亚马逊云科技

参考指南

Amazon 软件开发工具包和工具

Amazon 软件开发工具包和工具: 参考指南

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

Table of Contents

《 Amazon SDKs 和工具参考指南》中涵盖的内容

许多 SDKs 工具通过共享的设计规范或共享库共享一些共同的功能。

本指南包含有关以下内容的信息：

- 全局配置 Amazon SDKs 和工具— 如何使用共享`config`和`credentials`文件或环境变量来配置 Amazon SDKs 和工具。
- 使用和工具进行身份验证 Amazon SDKs 和访问— 确定您的代码或工具在使用开发 Amazon 时如何进行身份验证。 Amazon Web Services 服务
- Amazon SDKs 和工具设置参考 – 所有可用于身份验证和配置的标准化设置的参考。
- Amazon 通用运行时 (CRT) 库— 几乎所有 SDKs人都可以使用的共享 Amazon 公共运行时 (CRT) 库概述。
- Amazon SDK 和工具维护政策涵盖 Amazon 软件开发套件 (SDKs) 和工具（包括移动和物联网 (IoT)）SDKs的维护策略和版本控制及其底层依赖关系。

本 Amazon SDKs 和工具参考指南旨在成为适用于多种工具 SDKs 的信息库。除此处提供的任何信息外，还应使用您正在使用的 SDK 或工具的特定指南。以下是 SDK 和工具，其中包含本指南中的相关材料部分：

如果您正在使用：	本指南中与您相关的部分有：
• 任何 SDK 或工具	Amazon SDK 和工具维护政策
• Amazon Cloud9 • Amazon CDK • Amazon Toolkit for Azure DevOps • Amazon Toolkit for JetBrains • Amazon Toolkit for Visual Studio • Amazon Toolkit for Visual Studio Code • Amazon Serverless Application Model • Amazon CodeArtifact • Amazon CodeBuild	全局配置 Amazon SDKs 和工具 使用和工具进行身份验证 Amazon SDKs 和访问 Amazon SDK 和工具维护政策

如果您正在使用：	本指南中与您相关的部分有：
• Amazon CodeCatalyst • Amazon CodeCommit • Amazon CodeDeploy • Amazon CodePipeline	
• Amazon CLI • 适用于 C++ 的 Amazon SDK • 适用于 Go 的 Amazon SDK • 适用于 Java 的 Amazon SDK • 适用于 JavaScript 的 Amazon SDK • 适用于 Kotlin 的 Amazon SDK • 适用于 .NET 的 Amazon SDK • 适用于 PHP 的 Amazon SDK • 适用于 Python (Boto3) 的 Amazon SDK • 适用于 Ruby 的 Amazon SDK • 适用于 Rust 的 Amazon SDK • 适用于 Swift 的 Amazon SDK • Amazon Tools for Windows PowerShell	全局配置 Amazon SDKs 和工具 使用和工具进行身份验证 Amazon SDKs 和访问 Amazon SDKs 和工具设置参考 Amazon 通用运行时 (CRT) 库 Amazon SDK 和工具维护政策 Amazon SDKs 和 “工具” 版本生命周期

- 有关可帮助您开发应用程序的工具的概述 Amazon，请参阅构建工具 Amazon。
- 有关支持的信息，请参阅Amazon 知识中心。
- 有关 Amazon 术语，请参阅《Amazon Web Services 词汇表 参考资料》中的Amazon 词汇表。

全局配置 Amazon SDKs 和工具

使用 Amazon SDKs 和其他 Amazon 开发者工具（例如 Amazon Command Line Interface (Amazon CLI)），您可以与 Amazon 服务进行交互 APIs。但是，在尝试执行此操作之前，必须使用执行请求的操作所需的信息来配置 SDK 或工具。

这些信息包含以下各项：

- 识别 API 的调用方的凭证信息。凭证用于加密向 Amazon 服务器发出的请求。使用此信息 Amazon 确认您的身份，并可以检索与之相关的权限策略。然后，它可以确定允许您执行哪些操作。
- 其他配置详细信息，用于告知 Amazon CLI 或 SDK 如何处理请求、将请求发送到何处（发送到哪个 Amazon 服务端点）以及如何解释或显示响应。

每个 SDK 或工具都支持多个来源，您可以使用这些来源来提供所需的凭证和配置信息。有些来源是 SDK 或工具所独有的，您必须参阅该工具或 SDK 的文档，详细了解如何使用该方法。

但是， Amazon SDKs 和工具支持代码本身以外的主要来源的常用设置。本节将介绍以下主题：

主题

使用共享`config`和`credentials`文件进行全局配置 Amazon SDKs 和工具

共享 Amazon `config`和`credentials`文件是为 Amazon SDK 或工具指定身份验证和配置的最常用方式。

共享的 `config` 和 `credentials` 文件包含一组配置文件。配置文件是一组按键值对组成的配置设置 Amazon SDKs，由 Amazon Command Line Interface (Amazon CLI) 和其他工具使用。将配置值附加到配置文件中，以便配置 SDK/tool 何时使用该配置文件的某些方面。这些文件是 “共享” 的，因为这些值对任何应用程序、进程或 SDKs 对用户的本地环境都有影响。

共享config文件和credentials文件都是纯文本文件，仅包含 ASCII 字符（UTF-8 编码）。它们采用通常称为 INI 文件的形式。

配置文件

共享文件 config 和 credentials 中的设置与特定的配置文件相关联。可以在该文件中定义多个配置文件，从而创建将在不同开发环境中应用的不同设置配置。

如果未指定特定的命名 [default] 配置文件，则该配置文件包含由 SDK 或工具操作使用的值。您也可以创建单独的配置文件，您可以按名称明确引用这些配置文件。每个配置文件都可以根据应用程序和场景的需要使用不同的设置和值。

> **Note**
>
> [default]只是一个未命名的配置文件。此配置文件之所以命名为default，是因为如果用户未指定配置文件，则它是 SDK 使用的默认配置文件。它不为其他配置文件提供接替的默认值。如果您在 [default] 配置文件中设置了某些值，但未在命名配置文件中相应设置，则在使用命名配置文件时不会设置该值。

设置命名配置文件

[default] 配置文件和多个命名配置文件可以在同一个文件中共存。使用以下设置来选择 SDK 或工具在运行代码时将使用配置文件中的具体设置。此外还可以在代码中或使用 Amazon CLI时按命令选择配置文件。

通过设置以下配置之一来配置此功能：

AWS_PROFILE-环境变量

当此环境变量设置为命名配置文件或 “默认” 时，所有 SDK 代码和 Amazon CLI 命令都使用该配置文件中的设置。

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_PROFILE="my_default_profile_name";
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_PROFILE "my_default_profile_name"
```

`aws.profile`：JVM 系统属性

对于 JVM 上适用于 Kotlin 的 SDK 和适用于 Java 的 SDK 2.x，您可以设置 `aws.profile` 系统属性。当 SDK 创建服务客户端时，将会使用命名配置文中件的设置，但该设置在代码中被覆盖的情况除外。适用于 Java 的 SDK 1.x 不支持此系统属性。

> **Note**
>
> 如果应用程序位于运行多个应用程序的服务器上，我们建议您始终使用命名配置文件而不是默认配置文件。环境中的任何 Amazon 应用程序都会自动获取默认配置文件，并在它们之间共享。因此，如果其他人更新了其应用程序的默认配置文件，则可能会在无意中影响其他应用程序。为防止出现这种情况，请在共享的 `config` 文件中定义一个命名配置文件，然后在代码中设置该命名配置文件，从而在应用程序中使用该命名配置文件。如果您知道命名配置文件的作用域仅影响您的应用程序，则可以使用环境变量或 JVM 系统属性来设置该命名配置文件。

配置文件的格式

`config` 文件将归入各个节中。节是一个命名的设置集合，它一直持续到遇到另一个节定义行为止。

`config` 文件是使用以下格式的纯文本文件：

- 节中的所有条目均采用 `setting-name=value` 的一般形式。
- 可以通过以井号字符 (#) 开头来注释掉行。

节类型

节定义是将名称应用于设置集合的行。节定义行以方括号 ([]) 开头和结尾。方括号内有一个节类型标识符和该节的自定义名称。可以使用字母、数字、连字符 (-) 和下划线 (_)，但不能使用空格。

节类型：**`default`**

节定义行示例：`[default]`

`[default]` 是唯一一个不需要 `profile` 节标识符的配置文件。

下面的示例介绍一个有 `[default]` 配置文件的基本 `config` 文件。它设置了`region`设置。该行之后的所有设置都包含该配置文件中，直到遇到另一个节定义为止。

```
[default]
#Full line comment, this text is ignored.
region = us-east-2
```

节类型：**profile**

节定义行示例：[profile *dev*]

profile 节定义行是一个您可以应用到不同开发场景的命名配置分组。要更好地了解命名配置文件，请参阅前面关于配置文件的部分。

以下示例展示了一个带有 profile 节定义行和命名配置文件 foo 的 config 文件。该行之后的所有设置都包含该命名配置文件中，直到遇到另一个节定义为止。

```
[profile foo]
...settings...
```

某些设置有自己的嵌套子设置组，例如以下示例中的 s3 设置和子设置。通过缩进一个或多个空格将子设置与组相关联。

```
[profile test]
region = us-west-2
s3 =
    max_concurrent_requests=10
    max_queue_size=1000
```

节类型：**sso-session**

节定义行示例：[sso-session *my-sso*]

sso-session部分定义行命名了一组设置，您使用这些设置来配置配置文件以解析 Amazon 凭据 Amazon IAM Identity Center。有关配置单点登录身份验证的更多信息，请参阅 使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证。配置文件通过键值对链接到 sso-session 节，其中 sso-session 是密钥，您的 sso-session 节名称是值，例如 sso-session = <name-of-sso-session-section>。

以下示例配置了一个配置文件，该配置文件将使用 “my-sso” 中的令牌获取 “111122223333SampleRole” 账户中 “” IAM 角色的短期 Amazon 证书。在profile节中，使用 sso-session 密钥按名称引用 “my-sso” sso-session 节。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

节类型：**services**

节定义行示例：[services *dev*]

> **Note**
>
> 该services部分支持服务特定的端点自定义，并且仅在包含此功能的工具中 SDKs 可用。要查看此功能是否适用于您的 SDK，请参阅 Support Amazon SDKs by 和工具 以了解服务特定的端点。

services部分定义行命名了一组为 Amazon Web Services 服务 请求配置自定义终端节点的设置。配置文件通过键值对链接到 services 节，其中 services 是密钥，您的 services 节名称是值，例如 services = <name-of-services-section>。

该services部分进一步按<SERVICE> = 行分成小节，其中<SERVICE>是标 Amazon Web Services 服务 识键。标 Amazon Web Services 服务 识符基于 API 模型，将所有空格serviceId替换为下划线，并将所有字母小写。有关 services 节中要使用的所有服务标识符密钥的列表，请参阅特定于服务的端点的标识符。服务标识符密钥后面是嵌套的设置，每个设置单独成行，缩进两个空格。

以下示例使用services定义来配置端点，使其仅用于向 Amazon DynamoDB 服务发出的请求。在"local-dynamodb" services节中，使用services密钥按名称引用profile节。标 Amazon Web Services 服务 识符密钥是dynamodb。 Amazon DynamoDB 服务小节从线路开始dynamodb = 。后面紧跟的任何缩进行都包含在该小节中，并适用于该服务。

```
[profile dev]
services = local-dynamodb

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

有关自定义端点配置的更多信息，请参阅特定于服务的端点。

凭证文件的格式

credentials文件的规则通常与config文件的规则相同，唯一的不同是配置文件部分不是以单词profile开头。在方括号之间仅使用配置文件名称本身。以下示例展示了一个带有命名配置文件节 foo 的 credentials 文件。

```
[foo]
...credential settings...
```

只有以下被视为“密钥”或敏感的设置才能存储在 credentials 文件中：aws_access_key_id、aws_secret_access_key 和aws_session_token。尽管也可以将这些设置放在共享的 config 文件中，但我们建议您将这些敏感的值保存在单独的 credentials 文件中。这样，如有必要，您可以为每个文件提供单独的权限。

下面的示例介绍一个有 [default] 配置文件的基本 credentials 文件。它会设置 aws_access_key_id、aws_secret_access_key 和 aws_session_token 全局设置。

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRI
```

无论您在 credentials 文件中使用命名配置文件还是“default”，此处的任何设置都将与 config 文件中具有相同置配文件名称的任何设置合并。如果两个文件中都有共享相同名称的配置文件的凭证，则凭证文件中的密钥优先。

查找和更改共享credentials文件config Amazon SDKs 和工具的位置

共享credentials文件 Amazon config和文件是纯文本文件，其中包含 Amazon SDKs 和工具的配置信息。这些文件位于您的本地环境中，由 SDK 代码或在该环境中运行的 Amazon CLI 命令自动使用。例如，在您自己的计算机上或在 Amazon Elastic Compute Cloud 实例上开发时。

当 SDK 或工具运行时，将会检查这些文件并加载所有可用的配置设置。如果这些文件尚不存在，则 SDK 或工具会自动创建一个基础文件。

默认情况下，这些文件位于您的 home 或用户文件夹下名为 .aws 的文件夹中。

操作系统	文件的默认位置和名称

Linux 和 macOS	`~/.aws/config`
	
	`~/.aws/credentials`

Windows	`%USERPROFILE%\.aws\config`
	
	`%USERPROFILE%\.aws\credentials`

主目录解析

~ 仅在下列情况下才用于主目录解析：

- 作为路径的开始
- 其后紧接 / 或平台特定的分隔符。在 Windows 上，~/ 和 ~\ 都会解析到主目录。

在确定主目录时，系统会检查以下变量：

- （所有平台）HOME 环境变量
- （Windows 平台）USERPROFILE 环境变量
- （Windows 平台）HOMEDRIVE 和 HOMEPATH 环境变量的串连（$HOMEDRIVE$HOMEPATH）
- （可选，根据 SDK 或工具）特定于 SDK 或工具的主路径解析函数或变量

如有可能，如果在路径开头指定了用户的主目录（例如，~username/），则会将其解析到请求的用户名的起始目录（例如，/home/username/.aws/config）。

更改这些文件的默认位置

您可以使用以下任一方法来覆盖 SDK 或工具加载这些文件的位置。

使用环境变量

可以设置以下环境变量，将这些文件的位置或名称从默认值更改为自定义值：

- config 文件环境变量：**AWS_CONFIG_FILE**
- credentials 文件环境变量：**AWS_SHARED_CREDENTIALS_FILE**

Linux/macOS

您可以通过在 Linux 或 macOS 上运行以下导出命令来指定备用位置。

```
$ export AWS_CONFIG_FILE=/some/file/path/on/the/system/config-file-name
$ export AWS_SHARED_CREDENTIALS_FILE=/some/other/file/path/on/the/system/
credentials-file-name
```

Windows

您可以通过在 Windows 上运行以下setx命令来指定备用位置。

```
C:\> setx AWS_CONFIG_FILE c:\some\file\path\on\the\system\config-file-name
C:\> setx AWS_SHARED_CREDENTIALS_FILE c:\some\other\file\path\on\the\system
\credentials-file-name
```

有关使用环境变量配置系统的更多信息，请参阅使用环境变量进行全局配置 Amazon SDKs 和工具。

使用 JVM 系统属性

对于在 JVM 上运行的适用于 Kotlin 的 SDK 以及适用于 Java 的 SDK 2.x，您可以通过设置以下 JVM 系统属性，将这些文件的位置或名称从默认值更改为自定义值：

- `config` 文件 JVM 系统属性：**`aws.configFile`**
- `credentials` 文件环境变量：**`aws.sharedCredentialsFile`**

有关如何设置 JVM 系统属性的说明，请参阅the section called “如何设置 JVM 系统属性”。适用于 Java 的 SDK 1.x 不支持这些系统属性。

使用环境变量进行全局配置 Amazon SDKs 和工具

环境变量提供了另一种在使用和工具时指定配置选项 Amazon SDKs 和凭据的方法。环境变量在编写脚本或将某个命名配置文件临时设置为默认配置文件时非常实用。有关大多数支持的环境变量的列表 SDKs，请参阅环境变量列表。

选项的优先顺序

- 如果您使用环境变量来指定设置，则该设置将覆盖从共享 Amazon `config`和`credentials`文件中的配置文件加载的任何值。

- 如果您在 Amazon CLI 命令行中使用参数来指定设置，则该设置将覆盖配置文件中相应环境变量或配置文件中的任何值。

如何设置环境变量

下面的示例介绍您如何可以为默认用户配置环境变量。

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export
 AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
$ export AWS_REGION=us-west-2
```

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

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> setx
 AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
C:\> setx AWS_REGION us-west-2
```

使用 `set` 设置环境变量会更改使用的值，直到当前命令提示符会话结束，或者直到您将该变量设置为其他值。使用 `setx` 设置环境变量会更改当前命令提示符会话和运行该命令后创建的所有命令提示符会话中使用的值。它不 影响在运行该命令时已经运行的其他命令 shell。

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
PS C:
\> $Env:AWS_SESSION_TOKEN="AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Ol
PS C:\> $Env:AWS_REGION="us-west-2"
```

如果您在 PowerShell 提示符处设置环境变量（如前面的示例所示），则它只会在当前会话的持续时间内保存该值。要使环境变量设置在所有会话 PowerShell 和命令提示符会话中保持不变，请使

用控制面板中的系统应用程序将其存储。或者，您可以通过将变量添加到您的 PowerShell 个人资料中来为所有将来的 PowerShell 会话设置该变量。有关存储环境变量或跨会话保存环境变量的更多信息，请参阅PowerShell 文档。

无服务器环境变量设置

如果您使用无服务器架构进行开发，则还有其他设置环境变量的选项。根据您的容器，您可以对在这些容器中运行的代码使用不同的策略来查看和访问环境变量，这与非云环境类似。

例如，使用 Amazon Lambda，您可以直接设置环境变量。有关详细信息，请参阅《Amazon Lambda 开发人员指南》中的使用 Amazon Lambda 环境变量。

在无服务器框架中，通常可在环境设置下的提供者密钥下的 `serverless.yml` 文件中设置 SDK 环境变量。有关该 `serverless.yml` 文件的信息，请参阅无服务器框架文档中的 常规功能设置 。

无论您使用哪种机制来设置容器环境变量，都有一些变量由容器保留，例如在 定义的运行时系统环境变量 中为 Lambda 记录的变量。请务必查阅所用容器的官方文档，以确定如何处理环境变量以及是否存在任何限制。

使用 JVM 系统属性进行全局配置和 适用于 Java 的 Amazon SDK 适用于 Kotlin 的 Amazon SDK

JVM 系统属性提供了另一种方法来指定在 JVM 上运行 SDKs 的配置选项和凭据，例如 适用于 Java 的 Amazon SDK 和。 适用于 Kotlin 的 Amazon SDK有关支持的 JVM 系统属性的列表 SDKs，请参阅设置参考。

选项的优先顺序

- 如果您使用 JVM 系统属性来指定某个设置，则会覆盖环境变量中找到的或从共享的 AWS `config` 和 `credentials` 文件中加载的任何值。
- 如果您使用环境变量指定某个设置，则会覆盖从共享的 AWS `config` 和 `credentials` 文件中加载的任何值。

如何设置 JVM 系统属性

您可以通过多种方式设置 JVM 系统属性。

通过命令行

使用 -D 开关调用 java 命令时通过命令行设置 JVM 系统属性。除非在代码中显式覆盖该值，否则以下命令将为所有服务客户端进行 Amazon Web Services 区域 全局配置。

```
java -Daws.region=us-east-1 -jar <your_application.jar> <other_arguments>
```

如果需要设置多个 JVM 系统属性，请多次指定 -D 开关。

使用环境变量

如果您无法访问命令行调用 JVM 来运行应用程序，则可以使用 JAVA_TOOL_OPTIONS 环境变量来配置命令行选项。这种方法对于在 Java 运行时上运行 Amazon Lambda 函数或在嵌入式 JVM 中运行代码等情下非常实用。

除非您在代码中明确覆盖该值，否则以下示例将为所有服务客户端进行 Amazon Web Services 区域 全局配置。

Linux, macOS, or Unix

```
$ export JAVA_TOOL_OPTIONS="-Daws.region=us-east-1"
```

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

Windows Command Prompt

```
C:\> setx JAVA_TOOL_OPTIONS -Daws.region=us-east-1
```

使用 set 设置环境变量会更改使用的值，直到当前命令提示符会话结束，或者直到您将该变量设置为其他值。使用 setx 设置环境变量会更改当前命令提示符会话和运行该命令后创建的所有命令提示符会话中使用的值。它不 影响在运行该命令时已经运行的其他命令 shell。

在运行时上

您也可以使用以下示例所示的 System.setProperty 方法，通过代码在运行时上设置 JVM 系统属性。

```
System.setProperty("aws.region", "us-east-1");
```

> **Important**
>
> 所有 JVM 系统属性都应在初始化 SDK 服务客户端之前设置，否则服务客户端可能会使用其他值。

使用和工具进行身份验证 Amazon SDKs 和访问

在开发 S Amazon DK 应用程序或使用要使用的 Amazon 工具时 Amazon Web Services 服务，必须确定您的代码或工具的身份验证 Amazon方式。您可以通过不同的方式配置对 Amazon 资源的编程访问权限，具体取决于代码运行的环境和可用的 Amazon 访问权限。

以下选项是凭证提供者链的一部分。这意味着，通过相应地配置您的共享`credentials`文件 Amazon `config`和文件，您的 Amazon SDK 或工具将自动发现并使用该身份验证方法。

选择应用程序代码身份验证方法

选择一种方法来验证您的应用程序发 Amazon 出的呼叫。

你是否在 Amazon Web Services 服务 （ 例如亚马逊 EC2、Lambda、Amazon ECS、Amazon EKS 等 CodeBuild ）中运行代码？

如果您的代码在上运行 Amazon，则凭据可以自动提供给您的应用程序。例如，假设您的应用程序在 Amazon Elastic Compute Cloud 上托管，并且有一个该资源关联的 IAM 角色，则您的应用程序将会自动获取凭证。同样，如果您使用 Amazon ECS 或 Amazon EKS 容器，则容器内运行的代码可以该 SDK 的凭证提供者链自动获取为该 IAM 角色设置的凭证。

要在 Amazon Elastic Compute Cloud 实例中运行代码？

使用 IAM 角色对部署到 Amazon 的应用程序进行身份验证 EC2 – 使用 IAM 角色在 Amazon EC2 实例上安全地运行您的应用程序。

你的代码在 Amazon Lambda 函数里吗？

当您创建 Lambda 函数时，Lambda 会创建具有最低权限的执行角色。然后， Amazon 软件开发工具包或工具在运行时通过 Lambda 执行环境自动使用附加到 Lambda 的 IAM 角色。

您的代码是否在亚马逊弹性容器服务（亚马逊 EC2 上或 Amazon Fargate 亚马逊 ECS 上）？

使用任务的 IAM 角色。您必须创建一个任务角色并在 Amazon ECS 任务定义中指定该角色。然后，Amazon SDK 或工具会自动使用在运行时通过 Amazon ECS 元数据分配给任务的 IAM 角色。

要在 Amazon Elastic Kubernetes Service 中运行代码？

我们建议您使用 Amazon EKS 容器组身份。

注意：如果您认为服务账户的 IAM 角色（IRSA）可能更适合您的独特需求，请参阅《Amazon EKS 用户指南》中的比较 EKS 容器组身份和 IRSA。

你的代码在运行吗 Amazon CodeBuild

有关信息，请参阅使用基于身份的策略。 CodeBuild

要在其他 Amazon Web Services 服务中运行代码？

请参阅相关 Amazon Web Services 服务的专门指南。当你在上运行代码时 Amazon，SDK 凭证提供程序链可以自动为你获取和刷新凭证。

要创建移动应用程序或基于客户端的 Web 应用程序？

如果您正在创建需要访问的移动应用程序或基于客户端的 Web 应用程序 Amazon，请构建您的应用程序，使其能够使用 Web 联合身份验证动态请求临时 Amazon 安全证书。

利用 Web 联合身份验证，您不需要创建自定义登录代码或管理自己的用户身份。相反，应用程序用户可以使用知名的外部身份提供者（IdP）（例如，Login with Amazon、Facebook、Google 或任何其他 OpenID Connect (OIDC) 兼容的 IdP）登录。他们可以接收身份验证令牌，然后将该令牌交换为该映射中的临时安全证书 Amazon ，该证书到有权使用您的资源的 IAM 角色 Amazon Web Services 账户。

