC_RT_ASSOC_2" ]; then
  handle_error "Failed to associate public subnet in AZ2 with public route table"
fi

CREATED_RESOURCES+=("ROUTE_TABLE_ASSOCIATION:$PUBLIC_RT_ASSOC_2")

# Create private route table
echo "Creating private route table..."
PRIVATE_RT=$(aws ec2 create-route-table \
  --vpc-id "$VPC_ID" \
  --tag-specifications 'ResourceType=route-table,Tags=[{Key=Name,Value=Private-RT}]'
 \
  --query 'RouteTable.RouteTableId' \
  --output text)

if [ -z "$PRIVATE_RT" ]; then
  handle_error "Failed to create private route table"
fi

CREATED_RESOURCES+=("ROUTE_TABLE:$PRIVATE_RT")
echo "Private route table created with ID: $PRIVATE_RT"

# Associate private subnets with private route table
echo "Associating private subnet in $AZ1 with private route table..."
PRIVATE_RT_ASSOC_1=$(aws ec2 associate-route-table --route-table-id "$PRIVATE_RT" --
subnet-id "$PRIVATE_SUBNET_AZ1" --query 'AssociationId' --output text)

if [ -z "$PRIVATE_RT_ASSOC_1" ]; then
  handle_error "Failed to associate private subnet in AZ1 with private route table"
fi
```

```
CREATED_RESOURCES+=("ROUTE_TABLE_ASSOCIATION:$PRIVATE_RT_ASSOC_1")

echo "Associating private subnet in $AZ2 with private route table..."
PRIVATE_RT_ASSOC_2=$(aws ec2 associate-route-table --route-table-id "$PRIVATE_RT" --
subnet-id "$PRIVATE_SUBNET_AZ2" --query 'AssociationId' --output text)

if [ -z "$PRIVATE_RT_ASSOC_2" ]; then
  handle_error "Failed to associate private subnet in AZ2 with private route table"
fi

CREATED_RESOURCES+=("ROUTE_TABLE_ASSOCIATION:$PRIVATE_RT_ASSOC_2")

# Allocate Elastic IP for NAT Gateway
echo "Allocating Elastic IP for NAT Gateway..."
EIP_ALLOC=$(aws ec2 allocate-address --domain vpc --query 'AllocationId' --output
 text)

if [ -z "$EIP_ALLOC" ]; then
  handle_error "Failed to allocate Elastic IP"
fi

CREATED_RESOURCES+=("EIP:$EIP_ALLOC")
echo "Elastic IP allocated with ID: $EIP_ALLOC"

# Create NAT Gateway
echo "Creating NAT Gateway in public subnet in $AZ1..."
NAT_GW=$(aws ec2 create-nat-gateway \
  --subnet-id "$PUBLIC_SUBNET_AZ1" \
  --allocation-id "$EIP_ALLOC" \
  --tag-specifications
 'ResourceType=natgateway,Tags=[{Key=Name,Value=MyNATGateway}]' \
  --query 'NatGateway.NatGatewayId' \
  --output text)

if [ -z "$NAT_GW" ]; then
  handle_error "Failed to create NAT Gateway"
fi

CREATED_RESOURCES+=("NAT_GATEWAY:$NAT_GW")
echo "NAT Gateway created with ID: $NAT_GW"

# Wait for NAT Gateway to be available
echo "Waiting for NAT Gateway to be available..."
```

```
aws ec2 wait nat-gateway-available --nat-gateway-ids "$NAT_GW" || handle_error "NAT
 Gateway did not become available"

# Add route to NAT Gateway in private route table
echo "Adding route to NAT Gateway in private route table..."
aws ec2 create-route --route-table-id "$PRIVATE_RT" --destination-cidr-block
 0.0.0.0/0 --nat-gateway-id "$NAT_GW" || handle_error "Failed to add route to NAT
 Gateway"

# Enable auto-assign public IP for instances in public subnets
echo "Enabling auto-assign public IP for instances in public subnet in $AZ1..."
aws ec2 modify-subnet-attribute --subnet-id "$PUBLIC_SUBNET_AZ1" --map-public-ip-on-
launch || handle_error "Failed to enable auto-assign public IP for public subnet in
 AZ1"

echo "Enabling auto-assign public IP for instances in public subnet in $AZ2..."
aws ec2 modify-subnet-attribute --subnet-id "$PUBLIC_SUBNET_AZ2" --map-public-ip-on-
launch || handle_error "Failed to enable auto-assign public IP for public subnet in
 AZ2"

# Create security group for web servers
echo "Creating security group for web servers..."
WEB_SG=$(aws ec2 create-security-group \
  --group-name "WebServerSG-$(date +%s)" \
  --description "Security group for web servers" \
  --vpc-id "$VPC_ID" \
  --query 'GroupId' \
  --output text)

if [ -z "$WEB_SG" ]; then
  handle_error "Failed to create security group for web servers"
fi

CREATED_RESOURCES+=("SECURITY_GROUP:$WEB_SG")
echo "Security group for web servers created with ID: $WEB_SG"

# Allow HTTP and HTTPS traffic
echo "Allowing HTTP traffic to web servers security group..."
aws ec2 authorize-security-group-ingress --group-id "$WEB_SG" --protocol tcp --port
 80 --cidr 0.0.0.0/0 || handle_error "Failed to allow HTTP traffic"

echo "Allowing HTTPS traffic to web servers security group..."
aws ec2 authorize-security-group-ingress --group-id "$WEB_SG" --protocol tcp --port
 443 --cidr 0.0.0.0/0 || handle_error "Failed to allow HTTPS traffic"
```

```
# Note: In a production environment, you should restrict the source IP ranges for
 security
echo "NOTE: In a production environment, you should restrict the source IP ranges
 for HTTP and HTTPS traffic"

# Create security group for database servers
echo "Creating security group for database servers..."
DB_SG=$(aws ec2 create-security-group \
  --group-name "DBServerSG-$(date +%s)" \
  --description "Security group for database servers" \
  --vpc-id "$VPC_ID" \
  --query 'GroupId' \
  --output text)

if [ -z "$DB_SG" ]; then
  handle_error "Failed to create security group for database servers"
fi

CREATED_RESOURCES+=("SECURITY_GROUP:$DB_SG")
echo "Security group for database servers created with ID: $DB_SG"

# Allow MySQL/Aurora traffic from web servers only
echo "Allowing MySQL/Aurora traffic from web servers to database servers..."
aws ec2 authorize-security-group-ingress --group-id "$DB_SG" --protocol tcp --
port 3306 --source-group "$WEB_SG" || handle_error "Failed to allow MySQL/Aurora
 traffic"

# Verify VPC configuration
echo "Verifying VPC configuration..."
echo "VPC:"
aws ec2 describe-vpcs --vpc-id "$VPC_ID" || handle_error "Failed to describe VPC"

echo "Subnets:"
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID" || handle_error
 "Failed to describe subnets"

echo "Route tables:"
aws ec2 describe-route-tables --filters "Name=vpc-id,Values=$VPC_ID" || handle_error
 "Failed to describe route tables"

echo "Internet gateway:"
aws ec2 describe-internet-gateways --filters "Name=attachment.vpc-id,Values=$VPC_ID"
 || handle_error "Failed to describe Internet Gateway"
```

```
echo "NAT gateway:"
aws ec2 describe-nat-gateways --filter "Name=vpc-id,Values=$VPC_ID" || handle_error
 "Failed to describe NAT Gateway"

echo "Security groups:"
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$VPC_ID" ||
 handle_error "Failed to describe security groups"

echo ""
# Summary of created resources
echo "VPC creation completed successfully!"
echo "Summary of created resources:"
echo "- VPC: $VPC_ID"
echo "- Public Subnet in $AZ1: $PUBLIC_SUBNET_AZ1"
echo "- Public Subnet in $AZ2: $PUBLIC_SUBNET_AZ2"
echo "- Private Subnet in $AZ1: $PRIVATE_SUBNET_AZ1"
echo "- Private Subnet in $AZ2: $PRIVATE_SUBNET_AZ2"
echo "- Internet Gateway: $IGW_ID"
echo "- Public Route Table: $PUBLIC_RT"
echo "- Private Route Table: $PRIVATE_RT"
echo "- Elastic IP: $EIP_ALLOC"
echo "- NAT Gateway: $NAT_GW"
echo "- Web Servers Security Group: $WEB_SG"
echo "- Database Servers Security Group: $DB_SG"

# Deploy EC2 instances
echo ""
echo "Deploying EC2 instances..."

# Create key pair for SSH access
KEY_NAME="vpc-tutorial-key-$(date +%s)"
echo "Creating key pair $KEY_NAME..."
aws ec2 create-key-pair --key-name "$KEY_NAME" --query 'KeyMaterial' --output text >
 "${KEY_NAME}.pem" || handle_error "Failed to create key pair"
chmod 400 "${KEY_NAME}.pem"
echo "Key pair saved to ${KEY_NAME}.pem"
CREATED_RESOURCES+=("KEY_PAIR:$KEY_NAME")

# Get latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
```

```
  --query "sort_by(Images, &CreationDate)[-1].ImageId" --output text) ||
 handle_error "Failed to get AMI"
echo "Using AMI: $AMI_ID"

# Launch web server in public subnet
echo "Launching web server in public subnet..."
WEB_INSTANCE=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --count 1 \
  --instance-type t2.micro \
  --key-name "$KEY_NAME" \
  --security-group-ids "$WEB_SG" \
  --subnet-id "$PUBLIC_SUBNET_AZ1" \
  --associate-public-ip-address \
  --user-data '#!/bin/bash
    yum update -y
    yum install -y httpd
    systemctl start httpd
    systemctl enable httpd
    echo "<h1>Hello from $(hostname -f) in the public subnet</h1>" > /var/www/html/
index.html' \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=WebServer}]' \
  --query 'Instances[0].InstanceId' \
  --output text) || handle_error "Failed to launch web server"
echo "Web server instance created with ID: $WEB_INSTANCE"
CREATED_RESOURCES+=("INSTANCE:$WEB_INSTANCE")

# Wait for web server to be running
echo "Waiting for web server to be running..."
aws ec2 wait instance-running --instance-ids "$WEB_INSTANCE"

# Get web server public IP
WEB_PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$WEB_INSTANCE" \
  --query 'Reservations[0].Instances[0].PublicIpAddress' --output text)
echo "Web server public IP: $WEB_PUBLIC_IP"
echo "You can access the web server at: http://$WEB_PUBLIC_IP"

# Launch database server in private subnet
echo "Launching database server in private subnet..."
DB_INSTANCE=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --count 1 \
  --instance-type t2.micro \
  --key-name "$KEY_NAME" \
```

```
  --security-group-ids "$DB_SG" \
  --subnet-id "$PRIVATE_SUBNET_AZ1" \
  --user-data '#!/bin/bash
    yum update -y
    yum install -y mariadb-server
    systemctl start mariadb
    systemctl enable mariadb' \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=DBServer}]' \
  --query 'Instances[0].InstanceId' \
  --output text) || handle_error "Failed to launch database server"
echo "Database server instance created with ID: $DB_INSTANCE"
CREATED_RESOURCES+=("INSTANCE:$DB_INSTANCE")

# Wait for database server to be running
echo "Waiting for database server to be running..."
aws ec2 wait instance-running --instance-ids "$DB_INSTANCE"

# Get database server private IP
DB_PRIVATE_IP=$(aws ec2 describe-instances --instance-ids "$DB_INSTANCE" \
  --query 'Reservations[0].Instances[0].PrivateIpAddress' --output text)
echo "Database server private IP: $DB_PRIVATE_IP"

echo "EC2 instances deployed successfully!"
echo "- Web Server (Public): $WEB_INSTANCE ($WEB_PUBLIC_IP)"
echo "- Database Server (Private): $DB_INSTANCE ($DB_PRIVATE_IP)"
echo ""
echo "Note: To connect to the web server: ssh -i ${KEY_NAME}.pem ec2-user@
$WEB_PUBLIC_IP"
echo "To connect to the database server, you must first connect to the web server,
 then use it as a bastion host."
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE
if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
  echo "Cleaning up resources..."
  cleanup_resources
  echo "All resources have been cleaned up."
else
  echo "Resources will not be cleaned up. You can manually clean them up later."
fi

echo "Script completed at $(date)"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AllocateAddress
  - AssociateRouteTable
  - AttachInternetGateway
  - AuthorizeSecurityGroupIngress
  - CreateInternetGateway
  - :CreateKeyPair
  - CreateNatGateway
  - CreateRoute
  - CreateRouteTable
  - CreateSecurityGroup
  - CreateSubnet
  - CreateVpc
  - DeleteInternetGateway
  - :DeleteKeyPair
  - DeleteNatGateway
  - DeleteRouteTable
  - DeleteSecurityGroup
  - DeleteSubnet
  - DeleteVpc
  - DescribeAvailabilityZones
  - DescribeImages
  - DescribeInstances
  - DescribeInternetGateways
  - DescribeNatGateways
  - DescribeRouteTables
  - DescribeSecurityGroups
  - DescribeSubnets
  - DescribeVpcs

- DetachInternetGateway
- DisassociateRouteTable
- ModifySubnetAttribute
- ModifyVpcAttribute
- ReleaseAddress
- RunInstances：
- TerminateInstances

中转网关入门

以下代码示例展示了如何：

- 创建具有 DNS 支持和默认路由表设置的中转网关
- 等待中转网关变为可用
- 使用子网将两个 VPC 连接到中转网关
- 等待 VPC 附件变为可用
- 通过中转网关在 VPC 之间添加路由
- 测试 VPC 间的连接
- 清理资源，包括路由、附件和中转网关

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# Amazon VPC Transit Gateway CLI Script
# This script demonstrates how to create a transit gateway and connect two VPCs
# Modified to work with older AWS CLI versions that don't support transit gateway
 wait commands
```