要了解如何为您的 SDK 或工具进行配置，请参阅 假设角色使用网络身份或 OpenID Connect 进行身份验证和工具 Amazon SDKs。

有关移动应用程序，请考虑使用 Amazon Cognito。Amazon Cognito 充当身份凭证代理程序并为您完成许多联合身份验证工作。有关更多信息，请参阅 IAM 用户指南中的将 Amazon Cognito 用于移动应用程序。

要在本地开发和运行代码？

我们建议使用控制台凭据进行身份验证 Amazon SDKs 和使用工具。

在基于浏览器的快速身份验证流程之后，会 Amazon 自动生成适用于本地开发工具（如 CL Amazon I 和）的临时证书。 Amazon Tools for PowerShell Amazon SDKs

如果您使用身份中心进行 Amazon 账户访问

如果您已经有权访问 and/or 需要管理员工访问权限的 Amazon 账户，请使用 IAM Identity Center 对 Amazon SDK 和工具进行身份验证。作为安全最佳实践，我们建议 Amazon Organizations 与 IAM Identity Center 配合使用来管理所有 Amazon 账户的访问权限。您可以在 IAM 身份中心创建用户，使用 Microsoft Active Directory，使用 SAML 2.0 身份提供商 (IdP)，或者将您的 Id Amazon P 单独联合

到账户。要查看您所在的地区是否支持 IAM 身份中心，请参阅 使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证 Amazon Web Services 一般参考中的 IAM 身份中心终端节点和配额。

如果您正在寻找其他身份验证方式

创建一个权限最低的 IAM 用户，该用户有权`sts:AssumeRole`进入您的目标角色。然后，使用为该用户`source_profile`设置的设置，将您的个人资料配置为扮演角色。

您也可以通过环境变量或共享凭证文件使用临时 IAM Amazon 证书。请参阅使用短期凭证进行身份验证 Amazon SDKs 和工具。

注意：仅在沙盒或学习环境中，您可以考虑使用长期凭据进行身份验证 Amazon SDKs 和使用工具。

要在本地或者混合/按需 VM（例如从 Amazon S3 读取或写入 Amazon S3 的服务器，或者部署到云端的 Jenkins）中运行代码？

要使用 X.509 客户端证书？

是：请参阅使用 IAM Anywhere 角色进行身份验证 Amazon SDKs 和工具。您可以使用 IAM Roles Anywhere 在 IAM 中为在外部运行的服务器、容器和应用程序等工作负载获取临时安全证书 Amazon。要使用 IAM Roles Anywhere，您的工作负载必须使用 X.509 证书。

环境能否安全地连接到联合身份提供商（例如 Microsoft Entra 或 Okta）以请求临时证书？ Amazon

能：使用进程凭证提供者

使用进程凭证提供者在运行时自动检索凭证。这些系统可能会使用辅助工具或插件来获取凭证，并且可能使用 `sts:AssumeRole` 在幕后代入 IAM 角色。

否：使用通过注入的临时证书 Amazon Secrets Manager

使用通过注入的临时证书 Amazon Secrets Manager。有关获取短期访问密钥的选项，请参阅《IAM 用户指南》中的请求临时安全凭证。有关存储此类临时凭证的选项，请参阅 Amazon 访问密钥。

您可以使用这些凭证安全地从 Secrets Manager（其中可能存储了您生的产密钥或基于角色的长期凭证）检索更广泛的应用程序权限。

你使用的第三方工具不在里面 Amazon吗？

请参阅第三方提供者编写的文档，了解有关获取凭证的最佳指导。

如果第三方提供者没有提供文档，您能否安全地注入临时凭证？

是：使用环境变量和临时 Amazon STS 证书。

不能：使用存储在加密的密钥管理器中的静态访问密钥（最后手段）。

身份验证方法

在 Amazon 环境中运行的代码的身份验证方法

如果您的代码在上运行 Amazon，则凭据可以自动提供给您的应用程序。例如，假设您的应用程序在 Amazon Elastic Compute Cloud 上托管，并且有一个该资源关联的 IAM 角色，则您的应用程序将会自动获取凭证。同样，如果您使用 Amazon ECS 或 Amazon EKS 容器，则容器内运行的代码可以该 SDK 的凭证提供者链自动获取为该 IAM 角色设置的凭证。

- 使用 IAM 角色对部署到 Amazon 的应用程序进行身份验证 EC2 – 使用 IAM 角色在 Amazon EC2 实例上安全地运行您的应用程序。
- 您可以通过以下方式 Amazon 使用 IAM 身份中心进行编程交互：
  - Amazon CloudShell用于从控制台运行 Amazon CLI 命令。
  - 要尝试为软件开发团队提供基于云的协作空间，可以考虑使用 Amazon CodeCatalyst。

通过基于 Web 的身份提供者进行身份验证 - 移动或基于客户端的 Web 应用程序

如果您正在创建需要访问的移动应用程序或基于客户端的 Web 应用程序 Amazon，请构建您的应用程序，使其能够使用 Web 联合身份验证动态请求临时 Amazon 安全证书。

利用 Web 联合身份验证，您不需要创建自定义登录代码或管理自己的用户身份。相反，应用程序用户可以使用知名的外部身份提供者（IdP）（例如，Login with Amazon、Facebook、Google 或任何其他 OpenID Connect (OIDC) 兼容的 IdP）登录。他们可以接收身份验证令牌，然后将该令牌交换为该映射中的临时安全证书 Amazon ，该证书到有权使用您的资源的 IAM 角色 Amazon Web Services 账户。

要了解如何为您的 SDK 或工具进行配置，请参阅 假设角色使用网络身份或 OpenID Connect 进行身份验证和工具 Amazon SDKs。

有关移动应用程序，请考虑使用 Amazon Cognito。Amazon Cognito 充当身份凭证代理程序并为您完成许多联合身份验证工作。有关更多信息，请参阅 IAM 用户指南中的将 Amazon Cognito 用于移动应用程序。

本地（不在 Amazon中）运行的代码的身份验证方式

- 使用控制台凭据进行身份验证 Amazon SDKs 和使用工具— 此功能可与 Amazon 命令行界面和工具配合使用， PowerShell 并为您提供可刷新的凭据，这些凭据适用于本地开发工具（例如 Amazon CLI、 PowerShell 和 Amazon的工具）。

- 使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证— 作为安全最佳实践，我们建议 Amazon Organizations 与 IAM Identity Center 配合使用来管理所有人的访问权限 Amazon Web Services 账户。你可以在中创建用户 Amazon IAM Identity Center，使用 Microsoft Active Directory，使用 SAML 2.0 身份提供商 (IdP)，或者将你的 IdP 单独联合到其中。 Amazon Web Services 账户要查看您的地区是否支持 IAM Identity Center，请参阅 Amazon Web Services 一般参考 中的 Amazon IAM Identity Center 终端点和配额。
- 使用 IAM Anywhere 角色进行身份验证 Amazon SDKs 和工具— 您可以使用 IAM Roles Anywhere 在 IAM 中为在外部运行的服务器、容器和应用程序等工作负载获取临时安全证书 Amazon。要使用 IAM Roles Anywhere，您的工作负载必须使用 X.509 证书。
- 假设一个拥有身份验证 Amazon 凭证 Amazon SDKs 和工具的角色— 您可以扮演 IAM 角色来临时访问原本可能无法访问的 Amazon 资源。
- 使用 Amazon 访问密钥进行身份验证 Amazon SDKs 和工具— 其他可能不太方便或可能增加 Amazon 资源安全风险的选项。

有关访问管理的更多信息

I AM 用户指南包含以下有关安全控制 Amazon 资源访问的信息：

- IAM 身份（用户、用户组和角色）— 了解中身份的基础知识 Amazon。
- IAM 中的安全最佳实践 – 根据责任共担模式开发 Amazon 应用程序时应遵循的安全建议。

Amazon Web Services 一般参考 具有以下基础知识：

- 了解并获取您的 Amazon 凭证 – 控制台和以编程方式访问的访问密钥选项和管理实践。

使用 IAM Identity Center 可信身份传播（TIP）插件来访问 Amazon Web Services 服务

- 使用 TIP 插件进行访问 Amazon Web Services 服务— 如果您正在为 Amazon Q Business 或其他支持可信身份传播的服务创建应用程序，并且正在使用 适用于 Java 的 Amazon SDK 或 适用于 JavaScript 的 Amazon SDK，则可以使用 TIP 插件来获得简化的授权体验。

Amazon 构建者 ID

任何 Amazon Web Services 账户 你可能已经拥有或想要创作的 Amazon 构建者 ID 补充。虽然 Amazon Web Services 账户 充当你创建的 Amazon 资源的容器并为这些资源提供安全边界，但你的

Amazon 构建者 ID 代表你是一个个体。您可以使用登录 Amazon 构建者 ID 以访问开发者工具和服务，例如 Amazon Q 和 Amazon CodeCatalyst。

- 使用 Amazon 构建者 IDAmazon 登录 用户指南登录 — 了解如何创建和使用， Amazon 构建者 ID 并了解生成器 ID 提供的内容。
- CodeCatalyst概念- Amazon 构建者 ID 在 Amazon CodeCatalyst 用户指南中-了解如何 CodeCatalyst 使用 Amazon 构建者 ID.

使用控制台凭据进行身份验证 Amazon SDKs 和使用工具

在本地环境或其他非Amazon 计算服务环境中开发 Amazon 应用程序时，建议使用控制台 Amazon 凭据提供凭证。如果您正在使用诸如亚马逊弹性计算云 (Amazon EC2) Amazon CloudShell 或 Amazon EC2之类的 Amazon 资源进行开发，我们建议改为从该服务获取证书。

您也可以通过 IAM 身份中心进行身份验证使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证。此选项是组织管理员工访问权限的常用方法，需要启用 Identity Center。

如何工作？

使用控制台凭据登录进行 Amazon 本地开发允许您使用现有的 Amazon 管理控制台登录凭据以编程方式访问 Amazon 服务。在基于浏览器的身份验证流程之后， Amazon 生成适用于 PowerShell 本地开发工具（例如 CL Amazon I、和的工具）的临时证书。 Amazon SDKs此功能简化了配置和管理 Amazon CLI 凭证的过程，尤其是在您更喜欢交互式身份验证而不是管理长期访问密钥的情况下。

通过此流程，您可以使用在初始账户设置期间创建的根证书、IAM 用户或身份提供商提供的联合身份进行身份验证。

如果您 SDKs 用于开发，SDK 客户端将通过使用临时证书Amazon SDKs 和 Tools 标准化凭证提供商。您也可以配置登录凭证提供商。

Amazon CLI 和工具都支持通过 login 命令进行身份验证，用于 PowerShell：

- 使用控制台凭据登录进行 Amazon 本地开发
- 使用 Amazon Tools for PowerShell 用户指南中的控制台凭据登录

使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证

Amazon IAM Identity Center 可用于在非Amazon 计算服务环境中开发 Amazon 应用程序时提供 Amazon 凭证。如果您正在使用诸如亚马逊弹性计算云 (Amazon EC2) Amazon Cloud9或 Amazon EC2之类的 Amazon 资源进行开发，我们建议改为从该服务获取证书。

如果您已经使用身份中心进行 Amazon 账户访问或需要管理组织的访问权限，请使用 IAM Identity Center 身份验证。

在本教程中，您将建立 IAM Identity Center 访问权限，并将使用 Amazon 访问门户和，为您的软件开发工具包或工具配置访问权限 Amazon CLI。

- Amazon 访问门户是您手动登录 IAM 身份中心的网址。URL 的格式为 `d-xxxxxxxxxx.awsapps.com/start` 或 *`your_subdomain`*`.awsapps.com/start`。登录 Amazon 访问门户后，您可以查看 Amazon Web Services 账户 已为该用户配置的角色。此过程使用 Amazon 访问门户获取 SDK/tool 身份验证过程所需的配置值。
- Amazon CLI 用于配置您的软件开发工具包或工具，使其对您的代码发出的 API 调用使用 IAM 身份中心身份验证。此一次性过程会更新您的共享 Amazon `config`文件，然后在您运行代码时由您的 SDK 或工具使用该文件。

先决条件

在开始此过程之前，您应已经完成下列步骤：

- 如果您没有 Amazon Web Services 账户，请注册 Amazon Web Services 账户.
- 如果您尚未启用 IAM Identity Center，请按照《Amazon IAM Identity Center 用户指南》中 enable IAM Identity Center 部分的说明操作。

使用 IAM Identity Center 配置以编程方式访问权限

步骤 1：建立访问权限并选择相应的权限集

选择以下方法之一来访问您的 Amazon 证书。

我尚未通过 IAM Identity Center 确立访问权限

1. 按照《Amazon IAM Identity Center 用户指南》中 Configure user access with the default IAM Identity Center directory 说明的过程添加用户并添加管理员权限。

2. AdministratorAccess 权限集不应用于普通开发用途。相反，我们建议使用预定义的 PowerUserAccess 权限集，除非您的雇主已为此目的创建了自定义权限集。

   再次按照 Configure user access with the default IAM Identity Center directory 部分说明的过程操作，不过这一次不同的是：

   - 不要创建 *Admin team* 组，而是创建一个 *Dev team* 组，然后在说明的后续部分替换为该组。
   - 您可以使用现有用户，但必须将该用户添加到新的 *Dev team* 组中。
   - 不要创建 *AdministratorAccess* 权限集，而是创建一个 *PowerUserAccess* 组权限集，然后在说明的后续部分替换为该权限集。

   完成后，您应会获得以下资源：

   - 一个 Dev team 组。
   - 一个附加到 Dev team 组的 PowerUserAccess 权限集。
   - 您的用户已添加到 Dev team 组。
3. 退出门户并再次登录以查看您的 Amazon Web Services 账户 和Administrator或选项PowerUserAccess。在使用您的工具/SDK 时选择 PowerUserAccess。

我已经 Amazon 可以通过雇主管理的联合身份提供商（例如 Microsoft Entra 或 Okta）进行访问

Amazon 通过身份提供商的门户网站登录。如果您的云管理员已授予您PowerUserAccess（开发者）权限，则您 Amazon Web Services 账户 会看到您有权访问的权限和权限集。在您的权限集名称旁边，可以看到有关使用该权限集手动或以编程方式访问账户的选项。

自定义实现可能会产生不同的体验，例如不同的权限集名称。如果您不确定要使用哪个权限集，请联系 IT 团队以寻求帮助。

我已经 Amazon 可以通过雇主管理的 Amazon 访问门户进行访问

Amazon 通过 Amazon 访问门户登录。如果您的云管理员已向您授予 PowerUserAccess（开发人员）权限，您将看到您有权访问的 Amazon Web Services 账户 和您的权限集。在您的权限集名称旁边，可以看到有关使用该权限集手动或以编程方式访问账户的选项。

我已经 Amazon 可以通过雇主管理的联合自定义身份提供商进行访问

请联系您的 IT 团队以寻求帮助。

步骤 2：配置 SDKs 和使用 IAM 身份中心的工具

1. 在您的开发计算机上安装最新的 Amazon CLI。

   a. 参阅 Amazon Command Line Interface 用户指南中的安装或更新最新版本的 Amazon CLI。

   b. （可选）要验证是否 Amazon CLI 正在运行，请打开命令提示符并运行该`aws --version`命令。

2. 登录 Amazon 访问门户。您的雇主可能会提供此 URL，或者您可以按照步骤 1：建立访问权限通过电子邮件获得该 URL。如果没有，请在的控制面板上找到您的Amazon 访问门户 URL https://console.aws.amazon.com/singlesignon/。

   a. 在 Amazon 访问门户的账户选项卡中，选择要管理的个人账户。这时将会显示您的用户的角色。选择访问密钥，以获取该权限集的命令行或编程访问凭证。使用预定义的 `PowerUserAccess` 权限集，或者您或您的雇主创建的任何权限集，以将最低权限应用于开发。

   b. 在获取凭证对话框中，选择 MacOS 和 Linux 或 Windows，具体取决于您的操作系统。

   c. 选择 IAM Identity Center 凭证方法以获取下一个步骤所需的 `Issuer URL` 和 `SSO Region` 值。注意：`SSO Start URL` 可以与 `Issuer URL` 互换使用。

3. 在 Amazon CLI 命令提示符下，运行`aws configure sso`命令。出现提示时，输入在上一步中收集的配置值。有关此 Amazon CLI 命令的详细信息，请参阅使用`aws configure sso`向导配置您的个人资料。

   a. 对于提示 `SSO Start URL`，请输入您获得的 `Issuer URL` 值。

   b. 对于 CLI 配置文件名称，我们建议您在开始*`default`*时输入。有关如何设置非默认（已命名）配置文件及其关联环境变量的信息，请参阅 配置文件。

4. （可选）在 Amazon CLI 命令提示符下，通过运行`aws sts get-caller-identity`命令确认活动会话身份。响应应显示您配置的 IAM Identity Center 权限集。

5. 如果您使用的是 S Amazon DK，请在您的开发环境中为您的 SDK 创建应用程序。

   a. 对于某些人来说 SDKs，在使用 IAM Identity Center 身份验证之前，`SSOOIDC`必须将其他软件包（例如`SSO`和）添加到您的应用程序中。有关详细信息，请参阅特定的 SDK。

   b. 如果您之前配置了对的访问权限 Amazon，请查看您的共享 Amazon `credentials`文件是否有任何访问权限Amazon 访问密钥。由于 了解默认凭证提供者链 优先级，在 SDK 或工具使用 IAM Identity Center 凭证之前，您必须移除所有静态凭证。

要深入了解 SDKs 和工具如何使用和使用此配置刷新凭据，请参阅[如何解决 Amazon SDKs 和工具的 IAM 身份中心身份验证问题]。

要直接在共享的 `config` 文件中配置 IAM Identity Center 提供者设置，请参阅本指南中的 [IAM Identity Center 凭证提供者]。

刷新门户访问会话

您的访问权限最终将过期，并且 SDK 或工具将遇到身份验证错误。何时过期取决于您配置的会话时长。要在需要时再次刷新访问门户会话， Amazon CLI 请使用运行`aws sso login`命令。

您可以延长 IAM Identity Center 访问门户会话持续时间和权限集会话持续时间。这会延长您在需要再次使用 Amazon CLI手动登录之前运行代码的时间。有关更多信息，请参阅《Amazon IAM Identity Center 用户指南》中的以下主题：

- IAM Identity Center 会话持续时间 – [配置用户 Amazon 访问门户会话的持续时间]
- 权限集会话持续时间 – [设置会话持续时间]

如何解决 Amazon SDKs 和工具的 IAM 身份中心身份验证问题

相关 IAM Identity Center 术语

以下术语可帮助您了解 Amazon IAM Identity Center背后的流程和配置。对于其中一些身份验证概念，Amazon SDK 文档 APIs 使用的名称与 IAM Identity Center 不同。知道这两个名字会很有帮助。

下表介绍了备用名称之间的关系。

IAM Identity Center 名称	SDK API 名称	说明
身份中心	`sso`	尽管已重命名 Amazon 单点登录，但出于向后兼容目的，`sso`API 命名空间仍将保留其原始名称。有关更多信息，请参阅 Amazon IAM Identity Center 用户指南中的 [IAM Identity Center 重命名]。

IAM Identity Center 名称	SDK API 名称	说明
IAM Identity Center 控制台 管理控制台		用于配置单点登录的控制台。
Amazon 访问门户网址		您的 IAM Identity Center 账户独有的 URL，例如 `https://xxx.awsapps.com/start` 。您使用您的 IAM Identity Center 登录凭证来登录此门户。
IAM Identity Center 访问门户会话	身份验证会话	向调用者提供持有者访问令牌。
权限集会话		软件开发工具包内部用于进行调用的 IAM 会 Amazon Web Services 服务 话。在非正式讨论中，您可能会看到它被错误地称为 “角色会话”。
权限集凭证	Amazon 证书 sigv4 凭证	SDK 实际用于大多数 Amazon Web Services 服务 调用（特别是所有 sigv4 Amazon Web Services 服务 调用）的凭证。在非正式讨论中，您可能会看到它被错误地称为 “角色凭证”。
IAM Identity Center 凭证提供者	SSO 凭证提供者	如何获取凭证，例如提供功能的类或模块。

了解 SDK 凭据解析 Amazon Web Services 服务

IAM Identity Center API 将持有者令牌凭证交换为 sigv4 凭证。大多数 Amazon Web Services 服务 都是 sigv4 APIs，但也有一些例外，比如 Amazon CodeWhisperer 和。 Amazon CodeCatalyst以下内容

描述了通过 Amazon IAM Identity Center支持大多数应用程序代码 Amazon Web Services 服务 调用的凭证解析流程。

启动 Amazon 访问门户会话

- 使用您的凭证登录会话以开始该过程。
  - 使用 Amazon Command Line Interface (Amazon CLI) 中的`aws sso login`命令。如果您还没有活动会话，这将启动一个新的 IAM Identity Center 会话。
- 启动新会话时，您将收到来自 IAM Identity Center 的刷新令牌和访问令牌。 Amazon CLI 还会使用新的访问令牌和刷新令牌更新 SSO 缓存 JSON 文件，并使其可供使用。 SDKs
- 如果您已经有一个活动会话，则该 Amazon CLI 命令将重复使用现有会话，并且将在现有会话过期时过期。要了解如何设置 IAM Identity Center 会话的时长，请参阅用户指南中的配置用户 Amazon 访问门户会话的Amazon IAM Identity Center 持续时间。
  - 最大会话时长已延长至 90 天，以减少频繁登录的需求。

SDK 如何获取 Amazon Web Services 服务 通话凭证

SDKs Amazon Web Services 服务 当您为每个服务实例化客户端对象时，提供访问权限。将共享 Amazon `config`文件的选定配置文件配置为 IAM Identity Center 凭证解析时，将使用 IAM Identity Center 来解析您的应用程序的证书。

- 在创建客户端时，凭证解析过程将在运行时完成。

要 APIs 使用 IAM 身份中心单点登录检索 sigv4 的证书，软件开发工具包使用 IAM 身份中心访问令牌获取 IAM 会话。此 IAM 会话称为权限集会话，它通过担任 IAM 角色提供对软件开发工具包的 Amazon 访问权限。

- 权限集会话持续时间与 IAM Identity Center 会话持续时间是分开设置的。
  - 要了解如何设置权限集会话持续时间，请参阅Amazon IAM Identity Center 用户指南中的设置会话持续时间。
- 请注意，在大多数 S Amazon DK API 文档中，权限集Amazon 凭据也被称为凭证和 sigv4 凭证。

对软件开发工具包的 IAM Identity Center API getRoleCredentials的调用会返回权限集证书。软件开发工具包的客户端对象使用该代入的 IAM 角色来调用 Amazon Web Services 服务，例如让 Amazon S3 列出您账户中的存储桶。在权限集会话到期之前，客户端对象可以使用这些权限集凭证继续操作。

会话过期和刷新

使用 SSO 令牌提供商配置 时，将使用刷新令牌自动刷新从 IAM Identity Center 获取的每小时访问令牌。

- 如果访问令牌在 SDK 尝试使用它时已过期，SDK 将使用刷新令牌来尝试获取新的访问令牌。IAM Identity Center 会将刷新令牌与您的 IAM Identity Center 访问门户会话持续时间进行比较。如果刷新令牌未过期，IAM Identity Center 将使用另一个访问令牌进行响应。
- 此访问令牌可用于刷新现有客户端的权限集会话，也可以用于解析新客户端的凭证。

但是，如果 IAM Identity Center 访问门户会话已过期，则不会授予新的访问令牌。因此，无法更新权限集持续时间。只要现有客户端的缓存权限集会话时长超时，它就会过期（并且访问权限将丢失）。

在 IAM Identity Center 会话到期后，任何创建新客户端的代码都将无法通过身份验证。这是因为未缓存权限集凭证。在您拥有有效的访问令牌之前，您的代码将无法创建新客户端并完成凭证解析过程。

总而言之，当 SDK 需要新的权限集凭证时，SDK 会首先检查所有有效的现有凭证并使用这些凭证。无论凭证是针对新客户端，还是凭证已过期的现有客户端，这都适用。如果找不到凭证或凭证无效，则 SDK 会调用 IAM Identity Center API 来获取新凭证。要调用 API，它需要访问令牌。如果访问令牌已过期，SDK 会使用刷新令牌尝试从 IAM Identity Center 服务获取新的访问令牌。如果您的 IAM Identity Center 访问门户会话未过期，则会授予此令牌。

使用 IAM Anywhere 角色进行身份验证 Amazon SDKs 和工具

您可以使用 IAM Roles Anywhere 在 IAM 中为在外部运行的服务器、容器和应用程序等工作负载获取临时安全证书 Amazon。要使用 IAM Roles Anywhere，您的工作负载必须使用 X.509 证书。您的云管理员应提供所需的证书和私钥，以便将 IAM Roles Anywhere 配置为凭证提供者。