```
# Error handling
set -e
LOG_FILE="transit-gateway-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

# Function to wait for transit gateway to be available
wait_for_tgw() {
  local tgw_id=$1
  echo "Waiting for Transit Gateway $tgw_id to become available..."

  while true; do
    status=$(aws ec2 describe-transit-gateways --transit-gateway-ids "$tgw_id" --
query "TransitGateways[0].State" --output text)
    echo "Current status: $status"

    if [ "$status" = "available" ]; then
      echo "Transit Gateway is now available"
      break
    fi

    echo "Waiting for transit gateway to become available. Current state: $status"
    sleep 10
  done
}

# Function to wait for transit gateway attachment to be available
wait_for_tgw_attachment() {
  local attachment_id=$1
  echo "Waiting for Transit Gateway Attachment $attachment_id to become
 available..."

  while true; do
    status=$(aws ec2 describe-transit-gateway-vpc-attachments --transit-gateway-
attachment-ids "$attachment_id" --query "TransitGatewayVpcAttachments[0].State" --
output text)
    echo "Current status: $status"

    if [ "$status" = "available" ]; then
      echo "Transit Gateway Attachment is now available"
      break
    fi

    echo "Waiting for transit gateway attachment to become available. Current state:
 $status"
```

```
    sleep 10
  done
}

# Function to wait for transit gateway attachment to be deleted
wait_for_tgw_attachment_deleted() {
  local attachment_id=$1
  echo "Waiting for Transit Gateway Attachment $attachment_id to be deleted..."

  while true; do
    # Check if the attachment still exists
    count=$(aws ec2 describe-transit-gateway-vpc-attachments --filters
 "Name=transit-gateway-attachment-id,Values=$attachment_id" --query
 "length(TransitGatewayVpcAttachments)" --output text)

    if [ "$count" = "0" ]; then
      echo "Transit Gateway Attachment has been deleted"
      break
    fi

    status=$(aws ec2 describe-transit-gateway-vpc-attachments --transit-gateway-
attachment-ids "$attachment_id" --query "TransitGatewayVpcAttachments[0].State" --
output text 2>/dev/null || echo "deleted")

    if [ "$status" = "deleted" ]; then
      echo "Transit Gateway Attachment has been deleted"
      break
    fi

    echo "Waiting for transit gateway attachment to be deleted. Current state:
 $status"
    sleep 10
  done
}

# Function to clean up resources
cleanup() {
  echo "Error occurred. Cleaning up resources..."

  # Delete resources in reverse order
  if [ ! -z "$TGW_ATTACHMENT_1_ID" ]; then
    echo "Deleting Transit Gateway VPC Attachment 1: $TGW_ATTACHMENT_1_ID"
    aws ec2 delete-transit-gateway-vpc-attachment --transit-gateway-attachment-id
 "$TGW_ATTACHMENT_1_ID" || true
```

```
    wait_for_tgw_attachment_deleted "$TGW_ATTACHMENT_1_ID" || true
  fi

  if [ ! -z "$TGW_ATTACHMENT_2_ID" ]; then
    echo "Deleting Transit Gateway VPC Attachment 2: $TGW_ATTACHMENT_2_ID"
    aws ec2 delete-transit-gateway-vpc-attachment --transit-gateway-attachment-id
 "$TGW_ATTACHMENT_2_ID" || true
    wait_for_tgw_attachment_deleted "$TGW_ATTACHMENT_2_ID" || true
  fi

  if [ ! -z "$TGW_ID" ]; then
    echo "Deleting Transit Gateway: $TGW_ID"
    aws ec2 delete-transit-gateway --transit-gateway-id "$TGW_ID" || true
  fi

  exit 1
}

# Set up trap for error handling
trap cleanup ERR

echo "=== Amazon VPC Transit Gateway Tutorial ==="
echo "This script will create a transit gateway and connect two VPCs"
echo ""

# Get a valid availability zone dynamically
echo "Getting available AZ in current region..."
AZ=$(aws ec2 describe-availability-zones --query "AvailabilityZones[0].ZoneName" --
output text)
echo "Using availability zone: $AZ"

# Check if VPCs exist
echo "Checking for existing VPCs..."
VPC1_ID=$(aws ec2 describe-vpcs --filters "Name=tag:Name,Values=VPC1" --query
 "Vpcs[0].VpcId" --output text)
VPC2_ID=$(aws ec2 describe-vpcs --filters "Name=tag:Name,Values=VPC2" --query
 "Vpcs[0].VpcId" --output text)

if [ "$VPC1_ID" == "None" ] || [ -z "$VPC1_ID" ]; then
  echo "Creating VPC1..."
  VPC1_ID=$(aws ec2 create-vpc --cidr-block 10.1.0.0/16 --tag-specifications
 'ResourceType=vpc,Tags=[{Key=Name,Value=VPC1}]' --query Vpc.VpcId --output text)
  echo "Created VPC1: $VPC1_ID"
```

```
  # Create a subnet in VPC1
  echo "Creating subnet in VPC1..."
  SUBNET1_ID=$(aws ec2 create-subnet --vpc-id "$VPC1_ID" --cidr-
block 10.1.0.0/24 --availability-zone "$AZ" --tag-specifications
 'ResourceType=subnet,Tags=[{Key=Name,Value=VPC1-Subnet}]' --query Subnet.SubnetId
 --output text)
  echo "Created subnet in VPC1: $SUBNET1_ID"
else
  echo "Using existing VPC1: $VPC1_ID"
  SUBNET1_ID=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC1_ID" --
query "Subnets[0].SubnetId" --output text)
  if [ "$SUBNET1_ID" == "None" ] || [ -z "$SUBNET1_ID" ]; then
    echo "Creating subnet in VPC1..."
    SUBNET1_ID=$(aws ec2 create-subnet --vpc-id "$VPC1_ID" --cidr-
block 10.1.0.0/24 --availability-zone "$AZ" --tag-specifications
 'ResourceType=subnet,Tags=[{Key=Name,Value=VPC1-Subnet}]' --query Subnet.SubnetId
 --output text)
    echo "Created subnet in VPC1: $SUBNET1_ID"
  else
    echo "Using existing subnet in VPC1: $SUBNET1_ID"
  fi
fi

if [ "$VPC2_ID" == "None" ] || [ -z "$VPC2_ID" ]; then
  echo "Creating VPC2..."
  VPC2_ID=$(aws ec2 create-vpc --cidr-block 10.2.0.0/16 --tag-specifications
 'ResourceType=vpc,Tags=[{Key=Name,Value=VPC2}]' --query Vpc.VpcId --output text)
  echo "Created VPC2: $VPC2_ID"

  # Create a subnet in VPC2
  echo "Creating subnet in VPC2..."
  SUBNET2_ID=$(aws ec2 create-subnet --vpc-id "$VPC2_ID" --cidr-
block 10.2.0.0/24 --availability-zone "$AZ" --tag-specifications
 'ResourceType=subnet,Tags=[{Key=Name,Value=VPC2-Subnet}]' --query Subnet.SubnetId
 --output text)
  echo "Created subnet in VPC2: $SUBNET2_ID"
else
  echo "Using existing VPC2: $VPC2_ID"
  SUBNET2_ID=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC2_ID" --
query "Subnets[0].SubnetId" --output text)
  if [ "$SUBNET2_ID" == "None" ] || [ -z "$SUBNET2_ID" ]; then
    echo "Creating subnet in VPC2..."
    SUBNET2_ID=$(aws ec2 create-subnet --vpc-id "$VPC2_ID" --cidr-
block 10.2.0.0/24 --availability-zone "$AZ" --tag-specifications
```

```
 'ResourceType=subnet,Tags=[{Key=Name,Value=VPC2-Subnet}]' --query Subnet.SubnetId
 --output text)
    echo "Created subnet in VPC2: $SUBNET2_ID"
  else
    echo "Using existing subnet in VPC2: $SUBNET2_ID"
  fi
fi

# Get route tables for each VPC
RTB1_ID=$(aws ec2 describe-route-tables --filters "Name=vpc-id,Values=$VPC1_ID" --
query "RouteTables[0].RouteTableId" --output text)
RTB2_ID=$(aws ec2 describe-route-tables --filters "Name=vpc-id,Values=$VPC2_ID" --
query "RouteTables[0].RouteTableId" --output text)

echo "Route table for VPC1: $RTB1_ID"
echo "Route table for VPC2: $RTB2_ID"

# Step 1: Create the transit gateway
echo "Creating Transit Gateway..."
TGW_ID=$(aws ec2 create-transit-gateway \
  --description "My Transit Gateway" \
  --options
 AmazonSideAsn=64512,AutoAcceptSharedAttachments=disable,DefaultRouteTableAssociation=enable
 \
  --tag-specifications 'ResourceType=transit-
gateway,Tags=[{Key=Name,Value=MyTransitGateway}]' \
  --query TransitGateway.TransitGatewayId \
  --output text)

echo "Created Transit Gateway: $TGW_ID"

# Wait for the transit gateway to become available
wait_for_tgw "$TGW_ID"

# Step 2: Attach VPCs to the transit gateway
echo "Attaching VPC1 to Transit Gateway..."
TGW_ATTACHMENT_1_ID=$(aws ec2 create-transit-gateway-vpc-attachment \
  --transit-gateway-id "$TGW_ID" \
  --vpc-id "$VPC1_ID" \
  --subnet-ids "$SUBNET1_ID" \
  --tag-specifications 'ResourceType=transit-gateway-
attachment,Tags=[{Key=Name,Value=VPC1-Attachment}]' \
  --query TransitGatewayVpcAttachment.TransitGatewayAttachmentId \
  --output text)
```

```
echo "Created Transit Gateway VPC Attachment for VPC1: $TGW_ATTACHMENT_1_ID"

echo "Attaching VPC2 to Transit Gateway..."
TGW_ATTACHMENT_2_ID=$(aws ec2 create-transit-gateway-vpc-attachment \
  --transit-gateway-id "$TGW_ID" \
  --vpc-id "$VPC2_ID" \
  --subnet-ids "$SUBNET2_ID" \
  --tag-specifications 'ResourceType=transit-gateway-
attachment,Tags=[{Key=Name,Value=VPC2-Attachment}]' \
  --query TransitGatewayVpcAttachment.TransitGatewayAttachmentId \
  --output text)

echo "Created Transit Gateway VPC Attachment for VPC2: $TGW_ATTACHMENT_2_ID"

# Wait for the attachments to become available
wait_for_tgw_attachment "$TGW_ATTACHMENT_1_ID"
wait_for_tgw_attachment "$TGW_ATTACHMENT_2_ID"

# Step 3: Add routes between the transit gateway and VPCs
echo "Adding route from VPC1 to VPC2 via Transit Gateway..."
aws ec2 create-route \
  --route-table-id "$RTB1_ID" \
  --destination-cidr-block 10.2.0.0/16 \
  --transit-gateway-id "$TGW_ID"

echo "Adding route from VPC2 to VPC1 via Transit Gateway..."
aws ec2 create-route \
  --route-table-id "$RTB2_ID" \
  --destination-cidr-block 10.1.0.0/16 \
  --transit-gateway-id "$TGW_ID"

echo "Routes added successfully"

# Step 4: Display information for testing
echo ""
echo "=== Transit Gateway Setup Complete ==="
echo "Transit Gateway ID: $TGW_ID"
echo "VPC1 ID: $VPC1_ID"
echo "VPC2 ID: $VPC2_ID"
echo ""
echo "To test connectivity:"
echo "1. Launch an EC2 instance in each VPC"
echo "2. Configure security groups to allow ICMP traffic"
```

```
echo "3. Connect to one instance and ping the other instance's private IP"
echo ""

# Prompt user before cleanup
read -p "Press Enter to view created resources, or Ctrl+C to exit without
 cleanup..."

echo ""
echo "=== Resources Created ==="
echo "Transit Gateway: $TGW_ID"
echo "VPC1: $VPC1_ID"
echo "VPC2: $VPC2_ID"
echo "Subnet in VPC1: $SUBNET1_ID"
echo "Subnet in VPC2: $SUBNET2_ID"
echo "Transit Gateway Attachment for VPC1: $TGW_ATTACHMENT_1_ID"
echo "Transit Gateway Attachment for VPC2: $TGW_ATTACHMENT_2_ID"
echo ""

read -p "Do you want to clean up these resources? (y/n): " CLEANUP_CONFIRM
if [[ $CLEANUP_CONFIRM == "y" || $CLEANUP_CONFIRM == "Y" ]]; then
  echo "Starting cleanup..."

  # Delete routes
  echo "Deleting routes..."
  aws ec2 delete-route --route-table-id "$RTB1_ID" --destination-cidr-block
 10.2.0.0/16
  aws ec2 delete-route --route-table-id "$RTB2_ID" --destination-cidr-block
 10.1.0.0/16

  # Delete transit gateway attachments
  echo "Deleting Transit Gateway VPC Attachment for VPC1: $TGW_ATTACHMENT_1_ID"
  aws ec2 delete-transit-gateway-vpc-attachment --transit-gateway-attachment-id
 "$TGW_ATTACHMENT_1_ID"

  echo "Deleting Transit Gateway VPC Attachment for VPC2: $TGW_ATTACHMENT_2_ID"
  aws ec2 delete-transit-gateway-vpc-attachment --transit-gateway-attachment-id
 "$TGW_ATTACHMENT_2_ID"

  # Wait for attachments to be deleted
  wait_for_tgw_attachment_deleted "$TGW_ATTACHMENT_1_ID"
  wait_for_tgw_attachment_deleted "$TGW_ATTACHMENT_2_ID"

  # Delete transit gateway
  echo "Deleting Transit Gateway: $TGW_ID"
```

```
  aws ec2 delete-transit-gateway --transit-gateway-id "$TGW_ID"

  echo "Cleanup completed successfully"
else
  echo "Skipping cleanup. Resources will continue to incur charges until manually
 deleted."
fi

echo "Tutorial completed. See $LOG_FILE for detailed logs."
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateRoute
  - CreateSubnet
  - CreateTransitGateway
  - CreateTransitGatewayVpcAttachment
  - CreateVpc
  - DeleteRoute
  - DeleteTransitGateway
  - DeleteTransitGatewayVpcAttachment
  - DescribeAvailabilityZones
  - DescribeRouteTables
  - DescribeSubnets
  - DescribeTransitGatewayAttachments
  - DescribeTransitGatewayVpcAttachments
  - DescribeTransitGateways
  - DescribeVpcs

VPC IPAM 入门

以下代码示例展示了如何：

- 使用 CLI 设置和配置 Amazon VPC IP 地址管理器（IPAM）。
- 创建包含运营区域（例如 us-east-1、us-west-2）的 IPAM。
- 检索 IPAM 的私有范围 ID。

- 创建 IPv4 池（顶级池、区域池和开发池）的层次结构。
- 为每个池预置 CIDR 块（例如 10.0.0.0/8、10.0.0.0/16、10.0.0.0/24）。
- 使用从 IPAM 池分配的 CIDR 创建 VPC。
- 验证 IPAM 池分配和 VPC 创建情况。
- 对权限错误、CIDR 分配失败和依赖项冲突等常见问题进行故障排除。
- 清理 IPAM 资源（VPC、池、CIDR 和 IPAM），以避免不必要的费用。
- 探索 IPAM 高级 IPAM 功能的后续步骤。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# IPAM Getting Started CLI Script - Version 7
# This script creates an IPAM, creates a hierarchy of IP address pools, and
 allocates a CIDR to a VPC
# Fixed to correctly identify the private scope ID, wait for resources to be
 available, add locale to development pool,
# use the correct parameter names for VPC creation, and wait for CIDR provisioning
 to complete

# Set up logging
LOG_FILE="ipam_script.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting IPAM setup script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
```

```
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED:"
    echo "========================================="

    if [ -n "$VPC_ID" ]; then
        echo "VPC: $VPC_ID"
    fi

    if [ -n "$DEV_POOL_ID" ]; then
        echo "Development Pool: $DEV_POOL_ID"
    fi

    if [ -n "$REGIONAL_POOL_ID" ]; then
        echo "Regional Pool: $REGIONAL_POOL_ID"
    fi

    if [ -n "$TOP_POOL_ID" ]; then
        echo "Top-level Pool: $TOP_POOL_ID"
    fi

    if [ -n "$IPAM_ID" ]; then
        echo "IPAM: $IPAM_ID"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."

        # Delete resources in reverse order of creation to handle dependencies

        if [ -n "$VPC_ID" ]; then
            echo "Deleting VPC: $VPC_ID"
```

```
            aws ec2 delete-vpc --vpc-id "$VPC_ID" || echo "Failed to delete VPC"
            echo "Waiting for VPC to be deleted..."
            sleep 10
        fi

        if [ -n "$DEV_POOL_ID" ]; then
            echo "Deleting Development Pool: $DEV_POOL_ID"
            # First deprovision any CIDRs from the pool
            CIDRS=$(aws ec2 get-ipam-pool-cidrs --ipam-pool-id "$DEV_POOL_ID" --
query 'IpamPoolCidrs[].Cidr' --output text)
            for CIDR in $CIDRS; do
                echo "Deprovisioning CIDR $CIDR from Development Pool"
                aws ec2 deprovision-ipam-pool-cidr --ipam-pool-id "$DEV_POOL_ID" --
cidr "$CIDR" || echo "Failed to deprovision CIDR $CIDR"
                sleep 5
            done
            aws ec2 delete-ipam-pool --ipam-pool-id "$DEV_POOL_ID" || echo "Failed
 to delete Development Pool"
            echo "Waiting for Development Pool to be deleted..."
            sleep 10
        fi

        if [ -n "$REGIONAL_POOL_ID" ]; then
            echo "Deleting Regional Pool: $REGIONAL_POOL_ID"
            # First deprovision any CIDRs from the pool
            CIDRS=$(aws ec2 get-ipam-pool-cidrs --ipam-pool-id "$REGIONAL_POOL_ID"
 --query 'IpamPoolCidrs[].Cidr' --output text)
            for CIDR in $CIDRS; do
                echo "Deprovisioning CIDR $CIDR from Regional Pool"
                aws ec2 deprovision-ipam-pool-cidr --ipam-pool-id
 "$REGIONAL_POOL_ID" --cidr "$CIDR" || echo "Failed to deprovision CIDR $CIDR"
                sleep 5
            done
            aws ec2 delete-ipam-pool --ipam-pool-id "$REGIONAL_POOL_ID" || echo
 "Failed to delete Regional Pool"
            echo "Waiting for Regional Pool to be deleted..."
            sleep 10
        fi

        if [ -n "$TOP_POOL_ID" ]; then
            echo "Deleting Top-level Pool: $TOP_POOL_ID"
            # First deprovision any CIDRs from the pool
            CIDRS=$(aws ec2 get-ipam-pool-cidrs --ipam-pool-id "$TOP_POOL_ID" --
query 'IpamPoolCidrs[].Cidr' --output text)
```

```
            for CIDR in $CIDRS; do
                echo "Deprovisioning CIDR $CIDR from Top-level Pool"
                aws ec2 deprovision-ipam-pool-cidr --ipam-pool-id "$TOP_POOL_ID" --
cidr "$CIDR" || echo "Failed to deprovision CIDR $CIDR"
                sleep 5
            done
            aws ec2 delete-ipam-pool --ipam-pool-id "$TOP_POOL_ID" || echo "Failed
 to delete Top-level Pool"
            echo "Waiting for Top-level Pool to be deleted..."
            sleep 10
        fi

        if [ -n "$IPAM_ID" ]; then
            echo "Deleting IPAM: $IPAM_ID"
            aws ec2 delete-ipam --ipam-id "$IPAM_ID" || echo "Failed to delete IPAM"
        fi

        echo "Cleanup completed."
    else
        echo "Cleanup skipped. Resources will remain in your account."
    fi
}

# Function to wait for a pool to be in the 'create-complete' state
wait_for_pool() {
    local pool_id=$1
    local max_attempts=30
    local attempt=1
    local state=""

    echo "Waiting for pool $pool_id to be available..."

    while [ $attempt -le $max_attempts ]; do
        state=$(aws ec2 describe-ipam-pools --ipam-pool-ids "$pool_id" --query
 'IpamPools[0].State' --output text)

        if [ "$state" = "create-complete" ]; then
            echo "Pool $pool_id is now available (state: $state)"
            return 0
        fi

        echo "Attempt $attempt/$max_attempts: Pool $pool_id is in state: $state.
 Waiting..."
        sleep 10
```

```
        ((attempt++))
    done

    echo "Timed out waiting for pool $pool_id to be available"
    return 1
}

# Function to wait for a CIDR to be fully provisioned
wait_for_cidr_provisioning() {
    local pool_id=$1
    local cidr=$2
    local max_attempts=30
    local attempt=1
    local state=""

    echo "Waiting for CIDR $cidr to be fully provisioned in pool $pool_id..."

    while [ $attempt -le $max_attempts ]; do
        state=$(aws ec2 get-ipam-pool-cidrs --ipam-pool-id "$pool_id" --query
 "IpamPoolCidrs[?Cidr=='$cidr'].State" --output text)

        if [ "$state" = "provisioned" ]; then
            echo "CIDR $cidr is now fully provisioned (state: $state)"
            return 0
        fi

        echo "Attempt $attempt/$max_attempts: CIDR $cidr is in state: $state.
 Waiting..."
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for CIDR $cidr to be provisioned"
    return 1
}

# Step 1: Create an IPAM
echo "Creating IPAM..."
IPAM_RESULT=$(aws ec2 create-ipam \
    --description "My IPAM" \
    --operating-regions RegionName=us-east-1 RegionName=us-west-2)

if [ $? -ne 0 ]; then
    handle_error "Failed to create IPAM"
```

```
fi

IPAM_ID=$(echo "$IPAM_RESULT" | grep -o '"IpamId": "[^"]*' | cut -d'"' -f4)
echo "IPAM created with ID: $IPAM_ID"

# Wait for IPAM to be created and available
echo "Waiting for IPAM to be available..."
sleep 20

# Step 2: Get the IPAM Scope ID - FIXED to correctly identify the private scope
echo "Getting IPAM Scope ID..."
SCOPE_RESULT=$(aws ec2 describe-ipams --ipam-id "$IPAM_ID")

if [ $? -ne 0 ]; then
    handle_error "Failed to get IPAM details"
fi

# Extract the private scope ID directly from the IPAM details
PRIVATE_SCOPE_ID=$(echo "$SCOPE_RESULT" | grep -o '"PrivateDefaultScopeId": "[^"]*'
 | cut -d'"' -f4)
echo "Private Scope ID: $PRIVATE_SCOPE_ID"

if [ -z "$PRIVATE_SCOPE_ID" ]; then
    handle_error "Failed to get Private Scope ID"
fi

# Step 3: Create a Top-Level IPv4 Pool
echo "Creating Top-level IPv4 Pool..."
TOP_POOL_RESULT=$(aws ec2 create-ipam-pool \
    --ipam-scope-id "$PRIVATE_SCOPE_ID" \
    --address-family ipv4 \
    --description "Top-level pool")

if [ $? -ne 0 ]; then
    handle_error "Failed to create Top-level Pool"
fi

TOP_POOL_ID=$(echo "$TOP_POOL_RESULT" | grep -o '"IpamPoolId": "[^"]*' | cut -d'"' -
f4)
echo "Top-level Pool created with ID: $TOP_POOL_ID"

# Wait for the top-level pool to be available
if ! wait_for_pool "$TOP_POOL_ID"; then
    handle_error "Top-level Pool did not become available in time"
```

```
fi

# Provision CIDR to the top-level pool
echo "Provisioning CIDR to Top-level Pool..."
TOP_POOL_CIDR="10.0.0.0/8"
PROVISION_RESULT=$(aws ec2 provision-ipam-pool-cidr \
    --ipam-pool-id "$TOP_POOL_ID" \
    --cidr "$TOP_POOL_CIDR")

if [ $? -ne 0 ]; then
    handle_error "Failed to provision CIDR to Top-level Pool"
fi

echo "$PROVISION_RESULT"

# Wait for the CIDR to be fully provisioned
if ! wait_for_cidr_provisioning "$TOP_POOL_ID" "$TOP_POOL_CIDR"; then
    handle_error "CIDR provisioning to Top-level Pool did not complete in time"
fi

# Step 4: Create a Regional IPv4 Pool
echo "Creating Regional IPv4 Pool..."
REGIONAL_POOL_RESULT=$(aws ec2 create-ipam-pool \
    --ipam-scope-id "$PRIVATE_SCOPE_ID" \
    --source-ipam-pool-id "$TOP_POOL_ID" \
    --locale us-east-1 \
    --address-family ipv4 \
    --description "Regional pool in us-east-1")

if [ $? -ne 0 ]; then
    handle_error "Failed to create Regional Pool"
fi

REGIONAL_POOL_ID=$(echo "$REGIONAL_POOL_RESULT" | grep -o '"IpamPoolId": "[^"]*' |
 cut -d'"' -f4)
echo "Regional Pool created with ID: $REGIONAL_POOL_ID"

# Wait for the regional pool to be available
if ! wait_for_pool "$REGIONAL_POOL_ID"; then
    handle_error "Regional Pool did not become available in time"
fi

# Provision CIDR to the regional pool
echo "Provisioning CIDR to Regional Pool..."
```

```
REGIONAL_POOL_CIDR="10.0.0.0/16"
PROVISION_RESULT=$(aws ec2 provision-ipam-pool-cidr \
    --ipam-pool-id "$REGIONAL_POOL_ID" \
    --cidr "$REGIONAL_POOL_CIDR")

if [ $? -ne 0 ]; then
    handle_error "Failed to provision CIDR to Regional Pool"
fi

echo "$PROVISION_RESULT"

# Wait for the CIDR to be fully provisioned
if ! wait_for_cidr_provisioning "$REGIONAL_POOL_ID" "$REGIONAL_POOL_CIDR"; then
    handle_error "CIDR provisioning to Regional Pool did not complete in time"
fi

# Step 5: Create a Development IPv4 Pool - FIXED to include locale
echo "Creating Development IPv4 Pool..."
DEV_POOL_RESULT=$(aws ec2 create-ipam-pool \
    --ipam-scope-id "$PRIVATE_SCOPE_ID" \
    --source-ipam-pool-id "$REGIONAL_POOL_ID" \
    --locale us-east-1 \
    --address-family ipv4 \
    --description "Development pool")

if [ $? -ne 0 ]; then
    handle_error "Failed to create Development Pool"
fi

DEV_POOL_ID=$(echo "$DEV_POOL_RESULT" | grep -o '"IpamPoolId": "[^"]*' | cut -d'"' -
f4)
echo "Development Pool created with ID: $DEV_POOL_ID"

# Wait for the development pool to be available
if ! wait_for_pool "$DEV_POOL_ID"; then
    handle_error "Development Pool did not become available in time"
fi

# Provision CIDR to the development pool
echo "Provisioning CIDR to Development Pool..."
DEV_POOL_CIDR="10.0.0.0/24"
PROVISION_RESULT=$(aws ec2 provision-ipam-pool-cidr \
    --ipam-pool-id "$DEV_POOL_ID" \
    --cidr "$DEV_POOL_CIDR")
```

```
if [ $? -ne 0 ]; then
    handle_error "Failed to provision CIDR to Development Pool"
fi

echo "$PROVISION_RESULT"