第 1 步：配置 IAM Roles Anywhere

IAM Roles Anywhere 提供了一种获取在外部运行的工作负载或流程的临时证书的方法 Amazon。与证书颁发机构建立信任锚，以获取关联的 IAM 角色的临时凭证。该角色设置当您的代码使用 IAM Roles Anywhere 进行身份验证时您的工作负载将拥有的权限。

有关设置信任锚点、IAM 角色和 IAM Anywhere 角色配置文件的步骤，请参阅 IA M Roles Anywhere 用户指南中的随处 Amazon Identity and Access Management 角色中创建信任锚和配置文件。

> **Note**
>
> IAM Roles Anywhere 用户指南中的配置文件指的是 IAM Roles Anywhere 服务中的一个独特概念。它与共享 Amazon config文件中的配置文件无关。

第 2 步：使用 IAM Roles Anywhere

要从 IAM Roles Anywhere 获取临时安全凭证，请使用 IAM Roles Anywhere 提供的凭证助手。凭证工具可实现 IAM Roles Anywhere 的签名流程。

有关下载凭证帮助工具的说明，请参阅 IAM Roles Anywhere 用户指南中的从 Amazon Identity and Access Management 任何地方的角色获取临时安全证书。

要将来自 IAM 角色的 Anywhere 临时安全证书与 Amazon SDKs 和一起使用 Amazon CLI，您可以在共享 Amazon config文件中配置credential_process设置。 SDKs 和 Amazon CLI 支持用于credential_process进行身份验证的流程凭证提供程序。下面显示了要设置credential_process的一般结构。

```
credential_process = [path to helper tool] [command] [--parameter1 value] [--
parameter2 value] [...]
```

助手工具的 credential-process 命令以与 credential_process 设置兼容的标准 JSON 格式返回临时凭证。请注意，命令名称包含连字符，而设置名称包含下划线。命令需要使用以下参数：

- private-key – 签署请求的私钥的路径。
- certificate – 证书的路径。
- role-arn – 要为其获取临时凭证的角色的 ARN。
- profile-arn – 为指定角色提供映射的配置文件的 ARN。
- trust-anchor-arn – 用于身份验证的信任锚的 ARN。

您的云管理员将提供证书和私钥。所有三个 ARN 值都可以从 Amazon Web Services 管理控制台复制。以下示例显示了共享config文件，该文件配置了从助手工具检索临时凭证。

```
[profile dev]
credential_process = ./aws_signing_helper credential-process --certificate /
path/to/certificate --private-key /path/to/private-key --trust-anchor-
```

```
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-
arn arn:aws:iam::account:role/ROLE_ID
```

有关可选参数和其他帮助工具的详细信息，请参阅上 GitHub的 IAM Roles Anywhere 凭证助手。

有关 SDK 配置设置本身和流程凭证提供者的详细信息，请参阅本指南中的 进程凭证提供者。

假设一个拥有身份验证 Amazon 凭证 Amazon SDKs 和工具的角色

假设角色涉及使用一组临时安全凭证来访问您原本无法访问的 Amazon 资源。这些临时凭证由访问密钥 ID、秘密访问密钥和安全令牌组成。要了解有关 Amazon Security Token Service （ Amazon STS ）API 请求的更多信息，请参阅《Amazon Security Token Service API 参考》中的操作。

要设置您的 SDK 或工具来代入角色，必须先创建或标识要代入的特定角色。IAM 角色由角色 Amazon 资源名称（ ARN ）进行唯一标识。角色与另一个实体建立信任关系。使用该角色的可信实体可能是一个 Amazon Web Services 服务 或另一个实体 Amazon Web Services 账户。有关 IAM 角色的更多一般信息，请参阅《IAM 用户指南》中的 IAM 角色。

标识 IAM 角色后，如果您受到该角色的信任，则可以将您的 SDK 或工具配置为使用该角色授予的权限。

> **Note**
>
> Amazon 最佳做法是尽可能使用区域终端节点并配置您的终端节点Amazon Web Services 区域。

代入 IAM 角色

担任角色时， Amazon STS 返回一组临时安全证书。这些凭证来自另一个配置文件或运行代码的实例或容器。此类代入角色的最常见使用场景是，当您拥有一个账户的 Amazon 凭证，但您的应用程序需要访问另一个账户中的资源时。

步骤 1：设置 IAM 角色

要设置您的 SDK 或工具来代入角色，必须先创建或标识要代入的特定角色。IAM 角色使用角色 ARN 进行唯一标识。角色与另一个实体建立信任关系，通常是在您的账户内或用于跨账户访问。要了解更多信息，请参阅 IAM 用户手册中的创建 IAM 角色。

步骤 2：配置 SDK 或工具

将 SDK 或工具配置为从 credential_source 或 source_profile 获取凭证。

credential_source 用于从 Amazon ECS 容器、Amazon EC2 实例或环境变量中获取凭证。

source_profile 用于从另一个配置文件获取凭证。source_profile 还支持角色链，即配置文件的层次结构，然后使用代入的角色来代入另一个角色。

当您在配置文件中指定此项时，SDK 或工具会自动为您调用相应 Amazon STS AssumeRole的 API。要通过担任角色来检索和使用临时证书，请在共享 Amazon config文件中指定以下配置值。有关这些设置各自的更多信息，请参阅 代入角色凭证提供者设置 节。

- role_arn - 来自您在步骤 1 中创建的 IAM 角色
- 配置 credential_source 或 source_profile
- （可选）duration_seconds
- （可选）external_id
- （可选）mfa_serial
- （可选）role_session_name

以下示例显示了共享 config 文件中两个代入角色选项的配置：

```
role_arn = arn:aws:iam::123456789012:role/my-role-name
credential_source = Ec2InstanceMetadata
```

```
[profile-with-user-that-can-assume-role]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONGSTRI

[profile dev]
region = us-east-1
output = json
role_arn = arn:aws:iam::123456789012:role/my-role-name
source_profile = profile-with-user-that-can-assume-role
role_session_name = my_session
```

有关所有代入角色凭证提供程序设置的详细信息，请参阅本指南中的 代入角色凭证提供者。

假设角色使用网络身份或 OpenID Connect 进行身份验证和工具 Amazon SDKs

假设角色涉及使用一组临时安全凭证来访问您原本无法访问的 Amazon 资源。这些临时凭证由访问密钥 ID、秘密访问密钥和安全令牌组成。要了解有关 Amazon Security Token Service （ Amazon STS ）API 请求的更多信息，请参阅《Amazon Security Token Service API 参考》中的操作。

要设置您的 SDK 或工具来代入角色，必须先创建或标识要代入的特定角色。IAM 角色由角色 Amazon 资源名称（ ARN ）进行唯一标识。角色与另一个实体建立信任关系。使用该角色的可信实体可能是某个 Web 身份提供者、OpenID Connec（ OIDC ）或 SAML 联合身份验证。有关 IAM 角色的更多信息，请参阅《IAM 用户 用户指南》中的代入角色的方法。

在您的软件开发工具包中配置 IAM 角色后，如果将该角色配置为信任您的身份提供商，则可以进一步配置您的软件开发工具包以代入该角色以获得临时 Amazon 证书。

> **Note**
>
> Amazon 最佳做法是尽可能使用区域终端节点并配置您的终端节点Amazon Web Services 区域。

使用 Web 身份或 OpenID Connect 进行联合

你可以使用公共身份提供商（ 例如 Login With Amazon、Facebook、GoogleJWTs ）提供的 JSON 网络令牌 () 来获取临时 Amazon 证书`AssumeRoleWithWebIdentity`。根据它们的使用方式，它们JWTs可能被称为 ID 令牌或访问令牌。您也可以使用由与 OIDC 发现协议兼容的身份提供商 (IdPs) JWTs 签发，例如 EntraId 或。 PingFederate

如果您使用的是 Amazon Elastic Kubernetes Service，则此功能可让您为 Amazon EKS 集群中的每个服务账户指定不同的 IAM 角色。这个 Kubernetes 功能会分发 JWTs 到你的 pod 中，然后由该凭证提供者使用这些容器来获取临时证书。 Amazon 有关此 Amazon EKS 配置的更多信息，请参阅《Amazon EKS 用户指南》中的服务账户的 IAM 角色。但是，对于更简单的选项，我们建议您改用 Amazon EKS 容器组身份，前提是您的 SDK 支持它。

步骤 1：设置身份提供者和 IAM 角色

要配置与外部 IdP 的联合，请使用 IAM 身份提供商 Amazon 通知外部 IdP 及其配置。这将在您 Amazon Web Services 账户 和外部 IdP 之间建立信任。在将 SDK 配置为使用 JSON Web 令牌

（JWT）进行身份验证之前，必须先设置身份提供者（IdP）以及用于访问令牌的 IAM 角色。要进行设置，请参阅 IAM 用户指南中的 创建 Web 身份或 OpenID Connect 联合身份验证角色（控制台）。

步骤 2：配置 SDK 或工具

将 SDK 或工具配置为使用来自的 JSON Web 令牌 (JWT) Amazon STS 进行身份验证。

当您在配置文件中指定此项时，SDK 或工具会自动为您调用相应 Amazon STS `AssumeRoleWithWebIdentity`的 API。要使用 Web 联合身份验证检索和使用临时证书，请在共享 Amazon `config`文件中指定以下配置值。有关这些设置各自的更多信息，请参阅 代入角色凭证提供者设置 节。

- `role_arn` - 来自您在步骤 1 中创建的 IAM 角色
- `web_identity_token_file` - 来自外部 IdP
- （可选）`duration_seconds`
- （可选）`role_session_name`

以下是使用 Web 身份代入角色的共享 `config` 文件配置示例：

```
[profile web-identity]
role_arn=arn:aws:iam::123456789012:role/my-role-name
web_identity_token_file=/path/to/a/token
```

> **Note**
>
> 有关移动应用程序，请考虑使用 Amazon Cognito。Amazon Cognito 充当身份凭证代理程序并为您完成许多联合身份验证工作。但是，Amazon Cognito 身份提供商不像其他身份提供商那样包含在 SDKs 和工具核心库中。要访问 Amazon Cognito API，请在您的 SDK 或工具的构建或库中包含 Amazon Cognito 服务客户端。有关与的用法 Amazon SDKs，请参阅 Amazon Cognito 开发者指南中的代码示例。

有关所有代入角色凭证提供程序设置的详细信息，请参阅本指南中的 代入角色凭证提供者。

使用 Amazon 访问密钥进行身份验证 Amazon SDKs 和工具

使用 Amazon SDKs 和工具时，可以选择使用 Amazon 访问密钥进行身份验证。

使用短期凭证

我们建议将您的 SDK 或工具配置为使用 使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证 以使用延长的会话持续时间选项。

但是，要直接设置 SDK 或工具的临时凭证，请参阅 使用短期凭证进行身份验证 Amazon SDKs 和工具。

使用长期凭证

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

管理跨区域的访问权限 Amazon Web Services 账户

作为安全最佳实践，我们建议 Amazon Organizations 与 IAM Identity Center 配合使用来管理所有人的访问权限 Amazon Web Services 账户。有关更多信息，请参阅《IAM 用户指南》中的 IAM 安全最佳实践。

你可以在 IAM Identity Center 中创建用户，使用 Microsoft Active Directory，使用 SAML 2.0 身份提供商 (IdP)，或者将你的 IdP 单独联合到其中。 Amazon Web Services 账户您可以使用其中一种方法，为用户提供单点登录体验。您还可以强制执行多重身份验证 (MFA) 并使用临时证书 Amazon Web Services 账户 进行访问。这与 IAM 用户不同，后者是一种可以共享的长期凭证，并且可能会增加 Amazon 资源的安全风险。

仅为沙盒环境创建 IAM 用户

如果您不熟悉 Amazon，可以创建一个测试 IAM 用户，然后使用它来运行教程并探索 Amazon 所提供的内容。在学习时可以使用此类凭证，但我们建议您避免在沙盒环境之外使用。

对于以下用例，开始使用 IAM 用户可能是有意义的 Amazon：

- 开始使用您的 Amazon SDK 或工具，并在沙盒环境 Amazon Web Services 服务 中进行探索。
- 在学习过程中，运行不支持人工参与登录流程的计划脚本、作业和其他自动化流程。

如果您在这些用例之外使用 IAM 用户，请尽快过渡到 IAM Identity Center 或将您的身份提供商联合到该 Amazon Web Services 账户 中心。有关更多信息，请参阅 Amazon中的身份联合验证。

确保 IAM 用户访问密钥安全

您应该定期轮换 IAM 用户访问密钥。参阅《IAM 用户指南》，按照轮换访问密钥中的指导进行操作。如果您认为自己不小心共享了您的 IAM 用户访问密钥，请轮换您的访问密钥。

IAM 用户访问密钥应存储在本地计算机上的共享 Amazon `credentials`文件中。请勿将 IAM 用户访问密钥存储在您的代码中。请勿将包含 IAM 用户访问密钥的配置文件存储到任何源代码管理软件中。开源项目 git-secrets 等外部工具可以帮助您避免无意中将敏感信息提交到 Git 存储库。有关更多信息，请参阅 IAM 用户指南 中的 IAM 身份（用户、用户组和角色）。

要设置 IAM 用户以开始使用，请参阅 使用长期凭证进行身份验证 Amazon SDKs 和工具。

使用短期凭证进行身份验证 Amazon SDKs 和工具

我们建议配置您的 Amazon SDK 或工具，使其使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证与延长会话持续时间选项一起使用。但是，您可以复制和使用 Amazon 访问门户中提供的临时证书。在这些临时凭证过期后，将需要复制新凭证。您可以在配置文件中使用临时凭证，也可以将其用作系统属性和环境变量的值。

最佳实践：我们建议您的应用程序使用来自以下来源的临时凭证，而不是手动管理凭证文件中的访问密钥和令牌：

- 一种 Amazon 计算服务，例如在 Amazon Elastic Compute Cloud 或中运行您的应用程序 Amazon Lambda。
- 凭证提供者链中的其他选项，例如使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证。
- 也可使用 进程凭证提供者 来检索临时凭证。

使用从 Amazon 访问门户中检索到的短期凭证设置凭证文件

1. 创建共享凭证文件。
2. 在凭证文件中，粘贴以下占位符文本，直到粘贴有效的临时凭证为止。

```
[default]
aws_access_key_id=<value from Amazon access portal>
aws_secret_access_key=<value from Amazon access portal>
aws_session_token=<value from Amazon access portal>
```

3. 保存该文件。文件 `~/.aws/credentials` 现在应该存在于您的本地开发系统上。如果未指定特定的命名配置文件，此文件包含 SDK 或工具使用的[默认] 配置文件。

4. 登录 Amazon 访问门户。
5. 按照这些手动刷新凭证的说明从 Amazon 访问门户复制 IAM 角色证书。
   a. 对于链接的说明中的步骤 4，选择可授予访问权限以满足您的开发需求的 IAM 角色名称。此角色的名称通常类似于PowerUserAccess或开发人员。
   b. 对于链接的说明中的步骤 7，选择手动将配置文件添加到您的 Amazon 凭证文件选项并复制内容。
6. 将复制的凭证粘贴到您的本地`credentials`文件中。如果您使用的是`default`配置文件，则不需要生成的配置文件名称。您的文件应类似于以下内容。

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token=IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZ2luX2IQoJb3JpZVERYLONG
```

7. 保存 `credentials` 文件。

当 SDK 创建服务客户端时，它将访问这些临时凭证并将它们用于每个请求。在步骤 5a 中选择的 IAM 角色的设置决定了临时凭证的有效时间。最长持续时间为 12 小时。

在临时凭证过期后，重复步骤 4 到 7。

使用长期凭证进行身份验证 Amazon SDKs 和工具

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

如果您使用 IAM 用户运行您的代码，则开发环境中的软件开发工具包或工具将使用共享 Amazon `credentials`文件中的长期 IAM 用户证书进行身份验证。查看 IAM 主题中的安全最佳实践，并尽快过渡到 IAM Identity Center 或其他临时凭证。

有关凭证的重要警告和指南

有关凭证的警告

- 请勿使用账户的根凭证来访问 Amazon 资源。这些凭证可提供不受限的账户访问且难以撤销。

- 不得在应用程序文件中按字面输入访问密钥或凭证信息。如果您这样做，则在将项目上传到公共存储库或在其他情况下，会有意外暴露凭证的风险。
- 不得在项目区域中放入包含凭证的文件。
- 请注意，存储在共享 Amazon `credentials`文件中的所有凭据都以纯文本形式存储。

有关安全管理凭证的更多指南

有关如何安全管理 Amazon 证书的一般性讨论，请参阅中的管理 Amazon 访问密钥的最佳实践Amazon Web Services 一般参考。除了上述讨论内容外，请考虑以下事项：

- 对于 Amazon Elastic Container Service (Amazon ECS)，使用适用于任务的 IAM 角色。
- 对于在 Amazon EC2 实例上运行的应用程序，使用 IAM 角色。

先决条件：创建 Amazon 账户

要使用 IAM 用户访问 Amazon 服务，您需要一个 Amazon 账户和 Amazon 证书。

1. 创建账户。

   要创建 Amazon 账户，请参阅入门：你是首次 Amazon 使用吗？ 在《Amazon 账户管理 参考指南》中。

2. 创建管理用户。

   请勿使用 root 用户账户（您创建的初始账户）访问管理控制台和服务。而是创建一个管理用户账户，如《IAM 用户指南》的创建管理用户中所述。

   创建管理用户账户并记录登录详细信息后，务必注销根用户账户并使用管理账户重新登录。

这两个帐户都不适合在上面进行开发 Amazon 或在上运行应用程序 Amazon。作为最佳实践，您需要创建适合这些任务的用户、权限集或服务角色。有关更多信息，请参阅《IAM 用户指南》中的应用最低权限许可。

步骤 1：创建您的 IAM 用户

- 按照《IAM 用户指南》中的创建 IAM 用户（控制台）过程操作来创建 IAM 用户。创建 IAM 用户时：

- 我们建议您选择向用户提供 Amazon Web Services 管理控制台访问权限。这可让您通过可视化的环境查看与您运行的代码相关的 Amazon Web Services 服务 ，例如检查 Amazon CloudTrail 诊断日志或将文件上传到 Amazon Simple Storage Service。这在代码调试时将非常实用。
- 对于设置权限 - 权限选项，请选择直接附加策略，以确定您向该用户分配权限的方式。
  - 大多数“入门”开发工具包教程都使用 Amazon S3 服务作为示例。要向应用程序提供对 Amazon S3 的完全访问权限，请选择要附加到此用户的 `AmazonS3FullAccess` 策略。
- 您可以忽略该过程中有关设置权限边界或标签的可选步骤。

步骤 2：获取您的访问密钥

1. 在 IAM 控制台的导航窗格中，选择用户，然后选择您之前创建用户的 **`User name`**。
2. 在用户的页面上，选择安全凭证页面。然后，在访问密钥下，选择创建访问密钥。
3. 对于创建访问密钥步骤 1，选择 命令行界面 (CLI) 或 本地代码。这两个选项生成的密钥类型相同，可与 Amazon CLI 和一起使用 SDKs。
4. 对于创建访问密钥步骤 2，输入可选标记并选择下一步。
5. 对于创建访问密钥步骤 3，选择下载.csv 文件以保存包含您的 IAM 用户访问密钥和秘密访问密钥的 `.csv` 文件。稍后您将需要此信息。

> **Warning**
>
> 使用适当的安全措施来确保这些凭证的安全。

6. 选择 Done (完成)。

步骤 3：更新共享 **`credentials`** 文件

1. 创建或打开共享 Amazon `credentials` 文件。此文件在 Linux 和 macOS 系统上为 `~/.aws/credentials`，在 Windows 上为 `%USERPROFILE%\.aws\credentials`。有关更多信息，请参阅 凭证文件位置。
2. 将以下文本添加到共享 `credentials` 文件中。将示例 ID 值和示例密钥值替换为先前下载的 `.csv` 文件中的值。

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
```

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

3. 保存该文件。

共享 `credentials` 文件是存储凭证的最常见方式。它们也可以设置为环境变量，有关环境变量的名称，请参阅 Amazon 访问密钥。这是一种入门方式，但我们建议您尽快过渡到 IAM Identity Center 或其他临时凭证。停止使用长期凭证后，请记得从共享 `credentials` 文件中删除这些凭证。

使用 IAM 角色对部署到 Amazon 的应用程序进行身份验证 EC2

此示例介绍如何设置一个拥有 Amazon S3 访问权限的 Amazon Identity and Access Management 角色，以便在部署到亚马逊弹性计算云实例的应用程序中使用。

要在亚马逊弹性计算云实例上运行您的 Amazon 软件开发工具包应用程序，请创建一个 IAM 角色，然后授予您的亚马逊 EC2 实例访问该角色的权限。有关更多信息，请参阅《亚马逊 EC2 用户指南》EC2中的 Amazon IAM 角色。

创建一个 IAM 角色

您开发的 Amazon SDK 应用程序可能至少访问一个 SDK 应用程序 Amazon Web Services 服务 来执行操作。创建一个 IAM 角色来授予应用程序运行所需的权限。

例如，以下过程会创建一个将授予对 Amazon S3 的只读权限的角色。许多 Amazon SDK 指南都有从 Amazon S3 中读出的 “入门” 教程。

1. 登录 Amazon Web Services 管理控制台 并打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色，然后选择创建角色。
3. 对于 选择可信实体，在 可信实体类型，选择 Amazon Web Services 服务。
4. 在 “用例” 下，选择 Amazon EC2，然后选择 “下一步”。
5. 对于添加权限，请从策略列表中选中 Amazon S3 只读访问权限复选框，然后选择下一步。
6. 输入角色的名称，然后选择创建角色。请记住这个名称，因为您在创建 Amazon EC2 实例时会用到它。

启动 Amazon EC2 实例并指定您的 IAM 角色

您可以通过执行以下操作，使用您的 IAM 角色创建和启动 Amazon EC2 实例：

- 在 Amazon EC2 用户指南中按照 “快速启动实例” 进行操作。但在执行最后的提交步骤前，还应执行以下操作：
  - 在高级详细信息下，对于 IAM 实例配置文件，选择您在上一步中创建的角色。

通过此 IAM 和 Amazon EC2 设置，您可以将应用程序部署到亚马逊 EC2 实例，您的应用程序将拥有对 Amazon S3 服务的读取权限。

Connect 连接到 EC2 实例

连接到 Amazon EC2 实例，这样您就可以将应用程序传输到该实例，然后运行该应用程序。您将需要包含当您创建实例时在密钥对（登录）部分下所用密钥对私有部分的文件；即 PEM 文件。

为此，您可以按照相应实例类型的指南进行操作：连接到 Linux 实例或连接到 Windows 实例。当您连接时，请确保您可以将文件从开发计算机传输到您的实例。

> **Note**
>
> 在 Linux 或 macOS 终端上，您可以使用安全复制命令来复制应用程序。要将 `scp` 与某个密钥对结合使用，可以使用以下命令：`scp -i path/to/key file/to/copy ec2-user@ec2-xx-xx-xxx-xxx.compute.amazonaws.com:~`。
> 有关 Windows 的更多信息，请参阅将文件传输到 Windows 实例。

如果您使用的是 Amazon 工具包，则通常也可以使用工具包连接到实例。有关更多信息，请参阅您使用的工具包的特定用户指南。

在 EC2 实例上运行您的应用程序

1. 将您的应用程序文件从本地驱动器复制到您的 Amazon EC2 实例。
2. 启动应用程序并验证其运行结果是否与开发计算机上的结果相同。
3. （可选）验证应用程序是否使用 IAM 角色提供的凭证。
   a. 登录 Amazon Web Services 管理控制台 并打开 Amazon EC2 控制台，网址为 https://console.aws.amazon.com/ec2/。
   b. 选择实例。
   c. 依次选择操作、安全性和修改 IAM 角色。
   d. 对于 IAM 角色，请选择无 IAM 角色来分离 IAM 角色。

e. 选择更新 IAM 角色。

f. 再次运行该应用程序，并确认它返回了授权错误。

使用 TIP 插件进行访问 Amazon Web Services 服务

可信身份传播 (TIP) 是一项功能 Amazon IAM Identity Center ，允许的 Amazon Web Services 服务管理员根据用户属性（例如群组关联）授予权限。通过可信身份传播，可以向 IAM 角色添加身份上下文，以识别请求访问 Amazon 资源的用户。此上下文会传播到其他 Amazon Web Services 服务。