# Wait for the CIDR to be fully provisioned
if ! wait_for_cidr_provisioning "$DEV_POOL_ID" "$DEV_POOL_CIDR"; then
    handle_error "CIDR provisioning to Development Pool did not complete in time"
fi

# Step 6: Create a VPC Using an IPAM Pool CIDR - FIXED to use the correct parameter
 names and a smaller netmask length
echo "Creating VPC using IPAM Pool CIDR..."
VPC_RESULT=$(aws ec2 create-vpc \
    --ipv4-ipam-pool-id "$DEV_POOL_ID" \
    --ipv4-netmask-length 26 \
    --tag-specifications 'ResourceType=vpc,Tags=[{Key=Name,Value=IPAM-VPC}]')

if [ $? -ne 0 ]; then
    handle_error "Failed to create VPC"
fi

VPC_ID=$(echo "$VPC_RESULT" | grep -o '"VpcId": "[^"]*' | cut -d'"' -f4)
echo "VPC created with ID: $VPC_ID"

# Step 7: Verify the IPAM Pool Allocation
echo "Verifying IPAM Pool Allocation..."
ALLOCATION_RESULT=$(aws ec2 get-ipam-pool-allocations \
    --ipam-pool-id "$DEV_POOL_ID")

if [ $? -ne 0 ]; then
    handle_error "Failed to verify IPAM Pool Allocation"
fi

echo "IPAM Pool Allocation verified:"
echo "$ALLOCATION_RESULT" | grep -A 5 "Allocations"

echo ""
echo "IPAM setup completed successfully!"
echo ""

# Prompt for cleanup
```

```
cleanup_resources

echo "Script completed at $(date)"
exit 0
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateIpam
  - CreateIpamPool
  - CreateVpc
  - DeleteIpam
  - DeleteIpamPool
  - DeleteVpc
  - DeprovisionIpamPoolCidr
  - DescribeIpamPools
  - DescribeIpams
  - DescribeVpcs
  - GetIpamPoolAllocations
  - GetIpamPoolCidrs
  - ProvisionIpamPoolCidr

使用 Amazon CLI 及 Bash 脚本的 HealthImaging 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 HealthImaging 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`CreateDatastore`

以下代码示例演示了如何使用 `CreateDatastore`。

Amazon CLI 及 Bash 脚本

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function imaging_create_datastore
#
# This function creates an AWS HealthImaging data store for importing DICOM P10
 files.
#
# Parameters:
#       -n data_store_name - The name of the data store.
#
# Returns:
#       The datastore ID.
#    And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function imaging_create_datastore() {
  local datastore_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function imaging_create_datastore"
    echo "Creates an AWS HealthImaging data store for importing DICOM P10 files."
    echo "  -n data_store_name - The name of the data store."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) datastore_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$datastore_name" ]]; then
    errecho "ERROR: You must provide a data store name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws medical-imaging create-datastore \
    --datastore-name "$datastore_name" \
    --output text \
    --query 'datastoreId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports medical-imaging create-datastore operation failed.
$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDatastore。

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

`DeleteDatastore`

以下代码示例演示了如何使用 `DeleteDatastore`。

Amazon CLI 及 Bash 脚本

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function imaging_delete_datastore
#
# This function deletes an AWS HealthImaging data store.
#
# Parameters:
#       -i datastore_id - The ID of the data store.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function imaging_delete_datastore() {
  local datastore_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function imaging_delete_datastore"
    echo "Deletes an AWS HealthImaging data store."
    echo "  -i datastore_id - The ID of the data store."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) datastore_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$datastore_id" ]]; then
    errecho "ERROR: You must provide a data store ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws medical-imaging delete-datastore \
    --datastore-id "$datastore_id")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports medical-imaging delete-datastore operation failed.
$response"
    return 1
  fi

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDatastore。

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

`GetDatastore`

以下代码示例演示了如何使用 `GetDatastore`。

Amazon CLI 及 Bash 脚本

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function imaging_get_datastore
#
# Get a data store's properties.
#
# Parameters:
#       -i data_store_id - The ID of the data store.
#
# Returns:
#       [datastore_name, datastore_id, datastore_status, datastore_arn,  created_at,
 updated_at]
#    And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function imaging_get_datastore() {
  local datastore_id option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function imaging_get_datastore"
    echo "Gets a data store's properties."
```

```
    echo "  -i datastore_id - The ID of the data store."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) datastore_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$datastore_id" ]]; then
    errecho "ERROR: You must provide a data store ID with the -i parameter."
    usage
    return 1
  fi

  local response

  response=$(
    aws medical-imaging get-datastore \
      --datastore-id "$datastore_id" \
      --output text \
      --query "[ datastoreProperties.datastoreName,
datastoreProperties.datastoreId, datastoreProperties.datastoreStatus,
datastoreProperties.datastoreArn,  datastoreProperties.createdAt,
datastoreProperties.updatedAt]"
  )
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-datastores operation failed.$response"
    return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetDatastore。

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

`ListDatastores`

以下代码示例演示了如何使用 `ListDatastores`。

Amazon CLI 及 Bash 脚本

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function imaging_list_datastores
#
# List the HealthImaging data stores in the account.
#
# Returns:
#       [[datastore_name, datastore_id, datastore_status]]
#    And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function imaging_list_datastores() {
```

```
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function imaging_list_datastores"
    echo "Lists the AWS HealthImaging data stores in the account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response
  response=$(aws medical-imaging list-datastores \
    --output text \
    --query "datastoreSummaries[*][datastoreName, datastoreId, datastoreStatus]")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-datastores operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListDatastores。

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

使用 Amazon CLI 及 Bash 脚本的 IAM 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 IAM 结合使用，来执行操作和实现常见场景。

基本功能是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 基本功能
- 操作
- 场景

基本功能

了解基本功能

以下代码示例展示了如何创建用户并代入角色。

> **Warning**
>
> 为了避免安全风险，在开发专用软件或处理真实数据时，请勿使用 IAM 用户进行身份验证，而是使用与身份提供商的联合身份验证，例如 Amazon IAM Identity Center。

- 创建没有权限的用户。
- 创建授予列出账户的 Amazon S3 存储桶的权限的角色
- 添加策略以允许用户代入该角色。
- 代入角色并使用临时凭证列出 S3 存储桶，然后清除资源。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
```

```
 echo "This demo will create an IAM user, create an IAM role, and apply the role to
the user."
 echo_repeat "*" 88
 echo

 echo -n "Enter a name for a new IAM user: "
 get_input
 user_name=$get_input_result

 local user_arn
 user_arn=$(iam_create_user -u "$user_name")

 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created demo IAM user named $user_name"
 else
   errecho "$user_arn"
   errecho "The user failed to create. This demo will exit."
   return 1
 fi

 local access_key_response
 access_key_response=$(iam_create_user_access_key -u "$user_name")
 # shellcheck disable=SC2181
 if [[ ${?} != 0 ]]; then
   errecho "The access key failed to create. This demo will exit."
   clean_up "$user_name"
   return 1
 fi

 IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
 local key_name=${access_key_values[0]}
 local key_secret=${access_key_values[1]}

 echo "Created access key named $key_name"

 echo "Wait 10 seconds for the user to be ready."
 sleep 10
 echo_repeat "*" 88
 echo

 local iam_role_name
 iam_role_name=$(generate_random_name "test-role")
 echo "Creating a role named $iam_role_name with user $user_name as the principal."
```

```
local assume_role_policy_document="{
  \"Version\": \"2012-10-17\",
  \"Statement\": [{
      \"Effect\": \"Allow\",
      \"Principal\": {\"AWS\": \"$user_arn\"},
      \"Action\": \"sts:AssumeRole\"
      }]
  }"

local role_arn
role_arn=$(iam_create_role -n "$iam_role_name" -p "$assume_role_policy_document")

# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
  echo "Created IAM role named $iam_role_name"
else
  errecho "The role failed to create. This demo will exit."
  clean_up "$user_name" "$key_name"
  return 1
fi

local policy_name
policy_name=$(generate_random_name "test-policy")
local policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"s3:ListAllMyBuckets\",
                  \"Resource\": \"arn:aws:s3:::*\"}]}"

local policy_arn
policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
# shellcheck disable=SC2181
if [[ ${?} == 0 ]]; then
  echo "Created  IAM policy named $policy_name"
else
  errecho "The policy failed to create."
  clean_up "$user_name" "$key_name" "$iam_role_name"
  return 1
fi

if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
  echo "Attached policy $policy_arn to role $iam_role_name"
```

```
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret
```

```
 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing buckets
failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
"$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)
```

```
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets succeeded
 because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

此场景中使用的 IAM 函数。

```
################################################################################
```

```
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must supply
 a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
```

```
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
```

```
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
```

```
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
```

```
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    k) access_key="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi
```

```
  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must supply
 a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

操作

`AttachRolePolicy`

以下代码示例演示了如何使用 `AttachRolePolicy`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
```

```
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AttachRolePolicy。

`CreateAccessKey`

以下代码示例演示了如何使用 `CreateAccessKey`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
```

```
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateAccessKey。

`CreatePolicy`

以下代码示例演示了如何使用 `CreatePolicy`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
  echo "function iam_create_policy"
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreatePolicy。

`CreateRole`

以下代码示例演示了如何使用 `CreateRole`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
```

```
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateRole。

`CreateUser`

以下代码示例演示了如何使用 `CreateUser`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function iecho
```

```
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must supply
 a username:"
```

```
  echo "  -u user_name    The name of the user. It must be unique within the
account."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name already exists in the account."
  return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
  --output text \
  --query 'User.Arn')

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateUser。

`DeleteAccessKey`

以下代码示例演示了如何使用 `DeleteAccessKey`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
```

```
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteAccessKey。

`DeletePolicy`

以下代码示例演示了如何使用 `DeletePolicy`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeletePolicy。

`DeleteRole`

以下代码示例演示了如何使用 `DeleteRole`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
```

```
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteRole。

`DeleteUser`

以下代码示例演示了如何使用 `DeleteUser`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
```

```
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must supply
 a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user does not exist, we don't want to try to delete it.
if (! iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name does not exist in the account."
  return 1
fi

response=$(aws iam delete-user \
  --user-name "$user_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-user operation failed.$response"
  return 1
fi

iecho "delete-user response:$response"
iecho

return 0
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteUser。

`DetachRolePolicy`

以下代码示例演示了如何使用 `DetachRolePolicy`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_detach_role_policy"
   echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
   echo "  -n role_name   The name of the IAM role."
   echo "  -p policy_ARN -- The IAM policy document ARN."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:p:h" option; do
   case "${option}" in
     n) role_name="${OPTARG}" ;;
     p) policy_arn="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$role_name" ]]; then
   errecho "ERROR: You must provide a role name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$policy_arn" ]]; then
   errecho "ERROR: You must provide a policy ARN with the -p parameter."
   usage
   return 1
 fi

 response=$(aws iam detach-role-policy \
   --role-name "$role_name" \
   --policy-arn "$policy_arn")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DetachRolePolicy。

`GetUser`

以下代码示例演示了如何使用 `GetUser`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetUser。

ListAccessKeys

以下代码示例演示了如何使用 ListAccessKeys。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for the
 specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }
```

```
  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListAccessKeys。

`ListUsers`

以下代码示例演示了如何使用 `ListUsers`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the account."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListUsers。

UpdateAccessKey

以下代码示例演示了如何使用 UpdateAccessKey。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
###############################################################################
# function iam_update_access_key
#
# This function can activate or deactivate an IAM access key for the specified IAM
 user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to update.
#       -a            -- Activate the selected access key.
#       -d            -- Deactivate the selected access key.
#
# Example:
#       # To deactivate the selected access key for IAM user Bob
#       iam_update_access_key -u Bob -k AKIAIOSFODNN7EXAMPLE -d
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_update_access_key() {
  local user_name access_key status response
  local option OPTARG # Required to use getopts command in a function.
  local activate_flag=false deactivate_flag=false

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_update_access_key"
    echo "Updates the status of an AWS Identity and Access Management (IAM) access
 key for the specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to update."
    echo "  -a              Activate the access key."
    echo "  -d              Deactivate the access key."
```

```
    echo ""
 }

 # Retrieve the calling parameters.
   while getopts "u:k:adh" option; do
     case "${option}" in
       u) user_name="${OPTARG}" ;;
       k) access_key="${OPTARG}" ;;
       a) activate_flag=true ;;
       d) deactivate_flag=true ;;
       h)
         usage
         return 0
         ;;
       \?)
         echo "Invalid parameter"
         usage
         return 1
         ;;
     esac
   done
   export OPTIND=1

  # Validate input parameters
   if [[ -z "$user_name" ]]; then
     errecho "ERROR: You must provide a username with the -u parameter."
     usage
     return 1
   fi

   if [[ -z "$access_key" ]]; then
     errecho "ERROR: You must provide an access key with the -k parameter."
     usage
     return 1
   fi

   # Ensure that only -a or -d is specified
   if [[ "$activate_flag" == true && "$deactivate_flag" == true ]]; then
     errecho "ERROR: You cannot specify both -a (activate) and -d (deactivate) at
the same time."
     usage
     return 1
   fi
```

```
    # If neither -a nor -d is provided, return an error
    if [[ "$activate_flag" == false && "$deactivate_flag" == false ]]; then
      errecho "ERROR: You must specify either -a (activate) or -d (deactivate)."
      usage
      return 1
    fi

    # Determine the status based on the flag
    if [[ "$activate_flag" == true ]]; then
      status="Active"
    elif [[ "$deactivate_flag" == true ]]; then
      status="Inactive"
    fi

    iecho "Parameters:\n"
    iecho "    Username:   $user_name"
    iecho "    Access key: $access_key"
    iecho "    New status: $status"
    iecho ""

    # Update the access key status
    response=$(aws iam update-access-key \
      --user-name "$user_name" \
      --access-key-id "$access_key" \
      --status "$status" 2>&1)

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log $error_code
      errecho "ERROR: AWS reports update-access-key operation failed.\n$response"
      return 1
    fi

    iecho "update-access-key response: $response"
    iecho

    return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 UpdateAccessKey。

场景

设置基于属性的访问控制

以下代码示例展示了如何为 DynamoDB 实现基于属性的访问权限控制（ABAC）。

- 创建用于实现 ABAC 的 IAM 策略。
- 为不同部门创建带有标签的表。
- 根据标签列出并筛选表。

Amazon CLI 及 Bash 脚本

创建用于实现 ABAC 的 IAM 策略。