身份上下文包含在他们收到访问请求时 Amazon Web Services 服务 用于做出授权决策的信息。这些信息包括识别请求者（例如，IAM 身份中心用户）、请求访问权限的元数据（例如 Amazon Redshift）和访问范围（例如，只读权限）的元数据。 Amazon Web Services 服务 接收方 Amazon Web Services 服务 使用此上下文以及分配给用户的任何权限来授权访问其资源。有关更多信息，请参阅《 Amazon IAM Identity Center 用户指南》中的 “可信身份传播概述”。

使用 TIP 插件的先决条件

需要具有下列资源才能正常使用该插件：

1. 您必须使用 适用于 Java 的 Amazon SDK 或 适用于 JavaScript 的 Amazon SDK。
2. 确认您使用的服务支持可信身份传播。

   请参阅《Amazon IAM Identity Center 用户指南》中 Amazon managed applications that integrate with IAM Identity Center 表的 Enables trusted identity propagation through IAM Identity Center 列。
3. 启用 IAM Identity Center 和可信身份传播。

   请参阅《Amazon IAM Identity Center 用户指南》中的 TIP prerequisites and considerations。
4. 你必须有一个 Identity-Center-integrated应用程序。

   请参阅《Amazon IAM Identity Center 用户指南》中的 Amazon managed applications 或 Customer managed applications。
5. 您必须设置一个可信令牌颁发者（TTI）并将您的服务连接到 IAM Identity Center。

   请参阅《Amazon IAM Identity Center 用户指南》中的 Prerequisites for trusted token issuers 和 Tasks for setting up a trusted token issuer。

在代码中使用 TIP 插件

1. 创建可信身份传播插件实例。
2. 创建用于与您的交互的服务客户端实例， Amazon Web Services 服务 并通过添加可信身份传播插件来自定义服务客户端。

TIP 插件使用以下输入参数：

- **webTokenProvider**：客户为了从其外部身份提供者处获取 OpenID 令牌而实现的函数。
- **accessRoleArn**：该插件要使用用户的身份上下文代入的 IAM 角色的 ARN，用来获取身份增强型凭证。
- **applicationArn**：客户端或应用程序的唯一标识符字符串。此值是已配置 OAuth 授权的应用程序 ARN。
- **ssoOidcClient**:（可选）具有客户定义配置的 SSO OIDC 客户端，例如`SsoOidcClient`适用于 Java 或 for `client-sso-oidc`的 JavaScript客户端。如果未提供，则将实例化并使用采用 `applicationRoleArn` 的 OIDC 客户端。
- **stsClient**：（可选）具有客户定义配置的 Amazon STS 客户端，用于使用用户的身份上下文代入 `accessRoleArn`。如果未提供，则`applicationRoleArn`将实例化并使用正在使用的 Amazon STS 客户端。
- **applicationRoleArn**:（可选）要使用的 IAM 角色 ARN，`AssumeRoleWithWebIdentity`以便可以引导 OIDC 和 Amazon STS 客户端。
  - 如果未提供，则必须同时提供 `ssoOidcClient` 和 `stsClient` 参数。
  - 如果提供，则 `applicationRoleArn` 的值不能与 `accessRoleArn` 参数的值相同。`applicationRoleArn` 用来构建用于代入 accessRole 的 stsClient。如果两者都使用相同的角色 `applicationRoleaccessRole`，则意味着使用角色来扮演自己（自我角色假设），这是不鼓励的。 Amazon有关更多详细信息，请参阅公告。

`ssoOidcClient`、`stsClient` 和 `applicationRoleArn` 参数的注事意项

配置 TIP 插件时，根据您提供的参数应注意以下权限要求：

- 如果您提供了 `ssoOidcClient` 和 `stsClient`：
  - `ssoOidcClient` 上的凭证应具有 `oauth:CreateTokenWithIAM` 权限，以调用 Identity Center 来获取 Identity Center 特定的用户上下文。

  - stsClient 上的凭证应具有 accessRole 上的 sts:AssumeRole 和 sts:SetContext 权限。accessRole 还需要配置与 stsClient 上的凭证的信任关系。
- 如果您提供了 applicationRoleArn：
  - applicationRole 应具有所需资源（IdC 实例、accessRole）的 oauth:CreateTokenWithIAM、sts:AssumeRole 和 sts:SetContext 权限，因为该角色将用于构建 OIDC 和 STS 客户端。
  - applicationRole应该与用于生成的身份提供者建立信任关系webToken，因为webToken将用于通过插件的AssumeRoleWithWebIdentity调用来假设 ApplicationRole。

ApplicationRole 配置示例：

Web 令牌提供者的信任策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::ACCOUNT_ID:oidc-provider/
IDENTITY_PROVIDER_URL"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "IDENTITY_PROVIDER_URL:aud": "CLIENT_ID_TO_BE_TRUSTED"
                }
            }
        }
    ]
}
```

权限策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Resource": [
                "accessRoleArn"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "sso-oauth:CreateTokenWithIAM"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

使用 TIP 的代码示例

以下示例显示了如何使用 适用于 Java 的 Amazon SDK 或在代码中实现 TIP 插件 适用于 JavaScript 的 Amazon SDK。

Java

要在 适用于 Java 的 Amazon SDK 项目中使用 TIP 插件，您需要在项目`pom.xml`文件中将其声明为依赖项。

```
<dependency>
<groupId>software.amazon.awsidentity.trustedIdentityPropagation</groupId>
<artifactId>aws-sdk-java-trustedIdentityPropagation-java-plugin</artifactId>
   <version>2.0.0</version>
</dependency>
```

在源代码中，包括 `software.amazon.awssdk.trustedidentitypropagation` 的必需软件包语句。

以下示例演示了创建可信身份传播插件实例并将其添加到服务客户端的两种方式。这两个示例都使用 Amazon S3 作为服务并`S3AccessGrantsPlugin`用来管理用户的特定权限，但可以应用于任何 Amazon Web Services 服务 支持可信身份传播 (TIP) 的权限。

> **Note**
>
> 对于这些示例，您需要从 S3 访问权限管控设置用户特定的权限。有关更多详细信息，请参阅 S3 访问权限管控文档。

选项 1：构建并传递 OIDC 和 STS 客户端

```
SsoOidcClient oidcClient = SsoOidcClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

StsClient stsClient = StsClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
 TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .ssoOidcClient(oidcClient)
        .stsClient(stsClient)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
        .build();

S3Client s3Client =
        S3Client.builder().region(Region.US_EAST_1)
                .crossRegionAccessEnabled(true)
                .addPlugin(trustedIdentityPropagationPlugin)
                .addPlugin(accessGrantsPlugin)
                .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
        .key("path/to/object/fileName")
        .bucket("bucketName")
        .build());
```

选项 2：将客户端创建传递给插件 applicationRoleArn 并推迟创建

```
TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
 TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .applicationRoleArn(applicationRoleArn)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
        .build();

S3Client s3Client =
        S3Client.builder().region(Region.US_EAST_1)
                .crossRegionAccessEnabled(true)
                .addPlugin(trustedIdentityPropagationPlugin)
                .addPlugin(accessGrantsPlugin)
                .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
        .key("path/to/object/fileName")
        .bucket("bucketName")
        .build());
```

有关更多详细信息和来源，请参阅 trusted-identity-propagation-java上的 GitHub。

JavaScript

运行以下命令在您的 适用于 JavaScript 的 Amazon SDK 项目中安装 TIP 身份验证插件包：

```
$  npm i @aws-sdk-extension/trusted-identity-propagation
```

最终的 `package.json` 应包含与以下类似的依赖项：

```
  "dependencies": {
"@aws-sdk-extension/trusted-identity-propagation": "^2.0.0"
  },
```

在源代码中，导入所需的 `TrustedIdentityPropagationExtension` 依赖项。

以下示例演示了创建可信身份传播插件实例并将其添加到服务客户端的两种方式。这两个示例都使用 Amazon S3 作为服务，并利用 Amazon S3 访问权限来管理用户的特定权限，但可以应用于任何 Amazon Web Services 服务 支持可信身份传播 (TIP) 的权限。

> **Note**
>
> 对于这些示例，您需要从Amazon S3 访问权限管控中设置用户特定的权限。有关更多详细信息，请参阅 Amazon S3 访问权限管控文档。

选项 1：构建并传递 OIDC 和 STS 客户端

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";
import { TrustedIdentityPropagationExtension } from "@aws-sdk-extension/trusted-
identity-propagation";

const s3ControlClient = new S3ControlClient({
    region: "us-east-1",
    extensions: [
        TrustedIdentityPropagationExtension.create({
            webTokenProvider: async () => {
                return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';
            },
            ssoOidcClient: customOidcClient,
            stsClient: customStsClient,
            accessRoleArn: accessRoleArn,
            applicationArn: applicationArn,
        }),
    ],
});

const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  // Create a new S3 client with the temporary credentials
```

```
  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });

  // Use the temporary S3 client to perform the operation
  const s3Params = {
    Bucket: "BUCKET_NAME",
    Key: "S3_OBJECT_KEY",
  };
  const getObjectCommand = new GetObjectCommand(s3Params);
  const s3Object = await temporaryS3Client.send(getObjectCommand);

  const fileContent = await s3Object.Body.transformToString();

  // Process the S3 object data
  console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

选项 2：将客户端创建传递给插件 applicationRoleArn 并推迟创建

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";
import { TrustedIdentityPropagationExtension } from "@aws-sdk-extension/trusted-
identity-propagation";

const s3ControlClient = new S3ControlClient({
    region: "us-east-1",
    extensions: [
        TrustedIdentityPropagationExtension.create({
            webTokenProvider: async () => {
                return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';
            },
            accessRoleArn: accessRoleArn,
            applicationRoleArn: applicationRoleArn,
            applicationArn: applicationArn,
        }),
```

```
    ],
});

// Same S3 AccessGrants workflow as Option 1
const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });

  const s3Params = {
    Bucket: "BUCKET_NAME",
    Key: "S3_OBJECT_KEY",
  };
  const getObjectCommand = new GetObjectCommand(s3Params);
  const s3Object = await temporaryS3Client.send(getObjectCommand);

  const fileContent = await s3Object.Body.transformToString();

  console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

有关更多详细信息和来源，请参阅trusted-identity-propagation-js上的 GitHub。

Amazon SDKs 和工具设置参考

SDKs 提供特定于语言的内容 APIs 。 Amazon Web Services 服务它们负责成功进行 API 调用所需的一些繁重工作，包括身份验证、重试行为等。为此， SDKs 他们需要制定灵活的策略来获取用于您的请求的凭证，维护用于每项服务的设置，以及获取用于全局设置的值。

以下章节介绍了有关配置设置的详细信息：

- Amazon SDKs 和 Tools 标准化凭证提供商— 通用凭证提供商在多个 SDKs证书提供者之间实现标准化。
- Amazon SDKs 和 “工具” 标准化功能— 通用功能跨多个标准化 SDKs。

创建服务客户端

要以编程方式访问 Amazon Web Services 服务， SDKs 请分别 Amazon Web Services 服务使用客户端 class/object 。例如，如果您的应用程序需要访问亚马逊 EC2，则您的应用程序会创建一个 Amazon EC2 客户端对象来与该服务接口。然后，您可以使用服务客户端向该 Amazon Web Services 服务发出请求。在大多数情况下 SDKs，服务客户端对象是不可变的，因此您必须为向其发出请求的每个服务创建一个新的客户端，并使用不同的配置向同一服务发出请求。

设置的优先级

全局设置配置了大多数 SDKs 人支持并具有广泛 Amazon Web Services 服务影响的功能、凭证提供程序和其他功能。所有地方 SDKs 都有一系列地点（或来源），他们会检查这些地点（或来源），以便找到全局设置的值。以下是设置查找优先级的方法：

1. 在代码中或服务客户端本身上设置的任何显式设置均优先于其他任何设置。
   - 有些设置可以根据每个操作进行设置，也可以根据需要针对调用的每个操作进行更改。对于 Amazon CLI 或 Amazon Tools for PowerShell，它们采用您在命令行上输入的每个操作参数的形式。对于 SDK，显式分配可以采用您在实例化 Amazon Web Services 服务 客户端或配置对象时或有时在调用单个 API 时设置的参数的形式。
2. 仅限 Java/Kotlin：检查该设置的 JVM 系统属性。如果已设置，将使用该值来配置客户端。
3. 系统会检查环境变量。如果已设置，将使用该值来配置客户端。
4. SDK 会在共享的 `credentials` 文件中检查该设置。如果已设置，则客户端将使用该值。
5. 检查共享的 `config` 文件来查找该设置。如果存在该设置，则 SDK 将使用该设置。

- 可以使用 AWS_PROFILE 环境变量或 aws.profile JVM 系统属性来指定 SDK 要加载的配置文件。

6. 最后才会使用 SDK 源代码本身提供的任何默认值。

> **Note**
>
> 有些 SDKs 工具可能会按不同的顺序进行检查。此外，有些 SDKs 和工具还支持其他存储和检索参数的方法。例如， 适用于 .NET 的 Amazon SDK 支持名为 SDK 商店的其他来源。有关 SDK 或工具独有的提供者的更多信息，请参阅您正在使用的 SDK 或工具的特定指南。

顺序决定哪些方法优先使用并覆盖其他方法。例如，如果您在共享config文件中设置了配置文件，则只有在 SDK 或工具先检查其他位置之后，才能找到并使用该配置文件。这意味着，如果您在credentials文件中添加了设置，则会使用该设置而不是config文件中的设置。如果您使用设置和值配置环境变量，它将覆盖credentials和config文件中的该设置。最后，单个操作（Amazon CLI 命令行参数或 API 参数）或代码中的设置将覆盖该命令的所有其他值。

了解本指南的设置页面

本指南中设置参考部分的相关页面详细介绍了可以通过各种机制设定的可用设置。下表列出了配置和凭证文件设置、环境变量以及（对于 Java 和 Kotlin SDKs）可以在代码之外用于配置该功能的 JVM 设置。每个列表中链接的每个主题都会指向相应的设置页面。

- Config文件设置列表
- Credentials文件设置列表
- 环境变量列表
- JVM 系统属性列表

每个凭证提供者或功能都有一个页面，其中列出了用于配置该功能的设置。每个设置的值通常可以通过将该设置添加到配置文件中、设置环境变量或者（仅适用于 Java 和 Kotlin）设置 JVM 系统属性来进行设置。每个设置都会列出所有受支持的方法，用来在描述详细信息上方的块中设置值。尽管不同设置方法的优先级各不相同，但最终功能都是一样的。

描述中将包括默认值（如果有），如果您不执行任何操作，系统将会使用该值。描述中还定义了该设置的有效值。

例如，下面是 请求压缩 功能页面的设置示例。

`disable_request_compression` 示例设置的信息记录了以下内容：

- 有三种等效的方法可以用来在代码库之外控制请求压缩。您可以：
  - 使用 `disable_request_compression` 在配置文件中进行设置
  - 使用 `AWS_DISABLE_REQUEST_COMPRESSION` 将其设置为环境变量
  - 如果您使用的是 Java 或 Kotlin SDK，则还可以使用 `aws.disableRequestCompression` 将其设置为 JVM 系统属性

> **Note**
>
> 可能还有一种方法可以直接在代码中配置相同的功能，但由于这种方法因 SDK 而异，因此本参考未作介绍。如果需要在代码中直接设置配置，请参阅具体的 SDK 指南或 API 参考。

- 如果您不执行任何操作，则该值将默认为 `false`。
- 此布尔值设置的唯一有效值是 `true` 和 `false`。

每个功能页面的底部都有一个 Support b y Amazon SDKs 和 tools 表。

该表会显示您的 SDK 是否支持该页面上列出的设置。`Supported` 列指示支持级别，具有下列值：

- `Yes`：该 SDK 完全支持描述的设置。
- `Partial`：支持部分设置，或者行为与描述有所不同。对于 `Partial`，会用一条附加注释来说明偏差。
- `No`：不支持任何设置。这并不能说明代码中是否可以实现相同的功能；仅指示不支持列出的外部配置设置。

Config文件设置列表

下表中列出的设置可以在共享 Amazon `config`文件中分配。它们是全球性的，影响到所有人 Amazon Web Services 服务。 SDKs 而且工具还可能支持独特的设置和环境变量。要查看仅受单个 SDK 或工具支持的设置和环境变量，请参阅具体的 SDK 或工具指南。

设置名称	Details
account_id_endpoint_mode	基于账户的终端节点
api_versions	常规配置设置
auth_scheme_preference	身份验证方案
aws_access_key_id	Amazon 访问密钥
aws_account_id	基于账户的终端节点
aws_secret_access_key	Amazon 访问密钥
aws_session_token	Amazon 访问密钥
ca_bundle	常规配置设置
credential_process	进程凭证提供者
credential_source	代入角色凭证提供者
defaults_mode	智能配置默认值
disable_host_prefix_injection	主机前缀注入
disable_request_compression	请求压缩

设置名称	Details	
duration_seconds	代入角色凭证提供者	
ec2_metadata_service_endpoint	IMDS 凭证提供者	
ec2_metadata_service_endpoint_mode	IMDS 凭证提供者	
ec2_metadata_v1_disabled	IMDS 凭证提供者	
endpoint_discovery_enabled	端点发现	
endpoint_url	特定于服务的端点	
external_id	代入角色凭证提供者	
ignore_configured_endpoint_urls	特定于服务的端点	
max_attempts	重试行为	
metadata_service_num_attempts	Amazon EC2 实例元数据	
metadata_service_timeout	Amazon EC2 实例元数据	

设置名称	Details
mfa_serial	代入角色凭证提供者
output	常规配置设置
parameter_validation	常规配置设置
region	Amazon Web Services 区域
request_checksum_calculation	Amazon S3 数据完整性保护
request_min_compression_size_bytes	请求压缩
response_checksum_validation	Amazon S3 数据完整性保护
retry_mode	重试行为
role_arn	代入角色凭证提供者
role_session_name	代入角色凭证提供者
s3_disable_express_session_auth	S3 Express One Zone 会话身份验证
s3_disable_multiregion_access_points	Amazon S3 多区域访问点

设置名称	Details
s3_use_arn_region	Amazon S3 接入点
sdk_ua_app_id	应用程序 ID
sigv4a_signing_region_set	身份验证方案
source_profile	代入角色凭证提供者
sso_account_id	IAM Identity Center 凭证提供者
sso_region	IAM Identity Center 凭证提供者
sso_registration_scopes	IAM Identity Center 凭证提供者
sso_role_name	IAM Identity Center 凭证提供者
sso_start_url	IAM Identity Center 凭证提供者
sts_regional_endpoints	Amazon STS 区域端点
use_dualstack_endpoint	双堆栈和 FIPS 端点
use_fips_endpoint	双堆栈和 FIPS 端点
web_identity_token_file	代入角色凭证提供者

Credentials文件设置列表

下表中列出的设置可以在共享 Amazon credentials文件中分配。它们是全球性的，影响到所有人 Amazon Web Services 服务。 SDKs 而且工具还可能支持独特的设置和环境变量。要查看仅受单个 SDK 或工具支持的设置和环境变量，请参阅具体的 SDK 或工具指南。

设置名称	Details	
aws_access_key_id	Amazon 访问密钥	
aws_secret_access_key	Amazon 访问密钥	
aws_session_token	Amazon 访问密钥	

环境变量列表

下表列出了 SDKs 大多数支持的环境变量。它们是全球性的，影响到所有人 Amazon Web Services 服务。 SDKs 而且工具还可能支持独特的设置和环境变量。要查看仅受单个 SDK 或工具支持的设置和环境变量，请参阅具体的 SDK 或工具指南。

设置名称	Details	
AWS_ACCESS_KEY_ID	Amazon 访问密钥	
AWS_ACCOUNT_ID	基于账户的终端节点	
AWS_ACCOUNT_ID_ENDPOINT_MODE	基于账户的终端节点	
AWS_AUTH_SCHEME_PREFERENCE	身份验证方案	

设置名称	Details	
AWS_CA_BUNDLE	常规配置设置	
AWS_CONFIG_FILE	查找和更改共享credentials 文件configAmazon SDKs 和工具的位置	
AWS_CONTAINER_AUTHORIZATION_TOKEN	容器凭证提供者	
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE	容器凭证提供者	
AWS_CONTAINER_CREDENTIALS_FULL_URI	容器凭证提供者	
AWS_CONTAINER_CREDENTIALS_RELATIVE_URI	容器凭证提供者	
AWS_DEFAULTS_MODE	智能配置默认值	
AWS_DISABLE_HOST_PREFIX_INJECTION	主机前缀注入	
AWS_DISABLE_REQUEST_COMPRESSION	请求压缩	

设置名称	Details	
AWS_EC2_METADATA_DISABLED	IMDS 凭证提供者	
AWS_EC2_METADATA_SERVICE_ENDPOINT	IMDS 凭证提供者	
AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE	IMDS 凭证提供者	
AWS_EC2_METADATA_V1_DISABLED	IMDS 凭证提供者	
AWS_ENABLE_ENDPOINT_DISCOVERY	端点发现	
AWS_ENDPOINT_URL	特定于服务的端点	
AWS_ENDPOINT_URL_<SERVICE>	特定于服务的端点	
AWS_IGNORE_CONFIGURED_ENDPOINT_URLS	特定于服务的端点	
AWS_MAX_ATTEMPTS	重试行为	

设置名称	Details	
AWS_METADATA_SERVICE_NUM_ATTEMPTS	Amazon EC2 实例元数据	
AWS_METADATA_SERVICE_TIMEOUT	Amazon EC2 实例元数据	
AWS_PROFILE	使用共享config和credentials 文件进行全局配置 Amazon SDKs 和工具	
AWS_REGION	Amazon Web Services 区域	
AWS_REQUEST_CHECKSUM_CALCULATION	Amazon S3 数据完整性保护	
AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES	请求压缩	
AWS_RESPONSE_CHECKSUM_VALIDATION	Amazon S3 数据完整性保护	
AWS_RETRY_MODE	重试行为	
AWS_ROLE_ARN	代入角色凭证提供者	
AWS_ROLE_SESSION_NAME	代入角色凭证提供者	

设置名称	Details
AWS_S3_DISABLE_EXPRESS_SESSION_AUTH	S3 Express One Zone 会话身份验证
AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS	Amazon S3 多区域访问点
AWS_S3_USE_ARN_REGION	Amazon S3 接入点
AWS_SDK_UA_APP_ID	应用程序 ID
AWS_SECRET_ACCESS_KEY	Amazon 访问密钥
AWS_SESSION_TOKEN	Amazon 访问密钥
AWS_SHARED_CREDENTIALS_FILE	查找和更改共享credentials 文件configAmazon SDKs 和工具的位置
AWS_SIGV4A_SIGNING_REGION_SET	身份验证方案
AWS_STS_REGIONAL_ENDPOINTS	Amazon STS 区域端点
AWS_USE_DUALSTACK_ENDPOINT	双堆栈和 FIPS 端点

设置名称	Details
AWS_USE_FIPS_ENDPOINT	双堆栈和 FIPS 端点
AWS_WEB_IDENTITY_TOKEN_FILE	代入角色凭证提供者

JVM 系统属性列表

您可以将以下 JVM 系统属性用于 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK（以 JVM 为目标）。有关如何设置 JVM 系统属性的说明，请参阅the section called “如何设置 JVM 系统属性”。

设置名称	Details
aws.accessKeyId	Amazon 访问密钥
aws.accountId	基于账户的终端节点
aws.accountIdEndpointMode	基于账户的终端节点
aws.authSchemePreference	身份验证方案
aws.configFile	查找和更改共享credentials 文件configAmazon SDKs 和工具的位置
aws.defaultsMode	智能配置默认值
aws.disableEc2MetadataV1	IMDS 凭证提供者

设置名称	Details	
aws.disableHostPrefixInjection	主机前缀注入	
aws.disableRequestCompression	请求压缩	
aws.disableS3ExpressAuth	S3 Express One Zone 会话身份验证	
aws.ec2MetadataServiceEndpoint	IMDS 凭证提供者	
aws.ec2MetadataServiceEndpointMode	IMDS 凭证提供者	
aws.endpointDiscoveryEnabled	端点发现	
aws.endpointUrl	特定于服务的端点	
aws.endpointUrl<ServiceName>	特定于服务的端点	
aws.ignoreConfiguredEndpointUrls	特定于服务的端点	
aws.maxAttempts	重试行为	

设置名称	Details	
aws.profile	使用共享config和credentials 文件进行全局配置 Amazon SDKs 和工具	
aws.region	Amazon Web Services 区域	
aws.requestChecksumCalculation	Amazon S3 数据完整性保护	
aws.requestMinCompressionSizeBytes	请求压缩	
aws.responseChecksumValidation	Amazon S3 数据完整性保护	
aws.retryMode	重试行为	
aws.roleArn	代入角色凭证提供者	
aws.roleSessionName	代入角色凭证提供者	
aws.s3DisableMultiRegionAccessPoints	Amazon S3 多区域访问点	
aws.s3UseArnRegion	Amazon S3 接入点	
aws.secretAccessKey	Amazon 访问密钥	

设置名称	Details
aws.sessionToken	Amazon 访问密钥
aws.sharedCredentialsFile	查找和更改共享credentials 文件configAmazon SDKs 和工具的位置
aws.useDualstackEndpoint	双堆栈和 FIPS 端点
aws.useFipsEndpoint	双堆栈和 FIPS 端点
aws.webIdentityTokenFile	代入角色凭证提供者
sdk.ua.appId	应用程序 ID

Amazon SDKs 和 Tools 标准化凭证提供商

许多凭证提供商已标准化为一致的默认值，并且在许多 SDKs证书提供商中都以相同的方式工作。这种一致性可以提高跨多个编码时的生产力和清晰度 SDKs。所有设置都可以在代码中被覆盖。有关详细信息，请参阅您的特定 SDK API。

> **Important**
>
> 并非所有提供商都 SDKs 支持所有提供商，甚至支持提供商内部的所有方面。

主题

- 了解默认凭证提供者链
- 特定于 SDK 和工具的凭证提供者链
- Amazon 访问密钥

- 登录凭证提供商
- 代入角色凭证提供者
- 容器凭证提供者
- IAM Identity Center 凭证提供者
- IMDS 凭证提供者
- 进程凭证提供者

了解默认凭证提供者链

所有 SDKs 人都有一系列地点（或来源）供他们检查，以便找到用于向某人提出请求的有效凭证 Amazon Web Services 服务。找到有效凭证后，搜索即告停止。这种系统性搜索称为凭证提供程序链。

使用其中一个标准化凭证提供商时， Amazon SDKs 始终尝试在证书到期时自动续订证书。无论凭证提供者链中使用哪种提供者，内置的凭证提供者链都会确保应用程序能够刷新您的凭证。SDK 无需额外的代码即可完成此操作。

尽管每个 SDK 使用的链各不相同，但它们通常包括以下来源：

凭证提供者	说明
Amazon 访问密钥	Amazon IAM 用户的访问密钥（例如`AWS_ACCESS_KEY_ID` 、和`AWS_SECRET_ACCESS_KEY` ）。
使用 Web 身份或 OpenID Connect 进行联合 - 承担角色凭证提供者	使用知名的外部身份提供者（IdP）（例如，Login with Amazon、Facebook、Google 或任何其他 OpenID Connect (OIDC) 兼容的 IdP）登录。使用 () 中的 JSON 网络令牌 (JWT) 假设 IAM 角色的 Amazon Security Token Service 权限。Amazon STS
登录凭证提供商	获取您已登录的新控制台或现有控制台会话的凭证。
IAM Identity Center 凭证提供者	从中获取凭证 Amazon IAM Identity Center。
代入角色凭证提供者	具有 IAM 角色的权限即可访问其他资源。（检索角色的临时凭证，然后使用该凭证）。

凭证提供者	说明
容器凭证提供者	Amazon Elastic Container Service（Amazon ECS）和 Amazon Elastic Kubernetes Service（Amazon EKS）凭证。容器凭证提供者为客户的容器化应用程序获取凭证。
进程凭证提供者	自定义凭证提供者。从外部来源或流程（包括 IAM Roles Anywhere）获取您的凭证。
IMDS 凭证提供者	Amazon Elastic Compute Cloud（Amazon EC2）实例配置文件凭证。将 IAM 角色与您的每个 EC2 实例相关联。在该实例上运行的代码就可以使用该角色的临时凭证。凭证通过 Amazon EC2 元数据服务提供。

对于链中的每个步骤，都有多种分配设置值的方法。在代码中指定的设置值始终优先。但是，还有 环境变量 和 使用共享config和credentials文件进行全局配置 Amazon SDKs 和工具。有关更多信息，请参阅 设置的优先级。

特定于 SDK 和工具的凭证提供者链

要直接访问 SDK 或工具的特定凭证提供者链详细信息，请从以下列表中选择您的 SDK 或工具：

- Amazon CLI
- 适用于 C++ 的 SDK
- 适用于 Go 的 SDK
- 适用于 Java 的 SDK
- 适用于 JavaScript
- 适用于 Kotlin 的 SDK
- 适用于 .NET 的 SDK
- 适用于 PHP 的 SDK
- 适用于 Python (Boto3) 的 SDK
- 适用于 Ruby 的 SDK
- 适用于 Rust 的 SDK
- 适用于 Swift 的 SDK

- 用于 PowerShell

Amazon 访问密钥

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

Amazon IAM 用户的访问密钥可用作您的 Amazon 证书。 Amazon SDK 会自动使用这些 Amazon 凭据签署 API 请求 Amazon，以便您的工作负载可以安全、方便地访问您的 Amazon 资源和数据。建议始终使用`aws_session_token`，这样凭证才是临时的，过期后不再有效。不建议使用长期凭证。

> **Note**
>
> 如果无法刷新这些临时证书， Amazon 则 Amazon 可能会延长证书的有效期，这样您的工作负载就不会受到影响。

共享 Amazon `credentials`文件是存储凭据信息的推荐位置，因为它安全地位于应用程序源目录之外，并且与共享`config`文件的 SDK 特定设置是分开的。

要了解有关 Amazon 证书和使用访问密钥的更多信息，请参阅 IAM 用户指南中的Amazon 安全证书和管理 IAM 用户的访问密钥。

使用以下方法配置此功能：

`aws_access_key_id`-共享 Amazon **`config`**文件设置, **`aws_access_key_id`**-共享 Amazon **`credentials`**文件设置（推荐方法）, **`AWS_ACCESS_KEY_ID`** - 环境变量, **`aws.accessKeyId`**-JVM 系统属性：仅限 Java/Kotlin

指定作为证书一部分用于对用户进行身份验证的 Amazon 访问密钥。

`aws_secret_access_key`-共享 Amazon **`config`**文件设置, **`aws_secret_access_key`**-共享 Amazon **`credentials`**文件设置（推荐方法）, **`AWS_SECRET_ACCESS_KEY`** - 环境变量, **`aws.secretAccessKey`**-JVM 系统属性：仅限 Java/Kotlin

指定用作验证用户身份的凭证一部分的 Amazon 密钥。

aws_session_token-共享 Amazon **config**文件设置, **aws_session_token**-共享 Amazon **credentials**文件设置（推荐方法）, **AWS_SESSION_TOKEN** - 环境变量, **aws.sessionToken**-JVM 系统属性：仅限 Java/Kotlin

> 指定一个 Amazon 会话令牌，该令牌用作对用户进行身份验证的凭证的一部分。您会收到此值作为成功请求承担角色所返回的临时凭证的一部分。只有在手动指定临时安全凭证时才需要会话令牌。但是，我们建议您始终使用临时安全凭证代替长期凭证。有关安全建议，请参阅 IAM 中的安全最佳实践。

有关如何获取这些值的说明，请参阅 使用短期凭证进行身份验证 Amazon SDKs 和工具。

在config或credentials文件中设置这些必需值的示例：

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token = AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export
 AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
setx
 AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	不支持共享的config文件。
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 config 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	是	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	不支持环境变量。

登录凭证提供商

您可以使用现有的 Amazon 管理控制台登录凭证来获取可用于编程访问的短期证书。完成基于浏览器的身份验证流程后， Amazon 会生成适用于 PowerShell 本地开发工具（例如 CL Amazon I、 Amazon 和的工具）的临时证书。 Amazon SDKs

要生成这些证书，请在 Amazon CLI 中运行`aws login`命令，或者在 “ Amazon 工具” 中运行 `Invoke-AWSLogin` cmdlet。 PowerShell生成的短期证书将在本地缓存，供其重复使用。 Amazon SDKs短期证书将在 15 分钟后过期，但是 CLI SDKs 会根据需要自动刷新证书，最长可达 12 小时。刷新令牌到期后，系统将提示您通过 CLI 或重新登录 PowerShell。

登录命令将使用该设置更新您指定的配置文件，该`login_session`设置存储您在登录工作流程中选择的管理控制台会话的身份。