```
# Step 1: Create a policy document for ABAC
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
```

```
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

为不同部门创建带有标签的表。

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
        Key=Environment,Value=Development

# Create another table with different tags
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

根据标签列出并筛选表。

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text | xargs
 -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn" --output
 text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreatePolicy
  - CreateTable
  - ListTables

处理 Streams 和生存时间

以下代码示例演示了如何管理 DynamoDB Streams 和生存时间功能。

- 创建一个启用了 Streams 的表。
- 描述 Streams。
- 创建用于处理 Streams 的 Lambda 函数。
- 在表上启用 TTL。
- 添加具有 TTL 属性的项目。
- 描述 TTL 设置。

Amazon CLI 及 Bash 脚本

创建一个启用了 Streams 的表。

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

描述 Streams。

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"
```

```
# Describe the stream
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

为 Streams 创建 Lambda 函数。

```
# Step 1: Create an IAM role for the Lambda function
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

在表上启用 TTL。

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo

# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

添加具有 TTL 属性的项目。

```
# Calculate expiration time (current time + 1 day in seconds)
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
```

```
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

描述 TTL 设置。

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
    --table-name TTLDemo
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AttachRolePolicy
  - CreateRole
  - CreateTable
  - DescribeTable
  - DescribeTimeToLive
  - PutItem
  - UpdateTimeToLive

使用 Amazon CLI 及 Bash 脚本的 Amazon KMS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon KMS 结合使用，来执行操作和实现常见场景。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 场景

场景

处理表加密

以下代码示例演示了如何管理 DynamoDB 表的加密选项。

- 创建具有默认加密的表。
- 创建具有客户托管式 CMK 的表。
- 更新表加密设置。
- 描述表加密。

Amazon CLI 及 Bash 脚本

创建具有默认加密的表。

```
# Create a table with default encryption (AWS owned key)
aws dynamodb create-table \
    --table-name CustomerData \
    --attribute-definitions \
        AttributeName=CustomerID,AttributeType=S \
    --key-schema \
        AttributeName=CustomerID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS
```

创建具有客户托管式 CMK 的表。

```
# Step 1: Create a customer managed key in KMS
aws kms create-key \
    --description "Key for DynamoDB table encryption" \
    --key-usage ENCRYPT_DECRYPT \
    --customer-master-key-spec SYMMETRIC_DEFAULT

# Store the key ID for later use
KEY_ID=$(aws kms list-keys --query "Keys[?contains(KeyArn, 'Key for
 DynamoDB')].KeyId" --output text)

# Step 2: Create a table with the customer managed key
aws dynamodb create-table \
    --table-name SensitiveData \
```

```
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

更新表加密。

```
# Update a table to use a different KMS key
aws dynamodb update-table \
    --table-name CustomerData \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

描述表加密。

```
# Describe the table to see encryption settings
aws dynamodb describe-table \
    --table-name CustomerData \
    --query "Table.SSEDescription"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CreateKey
  - CreateTable
  - DescribeTable
  - UpdateTable

使用 Amazon CLI 及 Bash 脚本的 Lightsail 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Lightsail 结合使用，来执行操作和实现常见场景。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 场景

场景

开始使用 Lightsail

以下代码示例展示了如何：

- 浏览可用的蓝图和捆绑包以创建实例
- 使用 Amazon Linux 2023 创建 Lightsail 实例
- 监控实例状态直到运行
- 下载 SSH 访问的默认密钥对
- 创建并附加额外的块存储磁盘
- 创建实例快照以进行备份
- 清理资源以避免持续收费

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# Amazon Lightsail Getting Started CLI Script
# This script demonstrates how to create and manage Lightsail resources using the
 AWS CLI

# FIXES APPLIED:
# 1. Added polling mechanism to check disk state before attaching
# 2. Added polling mechanism to check snapshot state before proceeding with cleanup
# 3. Set AWS_REGION variable to us-west-2 for consistent region usage

# Set AWS region
export AWS_REGION="us-west-2"
```

```
echo "Using AWS region: $AWS_REGION"

# Set up logging
LOG_FILE="lightsail-script.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Lightsail Getting Started script at $(date)"

# Error handling function
handle_error() {
  echo "ERROR: $1"
  echo "Attempting to clean up resources..."
  cleanup_resources
  exit 1
}

# Function to check if a command succeeded
check_status() {
  if [ $? -ne 0 ]; then
    handle_error "$1"
  fi
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
INSTANCE_NAME="LightsailInstance-${RANDOM_ID}"
DISK_NAME="LightsailDisk-${RANDOM_ID}"
SNAPSHOT_NAME="LightsailSnapshot-${RANDOM_ID}"

# Array to track created resources
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
  CREATED_RESOURCES+=("$1:$2")
  echo "Created $1: $2"
}

# Function to clean up resources
cleanup_resources() {
  echo "Resources created by this script:"
  for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  $resource"
  done
```

```
  # Reverse the array to delete resources in reverse order
  for (( idx=${#CREATED_RESOURCES[@]}-1 ; idx>=0 ; idx-- )); do
    IFS=':' read -r type name <<< "${CREATED_RESOURCES[idx]}"

    case "$type" in
      "instance_snapshot")
        echo "Deleting instance snapshot: $name"
        aws lightsail delete-instance-snapshot --instance-snapshot-name "$name" --
region $AWS_REGION
        ;;
      "disk_snapshot")
        echo "Deleting disk snapshot: $name"
        aws lightsail delete-disk-snapshot --disk-snapshot-name "$name" --region
 $AWS_REGION
        ;;
      "disk")
        echo "Detaching disk: $name"
        aws lightsail detach-disk --disk-name "$name" --region $AWS_REGION
        sleep 10 # Wait for detach to complete
        echo "Deleting disk: $name"
        aws lightsail delete-disk --disk-name "$name" --region $AWS_REGION
        ;;
      "instance")
        echo "Deleting instance: $name"
        # Check instance state before attempting to delete
        INSTANCE_STATE=$(aws lightsail get-instance-state --instance-name "$name" --
region $AWS_REGION --query 'state.name' --output text 2>/dev/null)
        if [ "$INSTANCE_STATE" == "pending" ]; then
          echo "Instance is in pending state. Waiting for it to be ready before
 deleting..."
          MAX_WAIT=30
          WAITED=0
          while [ "$INSTANCE_STATE" == "pending" ] && [ $WAITED -lt $MAX_WAIT ]; do
            sleep 10
            WAITED=$((WAITED+1))
            INSTANCE_STATE=$(aws lightsail get-instance-state --instance-name
 "$name" --region $AWS_REGION --query 'state.name' --output text 2>/dev/null)
            echo "Instance state: $INSTANCE_STATE"
          done
        fi
        aws lightsail delete-instance --instance-name "$name" --region $AWS_REGION
        ;;
    esac
```

```
  done

  echo "Cleanup completed"
}

# Step 1: Verify AWS CLI configuration
echo "Step 1: Verifying AWS CLI configuration"
aws configure list
check_status "Failed to verify AWS CLI configuration"

# Step 2: Get available blueprints and bundles
echo "Step 2: Getting available blueprints and bundles"
echo "Available blueprints (showing first 5):"
aws lightsail get-blueprints --region $AWS_REGION --query 'blueprints[0:5].
[blueprintId,name]' --output table
check_status "Failed to get blueprints"

echo "Available bundles (showing first 5):"
aws lightsail get-bundles --region $AWS_REGION --query 'bundles[0:5].
[bundleId,name,price]' --output table
check_status "Failed to get bundles"

# Get available regions and availability zones
echo "Getting available regions and availability zones"
# Use a specific availability zone in us-west-2 region
AVAILABILITY_ZONE="us-west-2a"
echo "Using availability zone: $AVAILABILITY_ZONE"

# Step 3: Create a Lightsail instance
echo "Step 3: Creating Lightsail instance: $INSTANCE_NAME"
aws lightsail create-instances \
  --instance-names "$INSTANCE_NAME" \
  --availability-zone "$AVAILABILITY_ZONE" \
  --blueprint-id amazon_linux_2023 \
  --bundle-id nano_3_0 \
  --region $AWS_REGION
check_status "Failed to create Lightsail instance"
track_resource "instance" "$INSTANCE_NAME"

# Wait for the instance to be in a running state
echo "Waiting for instance to be in running state..."
# Wait for the instance to be ready (polling approach)
MAX_ATTEMPTS=30
ATTEMPTS=0
```

```
while [ $ATTEMPTS -lt $MAX_ATTEMPTS ]; do
  STATUS=$(aws lightsail get-instance-state --instance-name "$INSTANCE_NAME" --
region $AWS_REGION --query 'state.name' --output text)
  if [ "$STATUS" == "running" ]; then
    echo "Instance is now running"
    break
  fi
  echo "Instance status: $STATUS. Waiting..."
  ATTEMPTS=$((ATTEMPTS+1))
  sleep 10
done

if [ $ATTEMPTS -eq $MAX_ATTEMPTS ]; then
  handle_error "Instance failed to reach running state after 5 minutes"
fi

# Get instance details
echo "Getting instance details"
INSTANCE_IP=$(aws lightsail get-instance --instance-name "$INSTANCE_NAME" --region
 $AWS_REGION --query 'instance.publicIpAddress' --output text)
check_status "Failed to get instance IP address"
echo "Instance IP address: $INSTANCE_IP"

# Step 4: Download the default key pair
echo "Step 4: Downloading default key pair"
KEY_FILE="lightsail_key_${RANDOM_ID}.pem"
aws lightsail download-default-key-pair --region $AWS_REGION --output text >
 "$KEY_FILE"
check_status "Failed to download key pair"
chmod 400 "$KEY_FILE"
check_status "Failed to set permissions on key pair"
echo "Key pair downloaded to $KEY_FILE"

echo "To connect to your instance, use:"
echo "ssh -i $KEY_FILE ec2-user@$INSTANCE_IP"

# Step 5: Create a block storage disk
echo "Step 5: Creating block storage disk: $DISK_NAME"
aws lightsail create-disk \
  --disk-name "$DISK_NAME" \
  --availability-zone "$AVAILABILITY_ZONE" \
  --size-in-gb 8 \
  --region $AWS_REGION
check_status "Failed to create disk"
```

```
track_resource "disk" "$DISK_NAME"

# FIX: Wait for the disk to be available using polling instead of fixed sleep
echo "Waiting for disk to be available..."
MAX_ATTEMPTS=30
ATTEMPTS=0
while [ $ATTEMPTS -lt $MAX_ATTEMPTS ]; do
  DISK_STATE=$(aws lightsail get-disk --disk-name "$DISK_NAME" --region $AWS_REGION
 --query 'disk.state' --output text 2>/dev/null)
  if [ "$DISK_STATE" == "available" ]; then
    echo "Disk is now available"
    break
  fi
  echo "Disk status: $DISK_STATE. Waiting..."
  ATTEMPTS=$((ATTEMPTS+1))
  sleep 10
done

if [ $ATTEMPTS -eq $MAX_ATTEMPTS ]; then
  handle_error "Disk failed to become available after 5 minutes"
fi

# Attach the disk to the instance
echo "Attaching disk to instance"
aws lightsail attach-disk \
  --disk-name "$DISK_NAME" \
  --instance-name "$INSTANCE_NAME" \
  --disk-path /dev/xvdf \
  --region $AWS_REGION
check_status "Failed to attach disk to instance"

echo "Disk attached. To format and mount the disk, connect to your instance and
 run:"
echo "sudo mkfs -t ext4 /dev/xvdf"
echo "sudo mkdir -p /mnt/my-data"
echo "sudo mount /dev/xvdf /mnt/my-data"
echo "sudo chown ec2-user:ec2-user /mnt/my-data"

# Step 6: Create a snapshot of the instance
echo "Step 6: Creating snapshot of the instance: $SNAPSHOT_NAME"
aws lightsail create-instance-snapshot \
  --instance-name "$INSTANCE_NAME" \
  --instance-snapshot-name "$SNAPSHOT_NAME" \
  --region $AWS_REGION
```

```
check_status "Failed to create instance snapshot"
track_resource "instance_snapshot" "$SNAPSHOT_NAME"

# FIX: Wait for the snapshot to complete using polling instead of fixed sleep
echo "Waiting for snapshot to complete... (this may take several minutes)"
MAX_ATTEMPTS=60  # Increased timeout for snapshot creation
ATTEMPTS=0
while [ $ATTEMPTS -lt $MAX_ATTEMPTS ]; do
  SNAPSHOT_STATE=$(aws lightsail get-instance-snapshot --instance-snapshot-name
 "$SNAPSHOT_NAME" --region $AWS_REGION --query 'instanceSnapshot.state' --output
 text 2>/dev/null)
  if [ "$SNAPSHOT_STATE" == "completed" ]; then
    echo "Snapshot creation completed"
    break
  fi
  echo "Snapshot status: $SNAPSHOT_STATE. Waiting... ($ATTEMPTS/$MAX_ATTEMPTS)"
  ATTEMPTS=$((ATTEMPTS+1))
  sleep 10
done

if [ $ATTEMPTS -eq $MAX_ATTEMPTS ]; then
  echo "Warning: Snapshot creation is taking longer than expected but will continue
 in the background."
  echo "You can check its status later with: aws lightsail get-instance-snapshot --
instance-snapshot-name $SNAPSHOT_NAME --region $AWS_REGION"
fi

# Step 7: Clean up resources
echo "Step 7: Clean up resources"
echo "The script has created the following resources:"
for resource in "${CREATED_RESOURCES[@]}"; do
  echo "  $resource"
done

read -p "Do you want to clean up these resources? (y/n): " CLEANUP_CONFIRM
if [[ "$CLEANUP_CONFIRM" == "y" || "$CLEANUP_CONFIRM" == "Y" ]]; then
  cleanup_resources
else
  echo "Resources will not be cleaned up. You can manually delete them later."
  echo "To clean up manually, use the following commands:"
  echo "aws lightsail delete-instance-snapshot --instance-snapshot-name
 $SNAPSHOT_NAME --region $AWS_REGION"
  echo "aws lightsail detach-disk --disk-name $DISK_NAME --region $AWS_REGION"
  echo "aws lightsail delete-disk --disk-name $DISK_NAME --region $AWS_REGION"
```

```
  echo "aws lightsail delete-instance --instance-name $INSTANCE_NAME --region
 $AWS_REGION"
fi

echo "Script completed at $(date)"
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - AttachDisk
  - CreateDisk
  - CreateInstanceSnapshot
  - CreateInstances
  - DeleteDisk
  - DeleteInstance
  - DeleteInstanceSnapshot
  - DetachDisk
  - DownloadDefaultKeyPair
  - GetBlueprints
  - GetBundles
  - GetDisk
  - :GetInstance
  - GetInstanceSnapshot
  - GetInstanceState

使用 Amazon CLI 及 Bash 脚本的 Amazon S3 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon S3 结合使用，来执行操作和实现常见场景。

基本功能是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您演示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 结合来完成特定任务的代码示例。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 基本功能
- 操作
- 场景

基本功能

了解基本功能

以下代码示例展示了如何：

- 创建桶并将文件上载到其中。
- 从桶中下载对象。
- 将对象复制到存储桶中的子文件夹。
- 列出存储桶中的对象。
- 删除存储桶及其对象。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function s3_getting_started
#
# This function creates, copies, and deletes S3 buckets and objects.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function s3_getting_started() {
```

```
 {
   if [ "$BUCKET_OPERATIONS_SOURCED" != "True" ]; then
     cd bucket-lifecycle-operations || exit

     source ./bucket_operations.sh
     cd ..
   fi
 }

 echo_repeat "*" 88
 echo "Welcome to the Amazon S3 getting started demo."
 echo_repeat "*" 88
   echo "A unique bucket will be created by appending a Universally Unique
Identifier to a bucket name prefix."
   echo -n "Enter a prefix for the S3 bucket that will be used in this demo: "
   get_input
   bucket_name_prefix=$get_input_result
 local bucket_name
 bucket_name=$(generate_random_name "$bucket_name_prefix")

 local region_code
 region_code=$(aws configure get region)

 if create_bucket -b "$bucket_name" -r "$region_code"; then
   echo "Created demo bucket named $bucket_name"
 else
   errecho "The bucket failed to create. This demo will exit."
   return 1
 fi

 local file_name
 while [ -z "$file_name" ]; do
   echo -n "Enter a file you want to upload to your bucket: "
   get_input
   file_name=$get_input_result

   if [ ! -f "$file_name" ]; then
     echo "Could not find file $file_name. Are you sure it exists?"
     file_name=""
   fi
 done

 local key
 key="$(basename "$file_name")"
```

```
  local result=0
  if copy_file_to_bucket "$bucket_name" "$file_name" "$key"; then
    echo "Uploaded file $file_name into bucket $bucket_name with key $key."
  else
    result=1
  fi

  local destination_file
  destination_file="$file_name.download"
  if yes_no_input "Would you like to download $key to the file $destination_file?
 (y/n) "; then
    if download_object_from_bucket "$bucket_name" "$destination_file" "$key"; then
      echo "Downloaded $key in the bucket $bucket_name to the file
 $destination_file."
    else
      result=1
    fi
  fi

  if yes_no_input "Would you like to copy $key a new object key in your bucket? (y/
n) "; then
    local to_key
    to_key="demo/$key"
    if copy_item_in_bucket "$bucket_name" "$key" "$to_key"; then
      echo "Copied $key in the bucket $bucket_name to the  $to_key."
    else
      result=1
    fi
  fi

  local bucket_items
  bucket_items=$(list_items_in_bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    result=1
  fi

  echo "Your bucket contains the following items."
  echo -e "Name\t\tSize"
  echo "$bucket_items"
```

```
  if yes_no_input "Delete the bucket, $bucket_name, as well as the objects in it?
 (y/n) "; then
    bucket_items=$(echo "$bucket_items" | cut -f 1)

    if delete_items_in_bucket "$bucket_name" "$bucket_items"; then
      echo "The following items were deleted from the bucket $bucket_name"
      echo "$bucket_items"
    else
      result=1
    fi

    if delete_bucket "$bucket_name"; then
      echo "Deleted the bucket $bucket_name"
    else
      result=1
    fi
  fi

  return $result
}
```