```
[profile console]
login_session = arn:aws:iam::0123456789012:user/username
region = us-west-2
```

默认情况下，短期凭证和刷新令牌存储在 Linux 和 macOS 或 `%USERPROFILE%\.aws\login\cache` Windows 上的`~/.aws/login/cache`目录下的 JSON 文件中。文件名基于登录会话名称。您可以通过设置`AWS_LOGIN_CACHE_DIRECTORY`环境变量来覆盖该目录。

登录提供商设置

使用以下方法配置此功能：

`AWS_LOGIN_CACHE_DIRECTORY` - 环境变量

备用目录，CLI 和 SDKs 将在其中存储映射到登录会话配置文件的缓存凭据。

默认值：`~/.aws/login/cache`在 Linux 和 macOS 上，或者`%USERPROFILE%\.aws\login\cache`在 Windows 上。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	否	
适用于 Go 1.x (V1) 的 SDK	是	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	需要 CRT
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	否	

代入角色凭证提供者

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

假设角色涉及使用一组临时安全凭证来访问您原本无法访问的 Amazon 资源。这些临时凭证由访问密钥 ID、秘密访问密钥和安全令牌组成。

要设置您的 SDK 或工具来代入角色，必须先创建或标识要代入的特定角色。IAM 角色由角色 Amazon 资源名称（ARN）进行唯一标识。角色与另一个实体建立信任关系。使用该角色的可信实体可能是另一个 Amazon Web Services 服务、Web 身份提供商 Amazon Web Services 账户、OIDC 或 SAML 联合体。

标识 IAM 角色后，如果您受到该角色的信任，则可以将您的 SDK 或工具配置为使用该角色授予的权限。要执行此操作，请使用以下设置。

有关开始使用这些设置的指导，请参阅本指南中的 假设一个拥有身份验证 Amazon 凭证 Amazon SDKs 和工具的角色。

代入角色凭证提供者设置

使用以下方法配置此功能：

`credential_source`-共享 Amazon **`config`**文件设置

在 Amazon EC2 实例或 Amazon Elastic Container Service 容器中使用，指定 SDK 或工具在何处可以查找授权用于代入通过 `role_arn` 参数指定的角色的凭证。

默认值： 无

有效值：

- 环境 – 指定 SDK 或工具从环境变量 `AWS_ACCESS_KEY_ID` 和 `AWS_SECRET_ACCESS_KEY` 检索源凭证。
- Ec@@ 2 InstanceMetadata — 指定软件开发工具包或工具将使用附加到 EC2 实例配置文件的 IAM 角色来获取源证书。
- EcsContainer— 指定软件开发工具包或工具将使用附加到 Amazon ECS 容器的 IAM 角色或附加到 Ama zon EKS 容器的 IAM 角色来获取源证书。

不能在同一配置文件中同时指定 credential_source 和 source_profile。

在 config 文件中设置此项以表明凭证应来自 Amazon EC2 的示例：

```
credential_source = Ec2InstanceMetadata
role_arn = arn:aws:iam::123456789012:role/my-role-name
```

duration_seconds-共享 Amazon **config**文件设置

指定角色会话的最大持续时间（以秒为单位）。

仅当配置文件指定代入角色时，此设置才适用。

默认值： 3600 秒 (1 小时)

有效值：该值的范围在 900 秒（15 分钟）到角色配置的最大会话持续时间（43200 秒或 12 小时）之间。有关更多信息，请参阅 IAM 用户指南中的 查看角色的最大会话持续时间设置。

在 config 文件中设置此项的示例：

```
duration_seconds = 43200
```

external_id-共享 Amazon **config**文件设置

指定第三方用于在其客户账户中代入角色的唯一标识符。

仅当配置文件指定代入角色且该角色的信任策略需要 ExternalId 值时，此设置才适用。该值映射到配置文件指定角色时传递给 AssumeRole 操作的 ExternalId 参数。

默认值：无。

有效值：请参阅 IAM 用户指南中的如何在向第三方授予对您的 Amazon 资源的访问权限时使用外部 ID。

在 config 文件中设置此项的示例：

```
external_id = unique_value_assigned_by_3rd_party
```

mfa_serial-共享 Amazon **config**文件设置

指定用户在代入角色时必须使用的多重身份验证（MFA）设备的标识或序列号。

代入角色时，如果该角色的信任策略包含需要 MFA 身份验证的条件，则此项为必需项。有关 MFA 的更多信息，请参阅《IAM 用户指南》中的 IAM 中的Amazon 多重身份验证。

默认值：无。

有效值： 该值可以是硬件设备（例如 GAHT12345678）的序列号，也可以是虚拟 MFA 设备的 Amazon 资源名称（ARN）。ARN 的格式为 `arn:aws:iam::account-id:mfa/mfa-device-name`

在 config 文件中设置此项的示例：

此示例假设已为该账户创建并为某个用户启用了一个名为 MyMFADevice 的虚拟 MFA 设备。

```
mfa_serial = arn:aws:iam::123456789012:mfa/MyMFADevice
```

role_arn-共享 Amazon **config**文件设置, **AWS_ROLE_ARN** - 环境变量, **aws.roleArn**-JVM 系统属性：仅限 Java/Kotlin

指定要用于执行使用此配置文件请求操作的 IAM 角色的 Amazon 资源名称（ARN）。

默认值：无。

有效值：该值必须是 IAM 角色的 ARN，格式如下：`arn:aws:iam::account-id:role/role-name`

此外，您还必须指定以下设置之一：

- source_profile - 标识另一个配置文件，用于查找具有在此配置文件中代入该角色的权限的凭证。
- credential_source - 使用由当前环境变量标识的凭证或附加到 Amazon EC2 实例配置文件或 Amazon ECS 容器实例的凭证。
- web_identity_token_file - 为已在移动或 Web 应用程序中进行身份验证的用户使用公共身份提供者或任何 OpenID Connect（OIDC）兼容身份提供者。

role_session_name-共享 Amazon **config**文件设置, **AWS_ROLE_SESSION_NAME** - 环境变量, **aws.roleSessionName**-JVM 系统属性：仅限 Java/Kotlin

指定要附加到角色会话的名称。此名称显示在与此会话关联的条目的 Amazon CloudTrail 日志中，该会话可能在审核时有用。有关详细信息，请参阅《CloudTrail用户指南》中的 “Amazon CloudTrail 用户身份” 元素。

默认值：可选参数。如果未提供此值，只要配置文件代入角色，则将自动生成会话名称。

有效值：当 Amazon CLI 或 Amazon API 代表您调用AssumeRole操作（或操作等AssumeRoleWithWebIdentity操作）时，为RoleSessionName参数提供。该值成为您可以查询的代入角色用户 Amazon 资源名称 (ARN) 的一部分，并作为该配置文件调用的操作的 CloudTrail 日志条目的一部分显示。

arn:aws:sts::*123456789012*:assumed-role/*my-role-name*/***my-role_session_name***.

在 config 文件中设置此项的示例：

```
role_session_name = my-role-session-name
```

source_profile-共享 Amazon **config**文件设置

指定其他配置文件，其凭证用于代入由原始配置文件中的 role_arn 设置指定的角色。要了解如何在共享credentials文件 Amazon config和文件中使用配置文件，请参阅共享 config 文件和 credentials 文件。

如果您指定的配置文件也是代入角色配置文件，则将按顺序代入每个角色以完全解析凭证。当 SDK 遇到带有凭证的配置文件时，此链将会停止。角色链将您的 Amazon CLI 或 Amazon API 角色会话限制为最长一小时，并且无法延长。有关更多信息，请参阅 IAM 用户指南中的角色术语和概念。

默认值：无。

有效值：由 config 和 credentials 文件中定义的配置文件的名称组成的文本字符串。还必须在当前配置文件中指定 role_arn 的值。

不能在同一配置文件中同时指定 credential_source 和 source_profile。

在配置文件中设置此项的示例：

```
[profile A]
source_profile = B
role_arn =  arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_process = ./aws_signing_helper credential-process --certificate /
path/to/certificate --private-key /path/to/private-key --trust-anchor-
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-arn
 arn:aws:iam::account:role/ROLE_ID
```

在上例中，A 配置文件告诉 SDK 或工具自动查找关联的 B 配置文件的凭证。在此例中，B 配置文件使用 使用 IAM Anywhere 角色进行身份验证 Amazon SDKs 和工具 提供的凭证助手来获取 Amazon SDK 的凭证。然后，代码会使用这些临时凭证来访问 Amazon 资源。指定的角色必须附加允许运行所请求代码的 IAM 权限策略，例如命令 Amazon Web Services 服务、或 API 方法。配置文件执行的每项操作的 CloudTrail 日志中A都包含角色会话名称。

对于第二个角色链示例，如果您在 Amazon Elastic Compute Cloud 实例上有一个应用程序，并且想让该应用程序代入其他角色，则可以使用以下配置。

```
[profile A]
source_profile = B
role_arn =  arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_source=Ec2InstanceMetadata
```

配置文件 A 将使用来自 Amazon EC2 实例的凭证来代入指定角色，并且会自动续订凭证。

web_identity_token_file-共享 Amazon **config**文件设置, **AWS_WEB_IDENTITY_TOKEN_FILE** - 环境变量, **aws.webIdentityTokenFile**-JVM 系统属性：仅限 Java/Kotlin

指定文件路径，该文件包含来自支持的 OAuth 2.0 提供商或 OpenID Connect ID 身份提供商的访问令牌。

此设置允许使用 Web 身份联合验证提供者（例如 Google、Facebook 和 Amazon 等）进行身份验证。SDK 或开发人员工具加载此文件的内容，并在代表您调用 `AssumeRoleWithWebIdentity` 操作时将其作为 `WebIdentityToken` 参数传递。

默认值：无。

有效值：此值必须是路径和文件名。该文件必须包含身份提供商向您提供的 OAuth 2.0 访问令牌或 OpenID Connect 令牌。相对路径被视为相对于进程工作目录的相对路径。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	部分	`credential_source` 不支持。`duration_seconds` 不支持。`mfa_serial` 不支持。
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	部分	不支持 `mfa_serial` 。不支持 `duration_seconds` 。
适用于 Java 1.x 的 SDK	部分	不支持 `credential_source` 。不支持 `mfa_serial` 。不支持 JVM 系统属性。
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	部分	`credential_source` 不支持。
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

容器凭证提供者

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

容器凭证提供者会为客户的容器化应用程序获取凭证。该证书提供者对亚马逊弹性容器服务 (Amazon ECS) 和亚马逊 Elastic Kubernetes Service (Amazon EKS) 客户非常有用。 SDKs 尝试通过 GET 请求从指定的 HTTP 端点加载凭证。

如果您使用 Amazon ECS，我们建议您使用任务 IAM 角色来改进凭证隔离、授权和提高可审计性。配置后，Amazon ECS 会设置 SDKs和工具用来获取凭证的`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`环境变量。要配置 Amazon ECS 以使用此功能，请参阅《Amazon Elastic Container Service 开发人员指南》中的任务 IAM 角色。

如果您使用 Amazon EKS，我们建议您使用 Amazon EKS 容器组身份来改进凭证隔离，提高最低权限、可审计性，改善独立操作、可重用性和可扩展性。您的容器组（pod）和 IAM 角色都与 Kubernetes 服务账户相关联，以管理应用程序的证书。要了解有关 Amazon EKS 容器组身份的更多信息，请参阅《Amazon EKS 用户指南》中的 Amazon EKS 容器组身份。配置后，Amazon EKS 会设置`AWS_CONTAINER_CREDENTIALS_FULL_URI` SDKs和工具用来获取凭证的和`AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`环境变量。有关设置信息，请参阅 Amazon EKS 用户指南中的设置 Amazon EKS Pod 身份代理，或者在 Amazon 博客网站上的 A mazon EKS Pod Identity 简化了 Amazon EKS 集群上应用程序的 IAM 权限。

使用以下方法配置此功能：

AWS_CONTAINER_CREDENTIALS_FULL_URI - 环境变量

指定完整的 HTTP URL 端点，供 SDK 在请求凭证时使用。这包括方案和主机。

默认值：无。

有效值：有效的 URI。

注意：此设置是 *AWS_CONTAINER_CREDENTIALS_RELATIVE_URI* 的替代设置，只有在未设置 *AWS_CONTAINER_CREDENTIALS_RELATIVE_URI* 时才会使用。

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credentials
```

或者

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost:8080/get-credentials
```

AWS_CONTAINER_CREDENTIALS_RELATIVE_URI - 环境变量

指定 HTTP URL 端点，供 SDK 在请求凭证时使用。该值将附加到默认的 Amazon ECS 的主机名 `169.254.170.2` 上。

默认值：无。

有效值：有效的相对 URI。

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_CONTAINER_CREDENTIALS_RELATIVE_URI=/get-credentials?a=1
```

AWS_CONTAINER_AUTHORIZATION_TOKEN - 环境变量

指定纯文本的授权令牌。如果设置了此变量，SDK 将使用环境变量的值在 HTTP 请求上设置授权标头。

默认值：无。

有效值：字符串。

注意：此设置是 *AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE* 的替代设置，只有在未设置 *AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE* 时才会使用。

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN=Basic abcd
```

AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE - 环境变量

指定至包含纯文本授权令牌的文件的绝对文件路径。

默认值：无。

有效值：字符串。

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE=/path/to/token
```

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	
适用于 Java 2.x 的 SDK	是	当 Lambda SnapStart 被激活AWS_CONTAINER_CREDENTIALS_FULL_URI 并自动AWS_CONTAINER_AUTHORIZATION_TOKEN 用于身份验证时。

SDK	支持	备注或更多信息
适用于 Java 1.x 的 SDK	是	当 Lambda SnapStart 被激活AWS_CONTAINER_CREDENTIALS_FULL_URI 并自动AWS_CONTAINER_AUTHORIZATION_TOKEN 用于身份验证时。
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	当 Lambda SnapStart 被激活AWS_CONTAINER_CREDENTIALS_FULL_URI 并自动AWS_CONTAINER_AUTHORIZATION_TOKEN 用于身份验证时。
适用于 .NET 3.x 的 SDK	是	当 Lambda SnapStart 被激活AWS_CONTAINER_CREDENTIALS_FULL_URI 并自动AWS_CONTAINER_AUTHORIZATION_TOKEN 用于身份验证时。
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	当 Lambda SnapStart 被激活AWS_CONTAINER_CREDENTIALS_FULL_URI 并自动AWS_CONTAINER_AUTHORIZATION_TOKEN 用于身份验证时。
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

IAM Identity Center 凭证提供者

> **Note**
>
> 要帮助理解设置页面的布局或解释后面的 Su pport by Amazon SDK 和工具表，请参阅了解本指南的设置页面。

此身份验证机制 Amazon IAM Identity Center 用于获取您的代码的单点登录 (SSO) 访问 Amazon Web Services 服务 权限。

> **Note**
>
> 在 Amazon SDK API 文档中，IAM 身份中心凭证提供商被称为 SSO 凭证提供商。

启用 IAM Identity Center 后，您可以在共享 Amazon `config`文件中为其设置定义配置文件。此配置文件用于连接到 IAM Identity Center 访问门户。当用户成功通过 IAM Identity Center 进行身份验证后，门户将返回与该用户关联的 IAM 角色的短期凭证。要了解 SDK 如何从配置中获取临时证书并将其用于 Amazon Web Services 服务 请求，请参阅如何解决 Amazon SDKs 和工具的 IAM 身份中心身份验证问题。

通过 `config` 文件配置 IAM Identity Center 有两种方式：

- （推荐）SSO 令牌提供者配置：延长会话时长。包括对自定义会话时长的支持。
- 旧版不可刷新配置：使用固定的八小时会话。

在这两种配置中，您都需要在会话到期后重新登录。

以下两份指南包含有关 IAM Identity Center 的其他信息：

- Amazon IAM Identity Center 用户指南
- Amazon IAM Identity Center 门户 API 参考

要深入了解 SDK 和工具如何采用此配置以使用和刷新凭证，请参阅 如何解决 Amazon SDKs 和工具的 IAM 身份中心身份验证问题。

先决条件

您必须先启用 IAM Identity Center。有关启用 IAM Identity Center 身份验证的详细信息，请参阅《Amazon IAM Identity Center 用户指南》 中的 Enabling Amazon IAM Identity Center。

> **Note**
>
> 有关本页详细介绍的完整先决条件和所需的共享 `config` 文件配置，请参阅有关设置使用 IAM 身份中心对 Amazon SDK 和工具进行身份验证的指导说明。