此场景中使用的 Amazon S3 函数。

```
###############################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function create_bucket() {
  local bucket_name region_code response
```

```
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi

  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi

  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
```

```
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}

################################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
  source_file=$2
  destination_file_name=$3

  response=$(aws s3api put-object \
    --bucket "$bucket_name" \
    --body "$source_file" \
```

```
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function copy_item_in_bucket
#
```

```
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
  done
  delete_items=${delete_items%?} # Remove the final comma.
```

```
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n$response"
    return 1
  fi
}

################################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CopyObject
  - CreateBucket

- DeleteBucket
- DeleteObjects
- GetObject
- ListObjectsV2
- PutObject

操作

`CopyObject`

以下代码示例演示了如何使用 `CopyObject`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
################################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CopyObject。

`CreateBucket`

以下代码示例演示了如何使用 `CreateBucket`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
################################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
################################################################################
```

```
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function create_bucket() {
  local bucket_name region_code response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi

  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi

  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
```

```
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateBucket。

`DeleteBucket`

以下代码示例演示了如何使用 `DeleteBucket`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
```

```
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteBucket。

`DeleteObject`

以下代码示例演示了如何使用 `DeleteObject`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
```

```
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_item_in_bucket
#
# This function deletes the specified file from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - The key (file name) in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_item_in_bucket() {
  local bucket_name=$1
  local key=$2
  local response

  response=$(aws s3api delete-object \
    --bucket "$bucket_name" \
    --key "$key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObject。

`DeleteObjects`

以下代码示例演示了如何使用 `DeleteObjects`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
  done
  delete_items=${delete_items%?} # Remove the final comma.
```

```
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteObjects。

`GetObject`

以下代码示例演示了如何使用 `GetObject`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
```

```
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 GetObject。

`HeadBucket`

以下代码示例演示了如何使用 `HeadBucket`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function bucket_exists
#
# This function checks to see if the specified bucket already exists.
#
# Parameters:
#       $1 - The name of the bucket to check.
#
# Returns:
#       0 - If the bucket already exists.
#       1 - If the bucket doesn't exist.
###############################################################################
function bucket_exists() {
  local bucket_name
  bucket_name=$1

  # Check whether the bucket already exists.
  # We suppress all output - we're interested only in the return code.

  if aws s3api head-bucket \
    --bucket "$bucket_name" \
    >/dev/null 2>&1; then
    return 0 # 0 in Bash script means true.
  else
    return 1 # 1 in Bash script means false.
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 HeadBucket。

`ListObjectsV2`

以下代码示例演示了如何使用 `ListObjectsV2`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ListObjectsV2。

`PutObject`

以下代码示例演示了如何使用 `PutObject`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
  source_file=$2
  destination_file_name=$3
```

```
  response=$(aws s3api put-object \
    --bucket "$bucket_name" \
    --body "$source_file" \
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 PutObject。

场景

开始使用 S3

以下代码示例展示了如何：

- 创建具有唯一命名和区域配置的 S3 存储桶
- 配置存储桶安全设置，包括屏蔽公共访问权限
- 启用版本控制和默认加密以保护数据
- 上传带有和不带自定义元数据的对象
- 将对象从存储桶下载到本地存储
- 复制存储桶中的对象以整理文件夹中的数据
- 列出存储桶内容和带有特定前缀的对象
- 向存储桶添加标签以进行资源管理
- 清理所有资源，包括受版本控制的对象

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。在 Sample developer tutorials 存储库中查找完整示例，了解如何进行设置和运行。

```
#!/bin/bash

# Amazon S3 Getting Started Tutorial Script
# This script demonstrates basic S3 operations including:
# - Creating a bucket
# - Configuring bucket settings
# - Uploading, downloading, and copying objects
# - Deleting objects and buckets

# Latest fixes:
# 1. Fixed folder creation using temporary file
# 2. Corrected versioned object deletion in cleanup
# 3. Improved error handling for cleanup operations

# Set up error handling
set -e
trap 'cleanup_handler $?' EXIT

# Log file setup
LOG_FILE="s3-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

# Function to log messages
log() {
    echo "[$(date +"%Y-%m-%d %H:%M:%S")] $1"
}

# Function to handle errors
handle_error() {
    log "ERROR: $1"
    exit 1
}

# Function to check if a bucket exists
```

```
bucket_exists() {
    if aws s3api head-bucket --bucket "$1" 2>/dev/null; then
        return 0
    else
        return 1
    fi
}

# Function to delete all versions of objects in a bucket
delete_all_versions() {
    local bucket=$1
    log "Deleting all object versions from bucket $bucket..."

    # Get and delete all versions
    versions=$(aws s3api list-object-versions --bucket "$bucket" --query
 'Versions[].{Key:Key,VersionId:VersionId}' --output json 2>/dev/null)
    if [ -n "$versions" ] && [ "$versions" != "null" ]; then
        echo "{\"Objects\": $versions}" | aws s3api delete-objects --bucket
 "$bucket" --delete file:///dev/stdin >/dev/null 2>&1 || log "Warning: Some versions
 could not be deleted"
    fi

    # Get and delete all delete markers
    markers=$(aws s3api list-object-versions --bucket "$bucket" --query
 'DeleteMarkers[].{Key:Key,VersionId:VersionId}' --output json 2>/dev/null)
    if [ -n "$markers" ] && [ "$markers" != "null" ]; then
        echo "{\"Objects\": $markers}" | aws s3api delete-objects --bucket "$bucket"
 --delete file:///dev/stdin >/dev/null 2>&1 || log "Warning: Some delete markers
 could not be deleted"
    fi
}

# Function to handle cleanup on exit
cleanup_handler() {
    local exit_code=$1

    # Only run cleanup if it hasn't been run already
    if [ -z "$CLEANUP_DONE" ]; then
        cleanup
    fi

    exit $exit_code
}
```

```
# Function to clean up resources
cleanup() {
    log "Starting cleanup process..."
    CLEANUP_DONE=1

    # List all resources created for confirmation
    log "Resources created:"
    if [ -n "$BUCKET_NAME" ]; then
        log "- S3 Bucket: $BUCKET_NAME"

        # Only try to list objects if the bucket exists
        if bucket_exists "$BUCKET_NAME"; then
            # Check if any objects were created
            OBJECTS=$(aws s3api list-objects-v2 --bucket "$BUCKET_NAME" --query
 'Contents[].Key' --output text 2>/dev/null || echo "")
            if [ -n "$OBJECTS" ]; then
                log "- Objects in bucket:"
                echo "$OBJECTS" | tr '\t' '\n' | while read -r obj; do
                    log "  - $obj"
                done
            fi

            # Ask for confirmation before cleanup
            read -p "Do you want to proceed with cleanup and delete all resources?
 (y/n): " confirm
            if [[ $confirm != [yY] && $confirm != [yY][eE][sS] ]]; then
                log "Cleanup aborted by user."
                return
            fi

            # Delete all versions of objects
            delete_all_versions "$BUCKET_NAME"

            # Delete the bucket
            log "Deleting bucket $BUCKET_NAME..."
            aws s3api delete-bucket --bucket "$BUCKET_NAME" || log "Warning: Failed
 to delete bucket"
        else
            log "Bucket $BUCKET_NAME does not exist, skipping cleanup"
        fi
    fi

    # Clean up local files
    log "Removing local files..."
```

```
    rm -f sample-file.txt sample-document.txt downloaded-sample-file.txt empty-
file.tmp

    log "Cleanup completed."
}

# Generate a random bucket name
generate_bucket_name() {
    local hex_id
    hex_id=$(openssl rand -hex 6)
    echo "demo-s3-bucket-$hex_id"
}

# Main script execution
main() {
    log "Starting Amazon S3 Getting Started Tutorial"

    # Generate a unique bucket name
    BUCKET_NAME=$(generate_bucket_name)
    log "Generated bucket name: $BUCKET_NAME"

    # Step 1: Create a bucket
    log "Step 1: Creating S3 bucket..."

    # Get the current region or default to us-east-1
    REGION=$(aws configure get region)
    REGION=${REGION:-us-east-1}
    log "Using region: $REGION"

    if [ "$REGION" = "us-east-1" ]; then
        aws s3api create-bucket --bucket "$BUCKET_NAME" || handle_error "Failed to
 create bucket"
    else
        aws s3api create-bucket \
            --bucket "$BUCKET_NAME" \
            --region "$REGION" \
            --create-bucket-configuration LocationConstraint="$REGION" ||
 handle_error "Failed to create bucket"
    fi
    log "Bucket created successfully"

    # Configure bucket settings
    log "Configuring bucket settings..."
```

```
    # Block public access (security best practice)
    log "Blocking public access..."
    aws s3api put-public-access-block \
        --bucket "$BUCKET_NAME" \
        --public-access-block-configuration
"BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=tr
|| handle_error "Failed to configure public access block"

    # Enable versioning
    log "Enabling versioning..."
    aws s3api put-bucket-versioning \
        --bucket "$BUCKET_NAME" \
        --versioning-configuration Status=Enabled || handle_error "Failed to enable
versioning"

    # Set default encryption
    log "Setting default encryption..."
    aws s3api put-bucket-encryption \
        --bucket "$BUCKET_NAME" \
        --server-side-encryption-configuration '{"Rules":
[{"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}]}' ||
handle_error "Failed to set encryption"

    # Step 2: Upload an object
    log "Step 2: Uploading objects to bucket..."

    # Create a sample file
    echo "This is a sample file for the S3 tutorial." > sample-file.txt

    # Upload the file
    aws s3api put-object \
        --bucket "$BUCKET_NAME" \
        --key "sample-file.txt" \
        --body "sample-file.txt" || handle_error "Failed to upload object"
    log "Object uploaded successfully"

    # Upload with metadata
    echo "This is a document with metadata." > sample-document.txt
    aws s3api put-object \
        --bucket "$BUCKET_NAME" \
        --key "documents/sample-document.txt" \
        --body "sample-document.txt" \
        --content-type "text/plain" \
```

```
        --metadata "author=AWSDocumentation,purpose=tutorial" || handle_error
"Failed to upload object with metadata"
    log "Object with metadata uploaded successfully"

    # Step 3: Download an object
    log "Step 3: Downloading object from bucket..."
    aws s3api get-object \
        --bucket "$BUCKET_NAME" \
        --key "sample-file.txt" \
        "downloaded-sample-file.txt" || handle_error "Failed to download object"
    log "Object downloaded successfully"

    # Check if an object exists
    log "Checking if object exists..."
    aws s3api head-object \
        --bucket "$BUCKET_NAME" \
        --key "sample-file.txt" || handle_error "Object does not exist"
    log "Object exists"

    # Step 4: Copy object to a folder
    log "Step 4: Copying object to a folder..."

    # Create a folder structure using a temporary empty file
    log "Creating folder structure..."
    touch empty-file.tmp
    aws s3api put-object \
        --bucket "$BUCKET_NAME" \
        --key "favorite-files/" \
        --body empty-file.tmp || handle_error "Failed to create folder"

    # Copy the object
    log "Copying object..."
    aws s3api copy-object \
        --bucket "$BUCKET_NAME" \
        --copy-source "$BUCKET_NAME/sample-file.txt" \
        --key "favorite-files/sample-file.txt" || handle_error "Failed to copy
object"
    log "Object copied successfully"

    # List objects in the bucket
    log "Listing all objects in the bucket..."
    aws s3api list-objects-v2 \
        --bucket "$BUCKET_NAME" \
        --query 'Contents[].Key' \
```

```
        --output table || handle_error "Failed to list objects"

    # List objects with a specific prefix
    log "Listing objects in the favorite-files folder..."
    aws s3api list-objects-v2 \
        --bucket "$BUCKET_NAME" \
        --prefix "favorite-files/" \
        --query 'Contents[].Key' \
        --output table || handle_error "Failed to list objects with prefix"

    # Add tags to the bucket
    log "Adding tags to the bucket..."
    aws s3api put-bucket-tagging \
        --bucket "$BUCKET_NAME" \
        --tagging 'TagSet=[{Key=Project,Value=S3Tutorial},
{Key=Environment,Value=Demo}]' || handle_error "Failed to add tags"
    log "Tags added successfully"

    log "Tutorial completed successfully!"
}

# Execute the main function
main
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的以下主题。
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - HeadObject
  - ListObjectVersions
  - ListObjectsV2
  - PutBucketEncryption
  - PutBucketTagging
  - PutBucketVersioning
  - PutObject

- PutPublicAccessBlock

使用 Amazon CLI 及 Bash 脚本的 Amazon STS 示例

以下代码示例演示如何通过将 Amazon Command Line Interface及 Bash 脚本与 Amazon STS 结合使用，来执行操作和实现常见场景。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

每个示例都包含一个指向完整源代码的链接，您可以从中找到有关如何在上下文中设置和运行代码的说明。

主题

- 操作

操作

`AssumeRole`

以下代码示例演示了如何使用 `AssumeRole`。

Amazon CLI 及 Bash 脚本

> **Note**
>
> 查看 GitHub，了解更多信息。查找完整示例，了解如何在 Amazon 代码示例存储库中进行设置和运行。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
```

```
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 AssumeRole。

Amazon CLI 中的安全性

Amazon 十分重视云安全性。作为 Amazon 客户，您将从专为满足大多数安全敏感型企业的要求而打造的数据中心和网络架构中受益。

安全性是 Amazon 和您的共同责任。 责任共担模式 将其描述为云的安全性和云中的安全性：

- 云的安全性 — Amazon 负责保护在 Amazon 云中运行 Amazon 服务的基础设施。Amazon 还向您提供可安全使用的服务。作为Amazon 合规性计划的一部分，第三方审计人员将定期测试和验证安全性的有效性。要了解适用于 Amazon Command Line Interface 的合规性计划，请参阅合规性计划范围内的 Amazon 服务。
- 云中的安全性——您的责任由您使用的 Amazon 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

此文档将帮助您了解如何在使用 Amazon Command Line Interface (Amazon CLI) 时应用责任共担模式。以下主题说明如何配置 Amazon CLI 以实现您的安全性和合规性目标。您还将了解如何使用 Amazon CLI 来帮助您监控和保护 Amazon 资源。

主题

- Amazon CLI 中的数据保护
- 身份和访问管理
- 此 Amazon 产品或服务的合规性验证
- 此 Amazon 产品或服务的故障恢复能力
- 此 Amazon 产品或服务的基础设施安全性
- 强制在 Amazon CLI 中实施最低 TLS 版本

Amazon CLI 中的数据保护

Amazon 责任共担模式适用于 Amazon Command Line Interface 中的数据保护。如该模式中所述，Amazon 负责保护运行所有 Amazon Web Services 云 的全球基础架构。您负责维护对托管在此基础架构上的内容的控制。您还负责您所使用的 Amazon Web Services 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。

出于数据保护目的，我们建议您保护 Amazon Web Services 账户 凭证并使用 Amazon IAM Identity Center 或 Amazon Identity and Access Management（IAM）设置单个用户。这样，每个用户只获得履行其工作职责所需的权限。我们还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与 Amazon 资源进行通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用 Amazon CloudTrail 设置 API 和用户活动日记账记录。有关使用 CloudTrail 跟踪来捕获 Amazon 活动的信息，请参阅《Amazon CloudTrail 用户指南》中的 Working with CloudTrail trails。
- 使用 Amazon 加密解决方案以及 Amazon Web Services 服务 中的所有默认安全控制。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果在通过命令行界面或 API 访问 Amazon 时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅美国联邦信息处理标准（FIPS）140-3。

我们强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括使用控制台、API、Amazon CLI 或 Amazon SDK 处理 Amazon CLI 或其他 Amazon Web Services 服务时。在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

数据加密

所有安全服务均具有一项重要功能，即信息在未处于活动使用状态时都会加密。

静态加密

除了代表用户与Amazon服务交互所需的凭证以外，Amazon CLI 本身不存储任何客户数据。

如果您使用 Amazon CLI 来调用 Amazon 服务，此服务将客户数据传输到本地电脑进行存储，则请参阅该服务的《用户指南》中的“安全与合规”章节，了解如何存储、保护和加密数据的相关信息。

传输中加密

预设情况下，从运行 Amazon CLI 和 Amazon 服务终端节点的客户端电脑上传输的所有数据，均通过使用 HTTPS/TLS 连接发送所有内容来加密。

您无需执行任何操作即可使用 HTTPS/TLS。除非您通过使用 `--no-verify-ssl` 命令行选项为单个命令显式禁用它，否则它始终处于启用状态。

身份和访问管理

Amazon Identity and Access Management（IAM）是一项 Amazon Web Services 服务，可以帮助管理员安全地控制对 Amazon 资源的访问。IAM 管理员控制谁可以通过身份验证（登录）和授权（具有权限）来使用 Amazon 资源。IAM 是一项无需额外费用即可使用的。Amazon Web Services 服务

主题

- 受众
- 使用身份进行身份验证
- 使用策略管理访问
- Amazon Web Services 服务如何与 IAM 协同工作
- 对 Amazon 身份和访问进行问题排查

受众

使用 Amazon Identity and Access Management（IAM）的方式因您可以在 Amazon 中执行的操作而异。

服务用户 – 如果使用 Amazon Web Services 服务来执行任务，则管理员会为您提供所需的凭证和权限。当您使用更多 Amazon 特征来完成工作时，您可能需要额外权限。了解如何管理访问权限有助于您向管理员请求适合的权限。如果您无法访问 Amazon 中的功能，请参阅对 Amazon 身份和访问进行问题排查或您所使用的 Amazon Web Services 服务的用户指南。

服务管理员：如果您在公司负责管理 Amazon 资源，则您可能具有 Amazon 的完全访问权限。您有责任确定您的服务用户应访问哪些 Amazon 特征和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要详细了解您的公司如何将 IAM 与 Amazon 结合使用，请参阅您所使用的 Amazon Web Services 服务的用户指南。

IAM 管理员：如果您是 IAM 管理员，您可能希望了解如何编写策略以管理对 Amazon 的访问权限的详细信息。要查看您可在 IAM 中使用的 Amazon 基于身份的策略示例，请参阅您所使用的 Amazon Web Services 服务的用户指南。

使用身份进行身份验证

身份验证是您使用身份凭证登录 Amazon 的方法。您必须作为 Amazon Web Services 账户根用户、IAM 用户或通过担任 IAM 角色进行身份验证。

对于编程访问，Amazon 提供了 SDK 和 CLI 来对请求进行加密签名。有关更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的 Amazon 签名版本 4。

Amazon Web Services 账户 根用户

当您创建 Amazon Web Services 账户 时，最初使用的是一个对所有 Amazon Web Services 服务和资源拥有完全访问权限的登录身份（称为 Amazon Web Services 账户根用户）。我们强烈建议不要使用根用户进行日常任务。有关需要根用户凭证的任务，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

联合身份

作为最佳实践，请要求人类用户必须使用带有身份提供者的联合身份验证才能使用临时凭证访问 Amazon Web Services 服务。

联合身份是来自企业目录、Web 身份提供者的用户，或 Amazon Directory Service 中的用户（这些用户使用来自身份源的凭证访问 Amazon Web Services 服务）。联合身份代入可提供临时凭证的角色。

IAM 用户和群组

IAM 用户是对某个人员或应用程序具有特定权限的一个身份。建议使用临时凭证，而非具有长期凭证的 IAM 用户。有关更多信息，请参阅《IAM 用户指南》中的要求人类用户使用带有身份提供商的联合身份验证才能使用临时凭证访问 Amazon。

IAM 组指定一组 IAM 用户，便于更轻松地对大量用户进行权限管理。有关更多信息，请参阅《IAM 用户指南》中的 IAM 用户使用案例。

IAM 角色

IAM 角色是具有特定权限的身份，可提供临时凭证。您可以通过从用户切换到 IAM 角色（控制台）或调用 Amazon CLI 或 Amazon API 操作来担任角色。有关更多信息，请参阅《IAM 用户指南》中的担任角色的方法。

IAM 角色对于联合用户访问、临时 IAM 用户权限、跨账户访问、跨服务访问以及在 Amazon EC2 上运行的应用程序非常有用。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

使用策略管理访问

您将创建策略并将其附加到 Amazon 身份或资源，以控制 Amazon 中的访问。策略在与身份或资源关联时定义权限。当主体发出请求时，Amazon 会评估这些策略。大多数策略在 Amazon 中存储为 JSON 文档。有关 JSON 策略文档的更多信息，请参阅《IAM 用户指南》中的 JSON 策略概述。

管理员使用策略，通过定义哪个主体可以在什么条件下对哪些资源执行哪些操作来指定谁有权访问什么。

默认情况下，用户和角色没有权限。IAM 管理员创建 IAM 策略并将其添加到角色中，然后用户可以担任这些角色。IAM 策略定义权限，与执行操作所用的方法无关。

基于身份的策略

基于身份的策略是您附加到身份（用户、组或角色）的 JSON 权限策略文档。这些策略控制身份可以执行什么操作、对哪些资源执行以及在什么条件下执行。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

基于身份的策略可以是内联策略（直接嵌入到单个身份中）或托管策略（附加到多个身份的独立策略）。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。您必须在基于资源的策略中指定主体。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用来自 IAM 的 Amazon 托管式策略。

访问控制列表（ACL）

访问控制列表（ACL）控制哪些主体（账户成员、用户或角色）有权访问资源。ACL 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3、Amazon WAF 和 Amazon VPC 是支持 ACL 的服务示例。要了解有关 ACL 的更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的访问控制列表（ACL）概览。

其他策略类型

Amazon 支持额外的策略类型，这些策略类型可以设置由更常用的策略类型授予的最大权限：

- 权限边界 – 设置基于身份的策略可以授予 IAM 实体的最大权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。
- 服务控制策略（SCP）– 指定 Amazon Organizations 中组织或组织单元的最大权限。有关更多信息，请参阅《Amazon Organizations 用户指南》中的服务控制策略。

- 资源控制策略（RCP）– 设置对账户中资源的最大可用权限。有关更多信息，请参阅《Amazon Organizations 用户指南》中的资源控制策略（RCP）。
- 会话策略 – 在为角色或联合用户创建临时会话时，作为参数传递的高级策略。有关更多信息，请参阅《IAM 用户指南》中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解 Amazon 如何确定在涉及多种策略类型时是否允许请求，请参阅《IAM 用户指南》中的策略评估逻辑。

Amazon Web Services 服务如何与 IAM 协同工作

要大致了解 Amazon Web Services 服务如何与大多数 IAM 功能结合使用，请参阅《IAM 用户指南》中的与 IAM 结合使用的 Amazon 服务。