SSO 令牌提供商配置

当您使用 SSO 令牌提供程序配置时，您的 Amazon SDK 或工具会自动刷新您的会话，直到延长的会话时段为止。有关会话持续时间和最长持续时间的更多信息，请参阅Amazon IAM Identity Center 用户指南中的配置 Amazon 访问门户和 IAM Identity Center 集成应用程序的会话持续时间。

该`config`文件的`sso-session`部分用于对用于获取 SSO 访问令牌的配置变量进行分组，然后可以使用这些变量来获取 Amazon 凭证。有关 `config` 文件中此节的更多详细信息，请参阅 配置文件的格式。

以下共享 `config` 文件示例使用 `dev` 配置文件将 SDK 或工具配置为请求 IAM Identity Center 凭证。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

> **Note**
>
> 对于`sso_region`，请使用配置您的 IAM 身份中心实例的 Amazon 区域。这可能与您的默认 Amazon 区域不同。您可以在 IAM 身份中心控制台控制台控制面板上找到此值。

上面的示例展示您可以定义 sso-session 节并将其关联到某个配置文件。通常，sso_role_name必须在profile部分中设置sso_account_id和，这样 SDK 才能请求 Amazon 凭证。sso_regionsso_start_url、和，sso_registration_scopes必须在该sso-session部分中设置。

并不是所有 SSO 令牌配置场景都需要 sso_account_id 和 sso_role_name。如果您的应用程序仅使用支持持 Amazon Web Services 服务 有者身份验证的凭证，则不需要传统 Amazon 凭证。持有者身份验证是一种 HTTP 身份验证方案，它使用称为持有者令牌的安全令牌。在这种情况下，不需要 sso_account_id 和 sso_role_name。要确定该服务是否支持不记名令牌授权，请参阅个人 Amazon Web Services 服务 指南。

注册范围配置为 sso-session 的一部分。范围是 OAuth 2.0 中的一种机制，用于限制应用程序对用户账户的申请访问。上例设置了 sso_registration_scopes，来提供列出账户和角色的必要访问权限。

下例展示了如何在多个配置文件中重复使用相同的 sso-session 配置。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[profile prod]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole2

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

身份验证令牌缓存到 ~/.aws/sso/cache 目录下的磁盘上，文件名基于会话名称。

遗留的不可刷新配置

使用遗留的不可刷新配置不支持自动令牌刷新。我们建议改用 SSO 令牌提供商配置。

要使用传统的不可刷新配置，您必须在配置文件中指定以下设置：

- sso_start_url

- sso_region
- sso_account_id
- sso_role_name

可以使用 sso_start_url 和 sso_region 设置为配置文件指定用户门户。可以使用 sso_account_id 和 sso_role_name 设置来指定权限。

以下示例设置了 config 文件中的四个必需值。

```
[profile my-sso-profile]
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_region = us-west-2
sso_account_id = 111122223333
sso_role_name = SSOReadOnlyRole
```

身份验证令牌缓存到 ~/.aws/sso/cache 目录下的磁盘上，文件名基于 sso_start_url。

IAM Identity Center 凭证提供者设置

使用以下方法配置此功能：

sso_start_url-共享 Amazon **config**文件设置

指向您所在组织的 IAM Identity Center 颁发者 URL 或访问门户 URL 的 URL。有关详细信息，请参阅《Amazon IAM Identity Center 用户指南》中的 Using the Amazon access portal。

要找到此值，请打开 IAM Identity Center 控制台，查看控制面板，然后找到 Amazon 访问门户 URL。

- 或者，从的 2.22.0 版本开始 Amazon CLI，您可以改用Amazon 发行者 URL 的值。

sso_region-共享 Amazon **config**文件设置

其中 Amazon Web Services 区域 包含您的 IAM 身份中心门户主机；也就是您在启用 IAM 身份中心之前选择的区域。这与您的默认 Amazon 区域无关，也可能有所不同。

有关 Amazon Web Services 区域 及其代码的完整列表，请参阅中的区域终端节点Amazon Web Services 一般参考。要查找此值，请打开 IAM Identity Center 控制台，查看控制 面板，然后查找区域。

sso_account_id-共享 Amazon **config**文件设置

通过 Amazon Organizations 服务添加 Amazon Web Services 账户 的用于身份验证的数字 ID。

要查看可用账户列表，请前往 IAM Identity Center 控制台并打开 Amazon Web Services 账户页面。您还可以在Amazon IAM Identity Center 门户 API 参考中使用 ListAccountsAPI 方法查看可用账户列表。例如，您可以调用 “列表账户” Amazon CLI 方法。

`sso_role_name`-共享 Amazon **`config`**文件设置

以 IAM 角色配置的权限集的名称，用于定义用户生成的权限。角色必须存在于 Amazon Web Services 账户 指定的中`sso_account_id`。使用角色名称，而不是角色的 Amazon 资源名称（ARN）。

权限集附有 IAM policy 和自定义权限策略，并定义了用户对其分配的 Amazon Web Services 账户的访问权限级别。

要查看每个可用权限集的列表 Amazon Web Services 账户，请转到 IAM Identity Center 控制台并打开Amazon Web Services 账户页面。选择 Amazon Web Services 账户 表格中列出的正确权限集名称。您还可以使用 Port Amazon IAM Identity Center al API 参考中的 ListAccountRolesAPI 方法查看可用权限集列表。例如，你可以调用 list-account- Amazon CLI roles 方法。

`sso_registration_scopes`-共享 Amazon **`config`**文件设置

要为 `sso-session` 授权的范围的逗号分隔列表。应用程序可以请求一个或多个范围，向应用程序签发的访问令牌将仅限于授予的范围。要从 IAM Identity Center 服务中取回刷新后的令牌，必须赋予 `sso:account:access` 最小范围。有关可用访问范围选项的列表，请参阅《Amazon IAM Identity Center 用户指南》中的 Access scopes。

这些范围定义了为已注册的 OIDC 客户端请求授权的权限和客户端检索的访问令牌。范围授权对 IAM Identity Center 持有者令牌授权端点的访问。

此设置不适用于遗留的不可刷新配置。使用传统配置发布的令牌被隐式限制在 `sso:account:access` 作用域范围内。

Support by Amazon SDK 和工具

以下 SDK 支持本主题中所述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	

SDK	支持	备注或更多信息
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	`credentials` 文件中也支持配置值。
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	部分	仅限遗留的不可刷新配置。
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

> **Note**
>
> 以下是 Amazon 工具包和 IAM 身份中心身份验证的相关文档：
>
> - VS Code — 连接到 Amazon-涵盖通过起始网址和基于浏览器的身份验证登录 IAM 身份中心。
> - Visual Studio — Visu al Studio 的 Amazon Toolkit Amazon 中的 IAM 身份中心凭证 ——涵盖配置 SSO 配置文件。
> - JetBrains— 将的 Amazon 工具包 JetBrains 连接到您的 Amazon 帐户-介绍如何将的 Amazon 工具包 JetBrains 连接到您的 Amazon 帐户。

IMDS 凭证提供者

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

实例元数据服务（IMDS）提供有关您的实例的数据，您可以用它来配置或管理正在运行的实例。有关可用数据的更多信息，请参阅《Amazon EC2 用户指南》中的使用实例元数据。Amazon EC2 提供了可供实例使用的本地端点，该端点可以为实例提供各种信息。如果实例附加了角色，则它可以提供一组对该角色有效的凭证。 SDKs 可以使用该端点来解析作为其默认凭证提供程序链一部分的证书。默认使用实例元数据服务版本 2 (IMDSv2)，即使用会话令牌的更安全的 IMDS 版本。如果由于不可重试的情况（HTTP 错误代码 403、404、405）而失败，则使用后备方法。 IMDSv1

使用以下方法配置此功能：

`AWS_EC2_METADATA_DISABLED` - 环境变量

是否尝试使用 Amazon EC2 实例元数据服务（IMDS）来获取凭证。

默认值：`false`。

有效值：

- **`true`** – 请勿使用 IMDS 来获取凭证。
- **`false`** – 使用 IMDS 来获取凭证。

ec2_metadata_v1_disabled-共享 Amazon **config**文件设置, **AWS_EC2_METADATA_V1_DISABLED** - 环境变量, **aws.disableEc2MetadataV1**-JVM 系统属性：仅限 Java/Kotlin

如果 IMDSv2 失败，是否使用实例元数据服务版本 1 (IMDSv1) 作为后备方案。

> **Note**
>
> New SDKs 不支持 IMDSv1 ，因此不支持此设置。有关详细信息，请见表 Support by Amazon SDKs and too。

默认值：`false`。

有效值：

- **true**— 请勿 IMDSv1 用作备用。
- **false**— IMDSv1 用作备用。

ec2_metadata_service_endpoint-共享 Amazon **config**文件设置, **AWS_EC2_METADATA_SERVICE_ENDPOINT** - 环境变量, **aws.ec2MetadataServiceEndpoint**-JVM 系统属性：仅限 Java/Kotlin

IMDS 的端点。此值将覆盖软件开发工具包和 Amazon 工具用于搜索 Amazon EC2 实例元数据的默认位置。

默认值：如果 `ec2_metadata_service_endpoint_mode` 等于 `IPv4`，则默认端点为 `http://169.254.169.254`。如果 `ec2_metadata_service_endpoint_mode` 等于 `IPv6`，则默认端点为 `http://[fd00:ec2::254]`。

有效值：有效的 URI。

ec2_metadata_service_endpoint_mode-共享 Amazon **config**文件设置, **AWS_EC2_METADATA_SERVICE_ENDPOINT_MODE** - 环境变量, **aws.ec2MetadataServiceEndpointMode**-JVM 系统属性：仅限 Java/Kotlin

IMDS 的端点模式。

默认值：`IPv4`。

有效值：`IPv4`、`IPv6`。

> **Note**
>
> IMDS 凭证提供者是 了解默认凭证提供者链 的一部分。但是，只有在本系列中的其他几个提供者之后，才会检查 IMDS 凭证提供者。因此，如果您希望您的程序使用此提供者的凭证，则必须从配置中删除其他有效的凭证提供者或使用其他配置文件。或者，与其依赖凭证提供者链自动发现哪个提供者返回了有效凭证，不如在代码中指定使用的 IMDS 凭证提供者。创建服务客户端时，可直接指定凭证来源。

IMDS 凭证的安全性

默认情况下，当 Amazon 软件开发工具包未配置有效凭证时，软件开发工具包将尝试使用 Amazon EC2 实例元数据服务 (IMDS) 来检索 Amazon 角色的证书。通过将 `AWS_EC2_METADATA_DISABLED` 环境变量设置为 `true`，可以禁用此行为。这样可以防止不必要的网络活动，并增强不可信网络的安全性，此类网络可能会模拟 Amazon EC2 实例元数据服务。

> **Note**
>
> Amazon 无论这些设置如何，配置了有效凭证的 SDK 客户端都不会使用 IMDS 检索凭证。

禁用 Amazon EC2 IMDS 凭证

如何设置此环境变量取决于所使用的操作系统以及您是否希望更改保持不变。

Linux 和 macOS

使用 Linux 或 macOS 的客户可以使用以下命令设置此环境变量：

```
$ export AWS_EC2_METADATA_DISABLED=true
```

如果您希望此设置在多个 shell 会话和系统重启中保持不变，则可以将上述命令添加到您的 shell 配置文件中，例如 `.bash_profile`、`.zsh_profile` 或 `.profile`。

Windows

使用 Windows 的客户可以使用以下命令设置此环境变量：

```
$ set AWS_EC2_METADATA_DISABLED=true
```

如果您希望此设置在多个 shell 会话和系统重启中保持不变，则可以改用以下命令：

```
$ setx AWS_EC2_METADATA_DISABLED=true
```

> **Note**
>
> 该 setx 命令不会将该值应用于当前的 shell 会话，因此您需要重新加载或重新打开 shell 才能使更改生效。

Support by Amazon SDKs and too

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	部分	JVM 系统属性：使用 `com.amazonaws.sdk.disableEc2MetadataV1` 而不是 `aws.disableEc2MetadataV1` ；不支持 `aws.ec2MetadataServiceEndpoint` 和 `aws.ec2MetadataServiceEndpointMode` 。
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Kotlin 的 SDK	是	不使用 IMDSv1 后备。
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	不使用 IMDSv1 后备。
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	您可以使用在代码中显式禁用 IMDSv1 回退。`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true`
适用于 PowerShell V4 的工具	是	您可以使用在代码中显式禁用 IMDSv1 回退。`[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = $true`

进程凭证提供者

> **Note**
>
> 要帮助理解设置页面的布局或解释后面的 Su pport by Amazon SDK 和工具表，请参阅了解本指南的设置页面。

SDK 提供了一种扩展自定义用例的凭证提供者链的方法。此提供者可用于提供自定义实现，例如从本地凭证存储中检索凭证或与本地身份提供者集成。

例如，IAM Roles Anywhere 使用 `credential_process` 来代表您的应用程序获取临时凭证。要对此用途配置 `credential_process`，请参阅 使用 IAM Anywhere 角色进行身份验证 Amazon SDKs 和工具。

> **Note**
>
> 下面介绍了一种从外部进程获取凭证的方法，可在您在 Amazon之外运行软件时使用。如果您在 Amazon 计算资源上进行构建，请使用其他凭证提供程序。使用此选项时，应确保按照适用于操作系统的安全最佳实践，尽可能锁定 config 文件。确认您的自定义凭证工具未向其写入任何机密信息`StdErr`，因为软件开发工具包和 Amazon CLI 可以捕获和记录此类信息，从而有可能将其暴露给未经授权的用户。

使用以下方法配置此功能：

`credential_process`-共享 Amazon `config`文件设置

指定 SDK 或工具代表您运行的外部命令，以生成或检索用于该命令的身份验证凭证。该设置指定 SDK program/command 将调用的名称。当 SDK 调用该进程时，它会等待进程将 JSON 数据写入 `stdout`。自定义提供者必须以特定格式返回信息。该信息包含 SDK 或工具可用于对您进行身份验证的凭据。

> **Note**
>
> 进程凭证提供者是 了解默认凭证提供者链 的一部分。但是，只有在本系列中的其他几个提供者之后，才会检查进程凭证提供者。因此，如果您希望您的程序使用此提供者的凭证，则必须从配置中删除其他有效的凭证提供者或使用其他配置文件。或者，与其依赖凭证提供者链自动发现哪个提供者返回了有效凭证，不如在代码中指定使用的进程凭证提供者。创建服务客户端时，可直接指定凭证来源。

指定凭证程序的路径

该设置的值是一个字符串，其中包含指向 SDK 或开发工具代表您运行的程序的路径：

- 路径和文件名只能包含以下字符： A-Z、a-z、0-9、连字符 (-)、下划线 (_)、句点 (.)、正斜杠 (/)、反斜杠 (\) 和空格。
- 如果路径或文件名包含空格，请将完整路径和文件名用双引号 (" ") 括起来。

- 如果参数名称或参数值包含空格，则用双引号 (" ") 将该元素括起来。仅括起来名称或值，而不是名称值对。
- 请勿在字符串中包含任何环境变量。例如，您不能包含 $HOME 或 %USERPROFILE%。
- 不要将主文件夹指定为 ~。* 您必须指定完整路径或基文件名。如果存在基本文件名，则系统会尝试在 PATH 环境变量指定的文件夹中查找该程序。路径因操作系统而异：

  以下示例显示了在共享config文件中设置凭据_process。 Linux/macOS

  ```
  credential_process = "/path/to/credentials.sh" parameterWithoutSpaces "parameter with
   spaces"
  ```

  以下示例显示了在 Windows 上的共享 config 文件中设置 credential_process。

  ```
  credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter
   with spaces"
  ```

- 可以在专属配置文件中指定：

  ```
  [profile cred_process]
  credential_process = /Users/username/process.sh
  region = us-east-1
  ```

凭证计划的有效输出

SDK 按照配置文件中指定的方式运行该命令，然后从标准输出流中读取数据。您指定的命令，无论是脚本还是二进制程序，都必须在 STDOUT 上生成符合以下语法的 JSON 输出。

```
{
    "Version": 1,
    "AccessKeyId": "an AWS access key",
    "SecretAccessKey": "your AWS secret access key",
    "SessionToken": "the AWS session token for temporary credentials",
    "Expiration": "RFC3339 timestamp for when the credentials expire",
    "AccountId": "the AWS account ID associated with these credentials"
}
```

> **Note**
>
> 截至撰写本文之时，`Version` 密钥必须设置为 1。随时间推移和该结构的发展，该值可能会增加。

`Expiration` 密钥是采用 RFC3339 格式的时间戳。如果工具的输出中不存在 `Expiration` 密钥，则 SDK 假定凭证是不刷新的长期凭证。否则，将其视为临时凭证，并通过在其过期前重新运行 `credential_process` 命令来自动刷新凭证。

> **Note**
>
> SDK 不 缓存外部进程凭证，这一点不同于代入角色凭证。如果需要缓存，则必须在外部进程中实现。

> **Note**
>
> `AccountId`是可选的，因此无需凭证即可使用，但是提供凭证可以为支持它的服务优化端点解析。当 SDK 知道凭证属于哪个账户时，某些 Amazon 服务可以使用账户特定的终端节点。例如，Amazon S3 可以将请求路由到账户范围的终端节点，而不是全局终端节点。否`AccountId`则，SDK 必须依靠其他机制来确定账户，否则它将无法使用基于账户的端点路由。

外部进程可以返回非零返回代码，以指示在检索凭证时发生错误。

Support by Amazon SDK 和工具

以下 SDK 支持本主题中所述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	是	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

Amazon SDKs 和 “工具” 标准化功能

许多功能已标准化为一致的默认值，并且在许多功能上都以相同的方式工作 SDKs。这种一致性可以提高跨多个编码时的生产力和清晰度 SDKs。所有设置都可以在代码中被覆盖，请参阅您的特定 SDK API 了解详情。

> **Important**
>
> 并非所有功能都 SDKs 支持所有功能，甚至不是功能中的所有方面。

主题

基于账户的端点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

基于账户的端点通过使用 Amazon Web Services 账户 ID 来为支持此功能的服务路由请求，有助于确保高性能和可扩展性。使用支持基于账户的端点的 Amazon SDK 和服务时，SDK 客户端会构造和使用基于账户的端点，而不是区域性端点。如果 SDK 客户端看不到账户 ID，则该客户端将使用区域性端点。基于账户的终端节点的形式是`https://<account-id>.ddb.<region>.amazonaws.com`，你`<region>`的 Amazon Web Services 账户 ID 在哪里，在哪里`<account-id>`，以及。 Amazon Web Services 区域

使用以下方法配置此功能：

`aws_account_id`-共享 Amazon **`config`**文件设置, **`AWS_ACCOUNT_ID`** - 环境变量, **`aws.accountId`**-JVM 系统属性：仅限 Java/Kotlin

Amazon Web Services 账户 身份证。用于基于账户的端点路由。 Amazon Web Services 账户 ID 的格式类似于 111122223333。

对于某些服务，基于账户的端点路由可提高请求性能。

`account_id_endpoint_mode`-共享 Amazon **`config`**文件设置, **`AWS_ACCOUNT_ID_ENDPOINT_MODE`** - 环境变量, **`aws.accountIdEndpointMode`**-JVM 系统属性：仅限 Java/Kotlin

此设置用于在必要时关闭基于账户的端点路由，并绕过基于账户的规则。

默认值：`preferred`

有效值：

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

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	发布此功能的 SDK 版本	备注或更多信息
Amazon CLI v2	是	2.25.0	
Amazon CLI v1	是	1.38.0	
适用于 C++ 的 SDK	否		
适用于 Go V2 (1.x) 的 SDK	是	v1.35.0	
适用于 Go 1.x (V1) 的 SDK	否		
适用于 Java 2.x 的 SDK	是	v2.28.4	
适用于 Java 1.x 的 SDK	是	v1.12.771	
适用于 JavaScript 3.x 的软件开发工具包	是	v3.656.0	
适用于 JavaScript 2.x 的 SDK	否		
适用于 Kotlin 的 SDK	是	v1.3.37	
适用于.NET 4.x 的 SDK	是	4.0.0	

SDK	支持	发布此功能的 SDK 版本	备注或更多信息
适用于 .NET 3.x 的 SDK	否		
适用于 PHP 3.x 的 SDK	是	v3.318.0	
适用于 Python (Boto3) 的 SDK	是	1.37.0	
适用于 Ruby 3.x 的 SDK	是	v1.123.0	
适用于 Rust 的 SDK	是	发布-2025-04-24	
适用于 Swift 的 SDK	是	1.2.0	
适用于 PowerShell V5 的工具	否		
适用于 PowerShell V4 的工具	否		

应用程序 ID

> **Note**
>
> 要帮助理解设置页面的布局或解释后面的 Su pport by Amazon SDK 和工具表，请参阅了解本指南的设置页面。

一个 Amazon Web Services 账户 可以被多个客户应用程序用来拨打电话 Amazon Web Services 服务。应用程序 ID 为客户提供了一种识别哪个源应用程序使用进行了一组调用的方法 Amazon Web

Services 账户。 Amazon SDK 和服务不会使用或解释此值，除非将其显示在客户通信中。例如，此值可以包含在操作电子邮件中，也可以包含在中， Amazon Health Dashboard 以唯一标识您的哪些应用程序与通知相关联。

使用以下方法配置此功能：

sdk_ua_app_id-共享 Amazon **config**文件设置, **AWS_SDK_UA_APP_ID** - 环境变量, **sdk.ua.appId**-JVM 系统属性：仅限 Java/Kotlin

此设置是您分配给应用程序的唯一字符串，用于标识特定应用程序中的哪些应用程序 Amazon Web Services 账户 正在调用 Amazon。

默认值：None

有效值：字符串，最大长度为 50。允许使用字母、数字和以下特殊字符：!、#、$、%、&、'、*、+、-、.、^、_、`、|、、~

在 config 文件中设置此值的示例：

```
[default]
sdk_ua_app_id=ABCDEF
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_SDK_UA_APP_ID=ABCDEF
export AWS_SDK_UA_APP_ID="ABC DEF"
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_SDK_UA_APP_ID ABCDEF
setx AWS_SDK_UA_APP_ID="ABC DEF"
```

如果包含对所用 Shell 具有特殊含义的符号，请根据需要对该值进行转义。

Support by Amazon SDK 和工具

以下 SDK 支持本主题中所述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	不支持共享的config文件。
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	JVM 系统属性为 aws.userAgentAppId 。
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

Amazon EC2 实例元数据

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon EC2 在实例上提供了一项名为实例元数据服务（IMDS）的服务。要了解有关此服务的更多信息，请参阅《Amazon EC2 用户指南》中的使用实例元数据。尝试在已配置 IAM 角色的 Amazon EC2 实例上检索凭证时，默认情况下，与实例元数据服务的连接是可调节的。

使用以下方法配置此功能：

`metadata_service_num_attempts`-共享 Amazon **`config`**文件设置，
`AWS_METADATA_SERVICE_NUM_ATTEMPTS` - 环境变量

本设置指定了尝试从实例元数据服务检索数据时，在放弃前尝试的总次数。

默认值： 1

有效值：大于或等于 1 的数字。

`metadata_service_timeout`-共享 Amazon **`config`**文件设置，
`AWS_METADATA_SERVICE_TIMEOUT` - 环境变量

指定的从实例元数据服务检索数据时，发生超时前的秒数。

默认值： 1

有效值：大于或等于 1 的数字。

在 `config` 文件中设置这些值的示例：

```
[default]
metadata_service_num_attempts=10
metadata_service_timeout=10
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_METADATA_SERVICE_NUM_ATTEMPTS=10
export AWS_METADATA_SERVICE_TIMEOUT=10
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_METADATA_SERVICE_NUM_ATTEMPTS 10
setx AWS_METADATA_SERVICE_TIMEOUT 10
```

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	否	
适用于 Go V2 (1.x) 的 SDK	否	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	部分	仅支持 `AWS_METADATA_SERVICE_TIMEOUT` 。
适用于 Java 1.x 的 SDK	部分	仅支持 `AWS_METADATA_SERVICE_TIMEOUT` 。
适用于 JavaScript 3.x 的软件开发工具包	否	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	否	
适用于.NET 4.x 的 SDK	否	
适用于 .NET 3.x 的 SDK	否	

SDK	支持	备注或更多信息
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	否	
适用于 Rust 的 SDK	否	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	否	
适用于 PowerShell V4 的工具	否	

Amazon S3 接入点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon S3 服务提供接入点作为与 Amazon S3 存储桶交互的替代方式。接入点上可以应用唯一的策略和配置，而不是直接应用到存储桶。使用 Amazon SDKs，您可以在存储桶字段中使用接入点 Amazon 资源名称 (ARNs) 进行 API 操作，而不必明确指定存储桶名称。它们用于特定的操作，例如使用具有 `GetObject` 的接入点 ARN 从存储桶中获取对象，或者使用具有 `PutObject` 的接入点 ARN 将对象添加到存储桶。

要了解有关 Amazon S3 接入点的更多信息 ARNs，请参阅 Amazon S3 用户指南中的使用接入点。

使用以下方法配置此功能：

s3_use_arn_region-共享 Amazon **config**文件设置, **AWS_S3_USE_ARN_REGION** - 环境变量, **aws.s3UseArnRegion**-JVM 系统属性：仅限 Java/Kotlin，要直接在代码中配置值，请直接查阅您的特定 SDK。

此设置控制 SDK 是否使用接入点 ARN Amazon Web Services 区域 为请求构造区域终端节点。SDK 会验证 Amazon Web Services 区域 ARN 是否由与客户端 Amazon Web Services 区域 配置 Amazon 相同的分区提供服务，以防止最有可能失败的跨分区调用。如果多次定义，则优先使用代码配置的设置，其次是环境变量设置。

默认值：`false`

有效值：

- **`true`**— SDK 在构造终端节点 Amazon Web Services 区域 时使用 ARN，而不是客户端的配置 Amazon Web Services 区域。例外：如果客户端的配置 Amazon Web Services 区域 是 FIPS Amazon Web Services 区域，则它必须与 ARN 相匹配。 Amazon Web Services 区域否则将导致出现错误。
- **`false`** – SDK 在构造端点时使用客户端配置的 Amazon Web Services 区域 。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	是	不支持 JVM 系统属性。

SDK	支持	备注或更多信息
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	不遵循标准优先顺序；共享的 config 文件值优先于环境变量。
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	否	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	是	不遵循标准优先顺序；共享的 config 文件值优先于环境变量。
适用于 PowerShell V4 的工具	是	不遵循标准优先顺序；共享的 config 文件值优先于环境变量。

Amazon S3 多区域访问点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon S3 多区域接入点提供了一种全局端点，应用程序可以使用该端点来满足来自位于多个 Amazon Web Services 区域的 Amazon S3 存储桶的请求。您可以使用多区域接入点构建多区域应用程序，使用单个区域中使用的相同架构，然后在世界任何地方运行这些应用程序。

要了解有关多区域接入点的更多信息，请参阅 Amazon S3 用户指南中的 Amazon S3 中的多区域接入点。

要了解有关多区域接入点 Amazon 资源名称 (ARNs) 的更多信息，请参阅 Amaz on S3 用户指南中的使用多区域接入点发出请求。

要了解有关创建多区域接入点的更多信息，请参阅 Amazon S3 用户指南中的管理多区域接入点。

SigV4A 算法是用于签署全局区域请求的签名实现。该算法由 SDK 通过 Amazon 通用运行时 (CRT) 库上的依赖项来获得。

使用以下方法配置此功能：

`s3_disable_multiregion_access_points`-共享 Amazon **`config`**文件设置，**`AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS`** - 环境变量，**`aws.s3DisableMultiRegionAccessPoints`**-JVM 系统属性：仅限 Java/Kotlin，要直接在代码中配置值，请直接查阅您的特定 SDK。

此设置控制 SDK 是否可能尝试跨区域请求。如果多次定义，则优先使用代码配置的设置，其次是环境变量设置。

默认值：`false`

有效值：

- **`true`** – 停止使用跨区域请求。
- **`false`** – 使用多区域接入点启用跨区域请求。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	

SDK	支持	备注或更多信息
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

S3 Express One Zone 会话身份验证

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

S3 Express One Zone 是 Amazon S3 的高性能存储类别，可为频繁访问的数据提供低至几毫秒的延迟。当您使用 S3 Express One Zone 存储桶 Amazon SDKs 和工具时，会自动使用基于会话的身份验证，该身份验证针对数据请求的低延迟授权进行了优化。您可以将会话令牌与可用区级（对象级）操作结合使用，从而分散为一个会话中的多个请求进行授权有关的延迟，降低身份验证开销并提高整体请求性能。

S3 Express One Zone 存储桶使用包含可用区 ID 的专用命名格式，例如 `bucket-name--usw2-az1--x-s3`。当 SDK 检测到这种命名模式时，将会自动将请求路由到相应的 S3 Express One Zone 端点，并应用优化的身份验证流程。会话身份验证会创建临时的存储桶特定凭证，从而提供对存储桶的低延迟访问，并由 SDK 自动缓存和刷新。要了解更多信息，请参阅《Amazon S3 用户指南》中的 S3 Express One Zone。

S3 Express One Zone 存储桶会默认启用会话身份验证。

使用以下方法配置此功能：

`s3_disable_express_session_auth`-共享 Amazon **`config`**文件设置, **`AWS_S3_DISABLE_EXPRESS_SESSION_AUTH`** - 环境变量, **`aws.disableS3ExpressAuth`**-JVM 系统属性：仅限 Java/Kotlin

控制是否禁用 S3 Express One Zone 会话身份验证。设置为 `true` 时，SDK 将为 S3 Express One Zone 存储桶使用标准的 Sigv4 身份验证，而不是会话身份验证。

默认值：`false`

有效值：

- **`true`**：禁用 S3 Express One Zone 会话身份验证。
- **`false`**：启用 S3 Express One Zone 会话身份验证。

在 `config` 文件中设置此值的示例：

```
[default]
s3_disable_express_session_auth=true
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_S3_DISABLE_EXPRESS_SESSION_AUTH=true
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_S3_DISABLE_EXPRESS_SESSION_AUTH true
```

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
Amazon CLI v1	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x（V1）的 SDK	否	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	

SDK	支持	备注或更多信息
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	JVM 系统属性为 `aws.s3DisableExpressSessionAuth` 。
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

身份验证方案

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon 服务支持多种身份验证方案，例如 Amazon 签名版本 4 (Sigv4) 和 Amazon 签名版本 4a (sigv4a)。默认情况下，根据服务模型定义 SDKs 选择身份验证方案，并优先考虑提供最佳兼容性的方案。但您可以配置根据具体要求进行优化的首选身份身验证方案。

与 SigV4 不同，使用 SigV4a 签名的请求在多个 Amazon Web Services 区域中都有效。SigV4a 通过跨区域请求签名来提高可用性，方便在发生区域中断时自动失效转移到备用区域。这对于像我们的 Amazon 这样的 Amazon Identity and Access Management 全球服务特别有利 CloudFront。

有关这两种身份验证方案的更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的Amazon 签名版本 4。

使用以下方法配置此功能：

auth_scheme_preference-共享 Amazon **config**文件设置, **AWS_AUTH_SCHEME_PREFERENCE** - 环境变量, **aws.authSchemePreference**-JVM 系统属性：仅限 Java/Kotlin

按优先顺序指定首选身份验证方案列表（以逗号分隔）。当一项服务支持多种身份验证方案时，SDK 会尝试按指定顺序使用该列表中的方案，如果所有首选方案都不可，用则会回退到默认行为。

默认值：无。

有效值：以下一项或多项的逗号分隔列表：

- **sigv4**：签名版本 4（性能最快，单区域）
- **sigv4a**：签名版本 4a（可用性更强，跨区域支持，签名性能比 SigV4 慢）
- **httpBearerAuth**：HTTP 不记名令牌身份验证

方案名称之间的空格和制表符将被忽略。

在 config 文件中将该值设置为首选 SigV4a 的示例：

```
[default]
auth_scheme_preference=sigv4a,sigv4
```

sigv4a_signing_region_set-共享 Amazon **config**文件设置, **AWS_SIGV4A_SIGNING_REGION_SET** - 环境变量

为 Sigv4a 多区域签名指定以逗号分隔 Amazon Web Services 区域 的列表。如果所选的身份验证方案为 SigV4a，则将此用作为该请求设置的默认区域。

默认值：因请求而定。

有效值：以逗号分隔的 Amazon Web Services 区域列表。区域之间的空格和制表符将被忽略。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	否	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	否	

Amazon Web Services 区域

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon Web Services 区域 是使用时需要理解的重要概念 Amazon Web Services 服务。

使用 Amazon Web Services 区域，您可以访问 Amazon Web Services 服务 实际居住在特定地理区域的内容。这可用于保证您的数据和应用程序接近您和用户访问它们的位置。区域提供容错能力、稳定性和弹性，还可以减少延迟。使用区域，您能够创建保持可用且不受区域中断影响的冗余资源。

大多数 Amazon Web Services 服务 请求都与特定的地理区域相关联。除非您明确使用 Amazon Web Services 服务提供的复制功能，否则在一个区域中创建的资源在任何其他区域中都不存在。例如，Amazon S3 和 Amazon EC2 支持跨区域复制。某些服务（例如 IAM）没有区域资源。

Amazon Web Services 一般参考 包含有关以下内容的信息：

- 要了解区域和端点之间的关系，并查看现有区域端点的列表，请参阅Amazon 服务端点。
- 要查看当前各 Amazon Web Services 服务所有支持的区域和端点列表，请参阅服务端点和限额。

创建服务客户端

要以编程方式访问 Amazon Web Services 服务， SDKs 请分别 Amazon Web Services 服务使用客户端 class/object 。例如，如果您的应用程序需要访问 Amazon EC2，则您的应用程序将创建一个 Amazon EC2 客户端对象来与该服务交互。

如果没有在代码本身中为客户端显式指定区域，则客户端将默认使用通过以下 `region` 设置设定的区域。但是，可以为任何单个客户端对象显式设置客户端的活动区域。以这种方式设置区域优先于该特定服务客户端的任何全局设置。备用区域是在该客户端的实例化过程中指定的，该区域特定于您的 SDK（请查看您的特定 SDK 指南或 SDK 的代码库）。

使用以下方法配置此功能：

`region`-共享 Amazon **`config`**文件设置, **`AWS_REGION`** - 环境变量, **`aws.region`**-JVM 系统属性：仅限 Java/Kotlin

指定 Amazon 请求 Amazon Web Services 区域 使用的默认值。此区域用于未提供特定区域的 SDK 服务请求。

默认值：无。必须明确指定此值。

有效值：

- 可用于所选服务的任何区域代码，有关列表，请参阅 Amazon 一般参考中的 Amazon 服务端点。例如，值 `us-east-1` 将端点设置为 Amazon Web Services 区域 美国东部（弗吉尼亚州北部）。
- `aws-global`为除区域终端节点之外还支持单独的全局终端节点的服务指定全局终端节点，例如 Amazon Security Token Service (Amazon STS) 和亚马逊简单存储服务 (Amazon S3) Service。

在 `config` 文件中设置此值的示例：

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

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_REGION=us-west-2
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_REGION us-west-2
```

大多数 SDKs 都有一个 “配置” 对象，可用于在应用程序代码中设置默认区域。有关详细信息，请参阅您的特定 Amazon SDK 开发者指南。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	Amazon CLI v2 在中的任何值`AWS_REGION` 之前使用中的任何值`AWS_DEFAULT_REGION` （两个变量都被选中）。
Amazon CLI v1	是	Amazon CLI v1 使用`AWS_DEFAULT_REGION` 为此目的命名的环境变量。
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x（V1）的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	是	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	此 SDK 使用名为 `AWS_DEFAULT_REGION` 的环境变量来实现此目的。

SDK	支持	备注或更多信息
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

Amazon STS 区域终端节点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon Security Token Service (Amazon STS) 既提供全球服务，也提供区域服务。其中一些 Amazon SDKs和默认 CLIs 使用全球服务终端节点 (`https://sts.amazonaws.com`)，而有些则使用区域服务终端节点 (`https://sts.{region_identifier}.{partition_domain}`)。在默认启用的区域中，向 Amazon STS 全球终端节点发出的请求会自动在请求发起的同一区域中提供服务。在可选区域中，向 Amazon STS 全球终端节点发出的请求由一个 Amazon Web Services 区域美国东部（弗吉尼亚北部）提供服务。有关 Amazon STS 终端节点的更多信息，请参阅 Amazon Security Token Service API 参考 Amazon STS 中的终端节点或Amazon Identity and Access Management 用户指南 Amazon Web Services 区域中的管理。

Amazon 最佳做法是尽可能使用区域终端节点并配置您的终端节点Amazon Web Services 区域。非商业分区中的客户必须使用区域性端点。并非所有 SDKs 工具都支持此设置，但所有工具都围绕全球和区域端点定义了行为。有关更多信息，请参阅下文的 部分。

> **Note**
>
> Amazon 已对默认启用的区域中的 Amazon Security Token Service (Amazon STS) 全球终端节点 (`https://sts.amazonaws.com`) 进行了更改，以增强其弹性和性能。 Amazon STS

对全局终端节点的请求将自动以与您的工作负载 Amazon Web Services 区域 相同的方式处理。这些更改不会部署到选择加入的区域。我们建议您使用适当的 Amazon STS 区域终端节点。有关更多信息，请参阅《Amazon Identity and Access Management 用户指南》中的 Amazon STS 全局端点更改。

对于支持此设置的 SDKs 和工具，客户可以使用以下方式配置功能：

sts_regional_endpoints-共享 Amazon **config**文件设置, **AWS_STS_REGIONAL_ENDPOINTS** - 环境变量

此设置指定 SDK 或工具如何确定用于与 Amazon Security Token Service (Amazon STS) 通信的 Amazon Web Services 服务 端点。

默认值：`regional`，相关例外详见下表。

> **Note**
>
> 2022 年 7 月之后发布的所有新 SDK 主要版本都将默认为`regional`。新的 SDK 主要版本可能会删除此设置并使用`regional`行为。为了减少此变更对未来的影响，我们建议您尽可能在应用程序中开始使用`regional`。

有效值：（建议的值：*`regional`*）

- **`legacy`**— 使用全局 Amazon STS 终端节点`sts.amazonaws.com`。
- **`regional`**— SDK 或工具始终使用当前配置区域的 Amazon STS 终端节点。例如，如果将客户端配置为使用`us-west-2`，则对的所有调用都将 Amazon STS 发送到区域终端节点`sts.us-west-2.amazonaws.com`，而不是全球`sts.amazonaws.com`终端节点。要在启用此设置时向全局端点发送请求，您可以将区域设置为 `aws-global`。

在 `config` 文件中设置这些值的示例：

```
[default]
sts_regional_endpoints = regional
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_STS_REGIONAL_ENDPOINTS=regional
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_STS_REGIONAL_ENDPOINTS regional
```

Support Amazon SDKs by 和工具

> **Note**
>
> Amazon 最佳做法是尽可能使用区域终端节点并配置您的终端节点Amazon Web Services 区域。

下表汇总了 SDK 或工具的下列配置：

- 支持设置：是否支持 STS 区域性端点的共享 `config` 文件变量和环境变量。
- 默认设置值：该设置（如果支持）的默认值。
- 默认服务客户端目标 STS 端点：即使更改默认端点的设置不可用，客户端也会使用的默认端点。
- 服务客户端回退行为：当 SDK 本应使用区域性端点但尚未配置区域时的行为。无论使用区域性端点是属于默认行为，还是因为该设置选择了 `regional`，都会出现这种行为。

该表还使用了下列值：

- 全局端点：`https://sts.amazonaws.com`
- 区域性端点：基于应用程序使用的 Amazon Web Services 区域配置。
- **`us-east-1`（区域性）**：使用 `us-east-1` 区域性端点，但会话令牌要比典型的全局请求长。

SDK	默认设置值	默认服务客户端目标 STS 端点	服务客户端回退行为	备注或更多信息
Amazon CLI v2	不适用	区域端点	全局端点	

SDK		默认设置值	默认服务客户端目标 STS 端点	服务客户端回退行为	备注或更多信息
Amazon CLI v1	是	legacy	全局端点	全局端点	
适用于 C++ 的 SDK	否	不适用	区域端点	us-east-1（区域性）	
适用于 Go V2 (1.x) 的 SDK	否	不适用	区域端点	请求失败	
适用于 Go 1.x（V1）的 SDK	是	legacy	全局端点	全局端点	要使用共享 config 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	否	不适用	区域端点	请求失败	如果未配置区域，则 AssumeRole 和 AssumeRoleWithWebIdentity 将使用全局 STS 端点。
适用于 Java 1.x 的 SDK	是	legacy	全局端点	全局端点	
适用于 JavaScript 3.x 的软件开发工具包	否	不适用	区域端点	us-east-1（区域性）	
适用于 JavaScript 2.x 的 SDK	是	legacy	全局端点	全局端点	

SDK		默认设置值	默认服务客户端目标 STS 端点	服务客户端回退行为	备注或更多信息
适用于 Kotlin 的 SDK	否	不适用	区域端点	全局端点	
适用于.NET 4.x 的 SDK	否	不适用	区域端点	us-east-1（区域性）	
适用于 .NET 3.x 的 SDK	是	regional	全局端点	全局端点	
适用于 PHP 3.x 的 SDK	是	regional	全局端点	请求失败	
适用于 Python (Boto3) 的 SDK	是	regional	全局端点	全局端点	
适用于 Ruby 3.x 的 SDK	是	regional	区域端点	请求失败	
适用于 Rust 的 SDK	否	不适用	区域端点	请求失败	
适用于 Swift 的 SDK	否	不适用	区域端点	请求失败	
适用于 PowerShell V5 的工具	是	regional	全局端点	全局端点	
适用于 PowerShell V4 的工具	是	regional	全局端点	全局端点	

Amazon S3 数据完整性保护

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

一段时间 Amazon SDKs 以来，一直支持在向亚马逊简单存储服务上传数据或从中下载数据时进行数据完整性检查。以前，这些支持采用选择加入方式。现在，我们使用基于 CRC 的算法（例如 CRC32 或 CRC64 NVME）默认启用了这些检查。尽管各个 SDK 或工具都有其默认的算法，不过您也可以选择其他的算法。您还可以选择继续手动为上传提供预先计算的校验和。上传、分段上传、下载和加密模式均使用一致的行为，简化了客户端完整性检查的过程。

我们的 Amazon SDKs 最新版本 Amazon CLI 会自动计算每次上传的基于循环冗余校验 (CRC) 的校验和，并将其发送到 Amazon S3。Amazon S3 会在服务器端独立计算校验和，并使用提供的值对其进行验证，然后才会将对象及其校验和持久地存储在对象的元数据中。通过将校验和与对象一起存储在元数据中，下载对象时将可以自动返回相同的校验和并用于验证下载。您也可以随时验证存储在对象元数据中的校验和。

要详细了解校验和操作、分段上传或支持的校验和算法列表，请参阅《Amazon Simple Storage Service 用户指南》中的在 Amazon S3 中检查对象完整性。

分段上传：

Amazon S3 还为开发人员提供了涵盖单段上传和分段上传的一致完整对象校验和，。

分成多个部分上传文件时，会 SDKs 计算每个部分的校验和。Amazon S3 使用这些校验和来通过 `UploadPart` API 验证每个分段的完整性。此外，Amazon S3 会在您调用 `CompleteMultipartUpload` API 时验证整个文件的大小以及校验和。

如果您的 SDK 使用 Amazon S3 Transfer Manager 来协助分段上传，则将使用 Support Amazon SDKs by 和工具 表中特定于 SDK 的默认算法来验证分段的校验和。您可以通过将设置设置设置为`FULL_OBJECT`或选择使用 CRC64 NVME 算`checksum_type`法来选择启用完整的对象校验和。

如果您使用的是较早版本的 SDK 或 Amazon CLI：

如果您的应用程序使用 2024 年 12 月之前的软件开发工具包或工具，Amazon S3 仍会计算新对象的 CRC64 NVME 校验和，并将其存储在对象元数据中以备将来参考。您可以稍后将存储的 CRC 与您计

算的 CRC 进行比较，验证网络传输是否正确。此外，您仍然可以通过 PutObject 或 UploadPart 请求提供自己预先计算的校验和来手动扩展完整性保护，这是早期版本中解决此问题的标准方法。

使用以下方法配置此功能：

request_checksum_calculation-共享 Amazon **config**文件设置，**AWS_REQUEST_CHECKSUM_CALCULATION** - 环境变量，**aws.requestChecksumCalculation**-JVM 系统属性：仅限 Java/Kotlin

默认情况下，用户会在发送请求时选择启用请求校验和计算。用户可以在构建请求过程中选择任何一种可用的校验和算法。如果用户未进行选择，则将使用 SDK 特定的默认算法。有关各个 SDK 或工具的默认算法，请参阅 Support Amazon SDKs by 和工具表。

默认值：WHEN_SUPPORTED

有效值：

- **WHEN_SUPPORTED**：在 API 操作支持时（例如向 Amazon S3 传输数据时）对所有请求有效载荷执行校验和验证。
- **WHEN_REQUIRED**：仅在 API 操作要求时才执行校验和验证。

response_checksum_validation-共享 Amazon **config**文件设置，**AWS_RESPONSE_CHECKSUM_VALIDATION** - 环境变量，**aws.responseChecksumValidation**-JVM 系统属性：仅限 Java/Kotlin

默认情况下，用户在发送请求时会选择启用执行响应校验和验证。计算响应有效载荷的校验和，并与校验和响应标头进行比较。如果校验和验证失败，则在读取有效载荷时会向用户发出错误。

校验和响应标头还会指示校验和的算法。对于所有支持校验和的 Amazon S3 API 操作，Amazon S3 客户端都会尝试验证响应校验和。但如果 SDK 尚未实现指定的校验和算法，则会跳过此验证。

默认值：WHEN_SUPPORTED

有效值：

- **WHEN_SUPPORTED**：在 API 操作支持时（例如向 Amazon S3 传输数据时）对所有响应有效载荷执行校验和验证。
- **WHEN_REQUIRED**：仅在 API 操作支持且调用方已为该操作显式启用校验和时，才会执行校验和验证。例如，调用 Amazon S3 GetObject API 并且 ChecksumMode 参数设置为“启用”时。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

> **Note**
>
> 在下表中，“CRT”是指 Amazon 通用运行时 (CRT) 库，并且可能需要向您的项目添加其他依赖项。

SDK	支持	默认校验和算法	支持的校验和算法	备注或更多信息
Amazon CLI v2	是	CRC64NVME	CRC64NVME、、 CRC32 C、CRC32、、xxHash3 SHA1 SHA256、xxHash64、xxHash128、 SHA512	对于 Amazon CLI v1，默认算法和支持的算法将与 Python (Boto3) 相同。
适用于 C++ 的 SDK	是	CRC64NVME	CRC64NVME、、 CRC32 C、CRC32、、xxHash3 SHA1 SHA256、xxHash64、xxHash128、 SHA512	
适用于 Go V2 (1.x) 的 SDK	是	CRC32	CRC64NVME、CRC32、 CRC32 C、、 SHA1 SHA256	
适用于 Go 1.x (V1) 的 SDK	否			
适用于 Java 2.x 的 SDK	是	CRC32	CRC32 ， CRC32C ， SHA1 ， SHA256	

SDK	支持	默认校验和算法	支持的校验和算法	备注或更多信息
			仅通过 CRT：CRC64NVME、xxHash3、xxHash64、xxHash128、 SHA512	
适用于 Java 1.x 的 SDK	否			
适用于 JavaScript 3.x 的软件开发工具包	是	CRC32	CRC32 ， CRC32C ， SHA1 ， SHA256	
适用于 JavaScript 2.x 的 SDK	否			
适用于 Kotlin 的 SDK	是	CRC32	CRC32 ， CRC32C ， SHA1 ， SHA256	
适用于.NET 4.x 的 SDK	是	CRC32	CRC32, CRC32 C, SHA1, SHA256, SHA512	
适用于 .NET 3.x 的 SDK	是	CRC32	CRC32, CRC32 C, SHA1, SHA256, SHA512	
适用于 PHP 3.x 的 SDK	是	CRC32	CRC32, SHA1, SHA256 仅通过 CRT：C CRC32	`awscrt`需要扩展名才能使用 CRC32 C。

SDK	支持	默认校验和算法	支持的校验和算法	备注或更多信息
适用于 Python (Boto3) 的 SDK	是	CRC32	CRC32, SHA1, SHA256 仅通过 CRT：CRC32C、 CRC64NVME、xxHash3、xxHash64、xxHash128、SHA512	
适用于 Ruby 3.x 的 SDK	是	CRC32	CRC32, SHA1, SHA256 仅通过 CRT：CRC64NVME、CRC32	
适用于 Rust 的 SDK	是	CRC32	CRC64NVME、CRC32、 CRC32C、、 SHA1 SHA256	
适用于 Swift 的 SDK	是	CRC32	CRC64NVME、CRC32、 CRC32C、、 SHA1 SHA256	所有算法都需要 CRT 依赖项。
适用于 PowerShell V5 的工具	是	CRC32	CRC32, CRC32C,, SHA1, xxHash3 SHA256, SHA512	
适用于 PowerShell V4 的工具	是	CRC32	CRC32, CRC32 C, SHA1, SHA256, SHA512	

双堆栈和 FIPS 端点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

使用以下方法配置此功能：

use_dualstack_endpoint-共享 Amazon **config**文件设置, **AWS_USE_DUALSTACK_ENDPOINT** - 环境变量, **aws.useDualstackEndpoint**-JVM 系统属性：仅限 Java/Kotlin

开启或关闭 SDK 是否向双堆栈端点发送请求。要了解有关双堆栈终端节点的更多信息，请参阅《亚马逊简单存储服务用户指南》中的 “使用 Amazon S3 双堆栈终端节点”。 IPv4 IPv6 双堆栈端点适用于某些区域。

默认值：`false`

有效值：

- **true** – SDK 或工具将尝试使用双堆栈端点发出网络请求。如果服务和/或 Amazon Web Services 区域不存在双堆栈端点，则请求将失败。
- **false** – SDK 或工具将不会使用双堆栈端点发出网络请求。

use_fips_endpoint-共享 Amazon **config**文件设置, **AWS_USE_FIPS_ENDPOINT** - 环境变量, **aws.useFipsEndpoint**-JVM 系统属性：仅限 Java/Kotlin

开启或关闭 SDK 或工具是否向符合 FIPS 的端点发送请求。联邦信息处理标准 (FIPS) 是美国政府对数据及其加密的一系列安全要求。政府机构、合作伙伴以及希望与联邦政府开展业务的机构必须遵守 FIPS 指导方针。与标准 Amazon 端点不同，FIPS 端点使用经过 FIPS 140 验证的 TLS 软件库。如果启用此设置，并且您的服务中不存在 FIPS 终端节点 Amazon Web Services 区域，则 Amazon 呼叫可能会失败。 特定于服务的端点以及 Amazon Command Line Interface 覆盖此设置的`--endpoint-url`选项。

要详细了解通过其他方式指定 FIPS 终端节点 Amazon Web Services 区域，请参阅按服务划分的 FIPS 终端节点。有关亚马逊弹性计算云服务终端节点的更多信息，请参阅《亚马逊 EC2 API 参考》中的双栈（IPv4 和 IPv6）终端节点。

默认值：`false`

有效值：

- **`true`** – SDK 或工具将向符合 FIPS 的端点发送请求。
- **`false`** – SDK 或工具将不会向符合 FIPS 的端点发送请求。

Support by Amazon SDKs and 工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

端点发现

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

SDKs 使用端点发现来访问服务端点（URLs 访问各种资源），同时仍然可以灵活地根据 Amazon 需要 URLs 进行更改。这样，您的代码就可以自动检测新的端点。某些服务没有固定的端点。相反，您可以在运行时通过请求先获取端点来获得可用的端点。检索到可用端点后，代码会使用该端点访问其他操作。例如，对于 Amazon Timestream，SDK 会发出`DescribeEndpoints`请求以检索可用的端点，然后使用这些端点完成特定操作，例如`CreateDatabase`或`CreateTable`。

使用以下方法配置此功能：

`endpoint_discovery_enabled`-共享 Amazon **`config`**文件设置，**`AWS_ENABLE_ENDPOINT_DISCOVERY`** - 环境变量，**`aws.endpointDiscoveryEnabled`**-JVM 系统属性：仅限 Java/Kotlin，要直接在代码中配置值，请直接查阅您的特定 SDK。

开启或关闭 DynamoDB 的端点发现功能。

端点发现在 Timestream 中为必需，而在 Amazon DynamoDB 中为可选。此设置的默认值为 `true` 或 `false`，具体取决于端点发现功能对于该服务是否为必需。对于 Timestream 请求的默认值为 `true`，而对于 Amazon DynamoDB 请求的默认值为 `false`。

有效值：

- **`true`** – 对于端点发现是可选的服务，SDK 应自动尝试发现端点。
- **`false`** – 对于端点发现是可选的服务，SDK 不应自动尝试发现端点。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	是	要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	是	适用于 Java 的 SDK 2.x 使用 `AWS_ENDPOINT_DISCOVERY_ENABLED` 作为环境变量名称。
适用于 Java 1.x 的 SDK	部分	不支持 JVM 系统属性。
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	部分	仅支持 Timestream。
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

常规配置设置

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

SDKs 支持一些用于配置 SDK 整体 SDK 行为的常规设置。

使用以下方法配置此功能：

`api_versions`-共享 Amazon **`config`**文件设置

有些 Amazon 服务维护多个 API 版本以支持向后兼容。默认情况下，SDK 和 Amazon CLI 操作使用最新的可用 API 版本。如要求使用特定的 API 版本来处理您的请求，请在您的个人资料中添加该 `api_versions` 设置。

默认值：无。（SDK 使用的最新 API 版本。）

有效值：这是一个嵌套设置，后面有一行或多行缩进，每行标识一项 Amazon 服务和要使用的 API 版本。要了解有哪些 API 版本可用，请参阅该 Amazon 服务的文档。

该示例为config文件中的两个 Amazon 服务设置了特定的 API 版本。这些 API 版本仅用于在包含这些设置的配置文件下运行的命令。任何其他服务的命令都使用该服务的 API 的最新版本。

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

ca_bundle-共享 Amazon **config**文件设置, **AWS_CA_BUNDLE** - 环境变量

指定在建立 SSL/TLS 连接时使用的自定义证书包（带有.pem扩展名的文件）的路径。

默认值： 无

有效值：指定完整路径或基本文件名。如果存在基本文件名，则系统会尝试在 PATH 环境变量指定的文件夹中查找该程序。

在 config 文件中设置此值的示例：

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

由于操作系统的路径处理方式和路径字符转义方式方面的差异，以下是在 Windows 上的 config 文件中设置此值的示例：