要学习如何将特定的 Amazon Web Services 服务与 IAM 结合使用，请参阅相关服务的《用户指南》的安全部分。

对 Amazon 身份和访问进行问题排查

使用以下信息可帮助您诊断和修复在使用 Amazon 和 IAM 时可能遇到的常见问题。

主题

- 我无权在 Amazon 中执行操作
- 我无权执行 iam:PassRole
- 我希望允许我的 Amazon Web Services 账户以外的人访问我的 Amazon 资源

我无权在 Amazon 中执行操作

如果您收到错误提示，指明您无权执行某个操作，则必须更新策略以允许执行该操作。

当 `mateojackson` IAM 用户尝试使用控制台查看有关虚构 *`my-example-widget`* 资源的详细信息，但不拥有虚构 `awes:`*`GetWidget`* 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 awes:GetWidget on resource: my-example-widget
```

在此情况下，必须更新 mateojackson 用户的策略，以允许使用 awes:*GetWidget* 操作访问 *my-example-widget* 资源。

如果您需要帮助，请联系 Amazon 管理员。您的管理员是提供登录凭证的人。

我无权执行 iam:PassRole

如果您收到一个错误，表明您无权执行 iam:PassRole 操作，则必须更新策略以允许您将角色传递给。Amazon

有些 Amazon Web Services 服务 允许您将现有角色传递到该服务，而不是创建新服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 marymajor 的 IAM 用户尝试使用控制台在 Amazon 中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 iam:PassRole 操作。

如果您需要帮助，请联系 Amazon 管理员。您的管理员是提供登录凭证的人。

我希望允许我的 Amazon Web Services 账户以外的人访问我的 Amazon 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以代入角色。对于支持基于资源的策略或访问控制列表（ACL）的服务，您可以使用这些策略向人员授予对您的资源的访问权。

要了解更多信息，请参阅以下内容：

- 要了解 Amazon 是否支持这些特征，请参阅 Amazon Web Services 服务如何与 IAM 协同工作
- 要了解如何为您拥有的 Amazon Web Services 账户 中的资源提供访问权限，请参阅《IAM 用户指南》中的为您拥有的另一个 Amazon Web Services 账户 中的 IAM 用户提供访问权限。
- 要了解如何为第三方 Amazon Web Services 账户 提供您的资源的访问权限，请参阅《IAM 用户指南》中的为第三方拥有的 Amazon Web Services 账户 提供访问权限。
- 要了解如何通过身份联合验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（身份联合验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

此 Amazon 产品或服务的合规性验证

要了解某个 Amazon Web Services 服务 是否在特定合规性计划范围内，请参阅合规性计划范围内的 Amazon Web Services 服务，然后选择您感兴趣的合规性计划。有关常规信息，请参阅 、、Amazon Web Services 合规性计划。

您可以使用 Amazon Artifact 下载第三方审计报告。有关更多信息，请参阅在 Amazon Artifact 中下载报告。

您在使用 Amazon Web Services 服务 时的合规性责任由您的数据的敏感性、您公司的合规性目标以及适用的法律法规决定。有关您在使用 Amazon Web Services 服务时的合规责任的更多信息，请参阅 Amazon 安全性文档。

此 Amazon 产品或服务通过它支持的特定 Amazon Web Services（Amazon）服务遵循责任共担模式。有关 Amazon 服务安全性信息，请参阅 Amazon 服务安全性文档页面和合规性计划规定的 Amazon 合规性工作范围内的 Amazon 服务。

此 Amazon 产品或服务的故障恢复能力

Amazon 全球基础设施围绕 Amazon Web Services 区域和可用区而构建。

Amazon Web Services 区域 提供多个在物理上独立且隔离的可用区，这些可用区与延迟率低、吞吐量高且冗余性高的网络连接在一起。

利用可用区，您可以设计和运营在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错能力和可扩展性。

有关 Amazon 区域和可用区的更多信息，请参阅 Amazon全球基础设施。

此 Amazon 产品或服务通过它支持的特定 Amazon Web Services (Amazon) 服务遵循责任共担模式。有关 Amazon 服务安全性信息，请参阅 Amazon 服务安全性文档页面和合规性计划规定的 Amazon 合规性工作范围内的 Amazon 服务。

此 Amazon 产品或服务的基础设施安全性

此 Amazon 产品或服务使用托管式服务，因此受 Amazon 全球网络安全保护。有关 Amazon 安全服务以及 Amazon 如何保护基础设施的信息，请参阅 Amazon 云安全。要按照基础设施安全最佳实践设计您的 Amazon 环境，请参阅《安全性支柱 Amazon Well-Architected Framework》中的基础设施保护。

您可以使用 Amazon 发布的 API 调用通过网络访问此 Amazon 产品或服务。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 Amazon Security Token Service（Amazon STS）生成临时安全凭证来对请求进行签名。

此 Amazon 产品或服务通过它支持的特定 Amazon Web Services（Amazon）服务遵循责任共担模式。有关 Amazon 服务安全性信息，请参阅 Amazon 服务安全性文档页面和合规性计划规定的 Amazon 合规性工作范围内的 Amazon 服务。

强制在 Amazon CLI 中实施最低 TLS 版本

使用 Amazon Command Line Interface（Amazon CLI）时，传输层安全性协议（TLS）在保护 Amazon CLI 和 Amazon Web Services 服务之间的通信方面起着至关重要的作用。要提高与 Amazon 服务通信时的安全性，您应使用 TLS 1.2 或更高版本。

Amazon CLI 和 Amazon Web Services 服务可以使用提供加密、身份验证和数据完整性的 TLS 协议安全地交换数据。通过利用 TLS 协议，Amazon CLI 可确保您与 Amazon Web Services 服务的交互免遭未经授权的访问和数据泄露，从而增强 Amazon 生态系统的整体安全性。

Amazon 责任共担模式适用于 Amazon Command Line Interface 中的数据保护。如该模式中所述，Amazon 负责保护运行所有 Amazon Web Services 服务 的全球基础架构。您负责维护对托管在此基础架构上的内容的控制。您还负责您所使用的 Amazon Web Services 服务 的安全配置和管理任务。有关数据保护的更多信息，请参阅the section called “数据保护”。

要确保 Amazon CLI 版本 1 不使用 TLS 1.2 之前的 TLS 版本，您可能需要重新编译 OpenSSL 以强制实施此最低版本，然后重新编译 Python 以使用新构建的 OpenSSL。

主题

- 确定当前支持的协议
- 编译 OpenSSL 和 Python

确定当前支持的协议

首先，使用 OpenSSL 创建一个自签名证书，以用于测试服务器和 Python 开发工具包。

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out
 cert.pem -days 365
```

然后，使用 OpenSSL 启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

在新的终端窗口中，创建虚拟环境并安装 SDK for Python。

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

创建一个名为 `check.py` 的新 Python 脚本，该脚本使用此开发工具包的底层 HTTP 库。

```
$ import urllib3
URL = 'https://localhost:4433/'

http = urllib3.PoolManager(
    ca_certs='cert.pem',
    cert_reqs='CERT_REQUIRED',
)
r = http.request('GET', URL)
print(r.data.decode('utf-8'))
```

运行您的新脚本。

```
$ python check.py
```

这将显示有关所建立的连接的详细信息。在输出中搜索“协议：”。如果输出为“TLSv1.2”或更高版本，则开发工具包默认为 TLS v1.2 或更高版本。如果它是较早的版本，则需要重新编译 OpenSSL 并重新编译 Python。

但是，即使 Python 的安装默认为 TLS v1.2 或更高版本，如果服务器不支持 TLS v1.2 或更高版本，则 Python 仍可能重新协商到 TLS v1.2 之前的版本。要检查 Python 是否不会自动重新协商到较早版本，请使用以下命令重新启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

如果您使用的是较早版本的 OpenSSL，则可能没有可用的 `-no_tls_3` 标志。如果是这种情况，请删除该标志，因为您使用的 OpenSSL 版本不支持 TLS v1.3。然后，运行 Python 脚本。

```
$ python check.py
```

如果您正确安装了 Python 但未针对 TLS 1.2 之前的版本进行重新协商，则应收到 SSL 错误。

```
$ urllib3.exceptions.MaxRetryError: HTTPSConnectionPool(host='localhost',
 port=4433): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL:
 UNSUPPORTED_PROTOCOL] unsupported protocol (_ssl.c:1108)')))
```

如果您能够建立连接，则需要重新编译 OpenSSL 和 Python 以禁用对早于 TLS v1.2 的协议进行协商。

编译 OpenSSL 和 Python

为了确保开发工具包或 Amazon CLI 不对 TLS 1.2 之前的任何版本进行协商，您需要重新编译 OpenSSL 和 Python。要执行此操作，请复制以下内容以创建脚本并运行脚本。

```
#!/usr/bin/env bash
set -e

OPENSSL_VERSION="1.1.1d"
OPENSSL_PREFIX="/opt/openssl-with-min-tls1_2"
PYTHON_VERSION="3.8.1"
PYTHON_PREFIX="/opt/python-with-min-tls1_2"


curl -O "https://www.openssl.org/source/openssl-$OPENSSL_VERSION.tar.gz"
tar -xzf "openssl-$OPENSSL_VERSION.tar.gz"
cd openssl-$OPENSSL_VERSION
./config --prefix=$OPENSSL_PREFIX no-ssl3 no-tls1 no-tls1_1 no-shared
make > /dev/null
sudo make install_sw > /dev/null


cd /tmp
curl -O "https://www.python.org/ftp/python/$PYTHON_VERSION/Python-$PYTHON_VERSION.tgz"
tar -xzf "Python-$PYTHON_VERSION.tgz"
cd Python-$PYTHON_VERSION
./configure --prefix=$PYTHON_PREFIX --with-openssl=$OPENSSL_PREFIX --disable-shared > /
dev/null
```

```
make > /dev/null
sudo make install > /dev/null
```

这会编译具有静态链接的 OpenSSL 的 Python 版本，该版本不会自动协商早于 TLS 1.2 的任何版本。这也会在 /opt/openssl-with-min-tls1_2 目录中安装 OpenSSL，并在 /opt/python-with-min-tls1_2 目录中安装 Python。运行此脚本后，请确认安装新版本的 Python。

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

这应该打印出以下内容。

```
$ Python 3.8.1
```

要确认此新版本的 Python 不协商早于 TLS 1.2 的版本，请使用新安装的 Python 版本（即 /opt/python-with-min-tls1_2/bin/python3）重新运行 确定当前支持的协议 中的步骤。

Amazon CLI 用户指南文档历史记录

下表介绍了自 2019 年 1 月以来对 Amazon Command Line Interface 用户指南的重要补充。如需对此文档更新的通知，您可以订阅 RSS 源。

变更	说明	日期
更新了凭证和身份验证信息。	更新了凭证和身份验证方法的说明和示例。这包括更新相关的配置页面。为了适应此文档内容增加，相关凭证主题已移至新的身份验证和访问凭证部分。	2023 年 3 月 31 日
Amazon CLI V1 和 V2 的内容现已归入各自的指南中	为了清晰表达和易于使用，Amazon CLI 版本 1 和 Amazon CLI 版本 2 的内容现已归入各自的指南中。对于 Amazon CLI 版本 2，请参阅最新的 Amazon Command Line Interface 用户指南。	2021 年 11 月 2 日
已添加 Amazon CLI 别名信息	已添加 Amazon CLI 别名信息。别名是您可以在 Amazon Command Line Interface (Amazon CLI) 中创建的快捷方式，以缩短您经常使用的命令或脚本。	2021 年 3 月 11 日
更新筛选输出信息	更新了筛选条件的信息并移至自己的页面。	2021 年 2 月 1 日
Python 2.7、3.4 和 3.5 的弃用公告	Python 2.7 已于 2020 年 1 月 1 日被 Python Software Foundation 弃用。以后，使用 Amazon CLI 版本 1 的客户应过渡为使用 Python 3，至少使	2021 年 1 月 29 日

	用 Python 3.6。对于从 2021 年 7 月 19 日开始的 Amazon CLI 版本 1 的新版本，Python 2.7 支持已弃用。自 2021 年 2 月 1 日起，Python 3.4 和 3.5 将被弃用。	
添加了 Amazon S3 脚本示例	添加了 Amazon S3 生命周期脚本示例。	2020 年 10 月 15 日
添加了 Amazon EC2 脚本示例	添加了 Amazon EC2 实例类型脚本示例。	2020 年 10 月 15 日
添加了重试信息	在 Amazon CLI 中添加了重试功能和行为的重试页面。	2020 年 9 月 17 日
服务器端和客户端分页	更新了分页信息并集中在单个页面上。	2020 年 8 月 17 日
更新了 s3 命令页面	使用新示例和资源更新了高级别 s3 命令页面。	2020 年 7 月 30 日
更新的安装信息	将更新 Linux、macOS 和 Windows 的安装、更新和卸载信息。	2020 年 5 月 19 日
已更新以从 Amazon CLI 版本 1 中删除对 Python 2.6 和 3.3 的支持	自 2020 年 1 月 10 日起，Amazon CLI 版本 1 不再支持使用 Python 版本 2.6 或 3.3。您必须更新到更高版本的 Python 才能使用 Amazon CLI 版本 1.17 或更高版本。	2020 年 1 月 10 日
新的 MFA 部分	增加了一个新部分，其中介绍如何使用多重验证和角色访问 CLI。	2019 年 5 月 3 日
更新了“使用 CLI”部分	对使用说明和过程进行了重大改进和补充。	2019 年 3 月 7 日

更新了“安装 CLI”部分	Amazon CLI 安装说明和过程进行了重大改进和补充。	2019 年 3 月 7 日
更新了“配置 CLI”部分	Amazon CLI 配置说明和过程进行了重大改进和补充。	2019 年 3 月 7 日