```
[default]
ca_bundle = C:\\Users\\username\\.aws\\aws-custom-bundle.pem
```

Linux/macOS 通过命令行设置环境变量的示例：

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

Windows 通过命令行设置环境变量的示例：

```
setx AWS_CA_BUNDLE C:\dev\apps\ca-certs\cabundle-2019mar05.pem
```

output-共享 Amazon **config**文件设置

指定如何在 Amazon CLI 和其他工具中设置结果 Amazon SDKs 的格式。

默认值：json

有效值：

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

`parameter_validation`-共享 Amazon **`config`**文件设置

指定 SDK 或工具在将命令行参数发送到 Amazon 服务端点之前是否尝试验证这些参数。

默认值：`true`

有效值：

- **`true`** – 默认值。SDK 或工具执行命令行参数的客户端验证。这有助于 SDK 或工具确认参数是否有效，并捕获一些错误。在向 Amazon 服务端点发送请求之前，SDK 或工具可以拒绝无效的请求。
- **`false`**— SDK 或工具在将命令行参数发送到 Amazon 服务端点之前不会对其进行验证。Amazon 服务端点负责验证所有请求并拒绝无效的请求。

Support by Amazon SDKs and too

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	部分	`api_versions` 不支持。
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	部分	不支持 `api_versions` 和 `parameter_validation` 。

SDK	支持	备注或更多信息
适用于 Go 1.x (V1) 的 SDK	部分	不支持 `api_versions` 和 `parameter_validation` 。要使用共享 `config` 文件设置，必须开启从配置文件加载的功能；请参阅会话。
适用于 Java 2.x 的 SDK	否	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	否	
适用于.NET 4.x 的 SDK	否	
适用于 .NET 3.x 的 SDK	否	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	否	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	否	
适用于 PowerShell V4 的工具	否	

主机前缀注入

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

主机前缀注入是一项功能，它 Amazon SDKs 会自动为某些 API 操作在服务端点的主机名前添加前缀。此前缀可以是一个静态字符串，也可以是包含请求参数中数据的动态值。

例如，在使用 Amazon 简单存储服务对 Amazon S3 对象或存储桶执行操作时，软件开发工具包会在最终的 API 终端节点中替换您的存储桶名称和 Amazon Web Services 账户 ID。

虽然普通 Amazon 服务终端节点需要这种行为，但在使用自定义终端节点（例如 VPC 终端节点或本地测试工具）时，它可能会导致问题。对于这些情况，您可能需要禁用主机前缀注入。

使用以下方法配置此功能：

`disable_host_prefix_injection`-共享 Amazon **`config`**文件设置，**`AWS_DISABLE_HOST_PREFIX_INJECTION`** - 环境变量，**`aws.disableHostPrefixInjection`**-JVM 系统属性：仅限 Java/Kotlin

此设置用于控制 SDK 或工具是否将通过附加在 SDK 的客户端对象或变量中定义的主机前缀来修改端点主机名。

默认值：`false`

有效值：

- **`true`**：禁用主机前缀注入。SDK 不会修改端点主机名。
- **`false`**：启用主机前缀注入。SDK 将在端点主机名前附加主机前缀。

在 `config` 文件中设置此值的示例：

```
[default]
disable_host_prefix_injection = true
```

Linux/macOS 通过命令行设置环境变量的示例：

```
export AWS_DISABLE_HOST_PREFIX_INJECTION=true
```

Windows 通过命令行设置环境变量的示例：

```
setx AWS_DISABLE_HOST_PREFIX_INJECTION true
```

主机前缀注入示例

下表的示例显示了在启用和禁用主机前缀注入时如何 SDKs 修改最终端点。

- 主机前缀：在 SDK 客户端对象或代码变量上设置的主机前缀属性字符串模板。
- 输入：在 SDK 客户端对象或代码变量上设置的其他输入。
- 客户端端点：客户端的派生端点。
- 设置值：先前设置的解析值。
- 结果端点：SDK 客户端用于执行 API 调用的最终端点。

主机前缀	输入	客户端端点	设置值	结果端点
“数据。”	{}	"https://service.us-west-2.amazonaws.com"	false	"https://data.service.us-west-2.amazonaws.com"
“{Bucket}-{AccountId}。”	存储桶：“amzn-s3-demo-bucket1”，：“123456789012" AccountId	"https://service.us-west-2.amazonaws.com"	false	"https://amzn-s3-demo-bucket1-123456789012.service.us-west-2.amazonaws.com"
“数据。”	{}	"https://override.us-west-2.amazonaws.com"（作为替代端点）	true	"https://override.us-west-2.amazonaws.com"

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	否	不支持设置，但可以通过客户端使用 enableHostPrefixInjection 在代码中配置。
适用于 Go V2 (1.x) 的 SDK	否	可以使用中间件禁用。
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	否	不支持设置，但可以通过客户端使用 SdkAdvancedClientOption.DISABLE_HOST_PREFIX_INJECTION 在代码中配置。
适用于 Java 1.x 的 SDK	否	不支持设置，但可以通过客户端使用 withDisableHostPrefixInjection 在代码中配置。
适用于 JavaScript 3.x 的软件开发工具包	否	不支持设置，但可以通过客户端使用 disableHostPrefix 在代码中配置。
适用于 JavaScript 2.x 的 SDK	否	不支持设置，但可以通过客户端使用 hostPrefixEnabled 在代码中配置。
适用于 Kotlin 的 SDK	否	
适用于.NET 4.x 的 SDK	否	不支持设置，但可以通过客户端使用 DisableHostPrefixInjection 在代码中配置。
适用于 .NET 3.x 的 SDK	否	不支持设置，但可以通过客户端使用 DisableHostPrefixInjection 在代码中配置。
适用于 PHP 3.x 的 SDK	否	不支持设置，但可以通过客户端使用 disable_host_prefix_injection 在代码中配置。

SDK	支持	备注或更多信息
适用于 Python (Boto3) 的 SDK	是	可以通过客户端使用 `inject_host_prefix` 在代码中配置。
适用于 Ruby 3.x 的 SDK	否	不支持设置，但可以通过客户端使用 `disable_host_prefix_injection` 在代码中配置。
适用于 Rust 的 SDK	否	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	否	不支持设置，但可以使用参数 `-ClientConfig @{DisableHostPrefixInjection = $true}` 将其包含在特定的 cmdlet 中。
适用于 PowerShell V4 的工具	否	不支持设置，但可以使用参数 `-ClientConfig @{DisableHostPrefixInjection = $true}` 将其包含在特定的 cmdlet 中。

IMDS 客户端

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

SDKs 使用面向会话的请求实现实例元数据服务版本 2 (IMDSv2) 客户端。有关更多信息 IMDSv2，请参阅 Amazon EC2 用户指南 IMDSv2中的使用。IMDS 客户端可通过 SDK 代码库中提供的客户端配置对象进行配置。

使用以下方法配置此功能：

`retries` - 客户端配置对象成员

任何失败的请求的额外重试次数。

默认值：3

有效值：大于 0 的数字。

port - 客户端配置对象成员

端点的端口。

默认值：80

有效值：数字。

token_ttl - 客户端配置对象成员

令牌的 TTL。

默认值：21,600 秒（6 小时，分配的最长时间)。

有效值：数字。

endpoint - 客户端配置对象成员

IMDS 的端点。

默认值：如果 `endpoint_mode` 等于 `IPv4`，则默认端点为 `http://169.254.169.254`。如果 `endpoint_mode` 等于 `IPv6`，则默认端点为 `http://[fd00:ec2::254]`。

有效值：有效的 URI。

大多数人支持以下选项 SDKs。有关详细信息，请参阅您的特定 SDK 代码库。

endpoint_mode - 客户端配置对象成员

IMDS 的端点模式。

默认值：`IPv4`

有效值：`IPv4`、`IPv6`

http_open_timeout - 客户端配置对象成员（名称可能有所不同）

等待连接打开的秒数。

默认值：1 秒。

有效值：大于 0 的数字。

http_read_timeout - 客户端配置对象成员（名称可能有所不同）

读取一个数据块的秒数。

默认值：1 秒。

有效值：大于 0 的数字。

http_debug_output - 客户端配置对象成员（名称可能有所不同）

设置用于调试的输出流。

默认值：无。

有效值：有效的 I/O 直播，例如 STDOUT。

backoff - 客户端配置对象成员（名称可能有所不同）

在两次重试之间休眠的秒数，或者客户提供的回退功能可供调用。这会覆盖默认的指数回退策略。

默认值：因 SDK 而异。

有效值：因 SDK 而异。可以是数值，也可以是对自定义函数的调用。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	否	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x（V1）的 SDK	是	
适用于 Java 2.x 的 SDK	是	

SDK	支持	备注或更多信息
适用于 Java 1.x 的 SDK	是	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	是	
适用于 Kotlin 的 SDK	否	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

重试行为

> **Important**
>
> 此页面上描述的行为需要选择加入，直到它成为默认行为。`AWS_NEW_RETRIES_2026=true`在您的环境中设置。如果没有此设置，您的 SDK 将使用 2026 年之前的重试行为，这在退避时间、重试配额成本和特定于服务的默认值方面有所不同。有关详细信息，请参阅公告博客文章。

当对的请求因暂时性错误或限制而 Amazon Web Services 服务 失败时，SDK 可以自动重试该请求。本页介绍如何配置重试及其内部工作方式。

- 配置重试次数：选择重试模式，设置最大尝试次数，并了解配置优先级。
- 重试的工作原理: 重试流程、错误分类、退避公式、重试配额机制和特定于服务的行为。

配置重试次数

您可以控制 SDK 使用哪种重试策略以及重试次数。

选择重试模式

重试模式决定了请求失败时 SDK 的行为。有三种模式可供选择：标准模式、自适应模式和传统模式。

	标准	自适应	Legacy
重试配额	支持	是	因 SDK 而异
可以延迟初始请求	否	是	否
Error-type-specific 退缩	支持	是	因 SDK 而异
跨软件开发工具包实现标准化	支持	是	否
建议	所有工作负载的默认值	Single-resource，节流密集，耐延迟	仅向后兼容

标准模式（默认）

标准模式使用带抖动的指数退避重试失败的请求。它对瞬态错误（例如网络超时）使用较短的延迟，对限制错误（例如）使用更长的延迟。`ThrottlingException`

标准模式包括重试配额，一个令牌桶，用于每次重试时扣除令牌，并在请求成功时补充令牌。当可用令牌用完时，SDK 会在不重试的情况下返回错误，因此您的应用程序会快速失败，而不是等待不太可能成功的重试。这还可以减少重试流量，从而帮助更快地解决服务中断问题。在正常操作期间，配额保持已满且无效。重试配额永远不会延迟或阻止初始请求。只有重试次数会受到影响。有关更多信息，请参阅 重试配额（令牌桶）。

除非您有特殊理由选择其他模式，否则请使用标准模式。

自适应模式

自适应模式包括标准模式下的所有内容，以及客户端速率限制器。速率限制器跟踪限制响应并调整 SDK 发送请求的速率。与标准模式不同，当检测到限制时，自适应模式可以延迟或阻止初始请求，而不仅仅是重试。

速率限制器按照 SDK 客户端实例运行。来自客户端的所有请求都具有相同的速率限制，无论它们针对哪个 API 操作或资源。

何时使用自适应模式：

- 您的客户端以单个资源（例如，一个 DynamoDB 表）为目标，并且您希望频繁出现限制响应。这在自动工作流程、批处理器或大批量调用单个 API 操作的 AI 工作负载中很常见。
- 您希望 SDK 在服务发出限制信号时自动减速。

何时不使用自适应模式：

- 您的客户端向多个资源发送请求或为多个租户提供服务。限制一个资源会导致速率限制器减慢来自该客户端的所有请求，包括对未受影响的资源的请求。
- 初始请求需要可预测的延迟。

不建议将自适应模式作为常规默认模式。

传统模式

传统模式是每个 SDK 在引入标准模式之前使用的重试行为。它不包括标准化的重试配额。某些 SDK（例如 Java）在传统模式下有自己的重试配额实现，但各个 SDK 的行为并不一致。如果没有标准化配额，则在服务中断期间，客户端会继续全速重试。这会在不太可能成功的请求上占用线程和连接，同时增加可能延迟服务恢复的负载。

传统模式因软件开发工具包而异。重试次数、退避时间、可重试错误集和限制行为因语言而异。在软件开发工具包之间移动时，依赖于传统重试行为的代码的行为可能会有所不同。

可用版本：Java、Python、Ruby、PHP、C++、CLI

不适用于：.NET、Go、Kotlin、Rust、Swift、 JavaScript

存在传统模式是为了向后兼容。如果您当前使用传统模式，请切换到标准模式。

重试设置

以下设置控制重试行为。你可以通过环境变量、共享配置文件 (~/.aws/config) 或代码中的客户端配置来设置它们。

设置	它控制什么	环境变量	Config 文件密钥	默认
重试模式	使用哪种重试策略	AWS_RETRY_MODE	retry_mode	standard
最大尝试次数	包括初始请求在内的总尝试次数	AWS_MAX_ATTEMPTS	max_attempts	3（参见备注）

最大尝试值为，3表示 SDK 会发出一个初始请求和最多两次重试。将最大尝试次数设置为，1以完全禁用重试次数。

> **Note**
>
> DynamoDB 和 DynamoDB Streams 客户端默认为最大尝试次数。4这些服务使用较短的基本退避延迟（25 毫秒而不是 50 毫秒）来匹配其低延迟特征。额外的尝试使最后一次重试的最大退避率与其他服务相当。您可以使用上表中显示的相同设置来覆盖此设置。

配置优先级

当您在多个位置指定相同的设置时，SDK 会使用以下优先顺序（从高到低）来解析该值：

1. 代码中的显式客户端配置。直接在 SDK 客户端或其配置对象上设置的值。
2. 环境变量。例如，AWS_RETRY_MODE 或 AWS_MAX_ATTEMPTS。
3. 共享配置文件。retry_mode或max_attempts键入~/.aws/config。
4. SDK 默认。该设置的内置默认值。

这遵循标准的 Amazon SDK 配置优先级。在较高级别设置的值总是会覆盖在较低级别上设置的值。例如，如果您retry_mode=standard在中设置AWS_RETRY_MODE=adaptive为环境变量~/.aws/config，则 SDK 将使用自适应模式。

Language-specific 配置

本页面（`retry_mode`和`max_attempts`）中描述的跨SDK设置适用于所有软件开发工具包。但是，在代码中配置重试的 API 因语言而异。有关特定语言的配置选项，例如自定义退避策略、其他可重试错误和重试配额调整，请参阅您的 SDK 开发者指南。

重试的工作原理

本节介绍 Amazon SDK 如何处理失败的请求：哪些错误会触发重试、SDK 在两次尝试之间等待多长时间以及何时停止重试。

请求失败时会发生什么

当您通过 SDK 进行 API 调用时， Amazon SDK 将遵循以下顺序：

1. 仅限自适应模式：SDK 会检查客户端速率限制器。如果检测到限制，SDK 可能会在发送请求之前延迟或阻止请求。
2. SDK 将请求发送到 Amazon Web Services 服务 终端节点。
3. 如果服务返回成功响应，SDK 会将结果返回到您的代码中。
4. 如果请求失败，SDK 会将错误分类为暂时错误、限制错误或不可重试错误。请参阅哪些错误会被重试。
5. 如果错误不可重试，SDK 会立即将错误返回到您的代码中。未尝试重试。
6. 如果错误是可重试的，SDK 会检查它是否已达到最大尝试次数。如果是，它会将错误返回到您的代码中。
7. 软件开发工具包会检查重试配额（令牌桶）. 如果代币预算耗尽，SDK 不会重试，而是将错误返回到您的代码中。例外：对于Long-polling 操作，SDK 在返回错误之前仍会应用退避延迟。
8. SDK 根据错误类型和重试尝试次数计算退避延迟。请参阅SDK 要等多久。
9. SDK 等待计算出的延迟，然后从步骤 2 开始再次发送请求。

SDK 会重复此循环，直到请求成功、达到最大尝试次数、用完重试配额或出现不可重试的错误。整个过程是自动的。您的应用程序要么看到成功响应，要么看到最终错误。

哪些错误会被重试

SDK 将每个失败的请求分为三类之一：暂时请求、限制请求或不可重试。此分类决定了 SDK 是否重试请求以及在重试之前等待多长时间。

分类基于服务响应中的错误代码和 HTTP 状态码。例如，带有错误代码的 HTTP 400 `RequestTimeout` 被归类为临时并重试。带有`ValidationException`的 HTTP 400 被归类为不可重试并立即返回。

错误分类

使用较短的基准延迟（50 ms）重试@@ 瞬态错误：

错误代码
`RequestTimeout`
`RequestTimeoutException`
`InternalError`
`IDPCommunicationError`
I/O 失败（连接重置、DNS 解析失败、套接字超时）
（任何没有识别错误代码的 HTTP 500、502、503 或 504）

使用更长的基本延迟（1,000 ms）重试@@ 限制错误：

错误代码
`Throttling`
`ThrottlingException`
`ThrottledException`
`RequestThrottledException`
`TooManyRequestsException`
`ProvisionedThroughputExceededException`
`TransactionInProgressException`

错误代码
LimitExceededException
PriorRequestNotComplete
RequestThrottled
EC2ThrottledException
RequestLimitExceeded
SlowDown
BandwidthLimitExceeded

Non-retryable 错误（例如AccessDeniedException、ValidationException、ResourceNotFoundException）会立即返回到您的代码中。

> **Note**
>
> 带有限制错误代码的 HTTP 5XX 被归类为限制错误，而不是暂时错误，即使 5XX 错误通常是暂时性的。SDK 首先匹配错误代码，然后回退到 HTTP 状态码。

限制错误意味着服务由于速率限制而主动拒绝了您的请求，因此 SDK 会等待更长的时间才能重试，让服务有时间恢复容量。SDK 要等多久有关具体延迟，请参阅。

SDK 要等多久

SDK 使用指数退避和完全抖动。平均而言，每次重试的等待时间都比上次更长，通过随机化来分散来自多个客户端的请求。

按错误类型划分的基本延迟

基本延迟取决于错误是瞬态的还是节流的：

错误类型	基础延迟	理由
瞬态（非节流）	50 毫秒	瞬态错误通常会在几毫秒内解决。短的基础延迟可实现快速恢复。
节流	1,000 ms	该服务对请求进行了速率限制。基础延迟越长，可以有时间恢复容量。

退避公式

SDK 使用以下公式计算每次重试延迟：

```
delay = random(0, 1) × min(20,000 ms, base_delay × 2^retry)
```

其中：

- `random(0, 1)`返回介于 0 和 1 之间的均匀分布值
- `base_delay`对于瞬态错误为 50 毫秒，对于节流错误为 1,000 毫秒
- `retry`第一次重试（第二次整体请求尝试）从 0 开始

最大退避上限为 20 秒。无论尝试了多少次，个别延迟都不会超过 20 秒。

行之有效的例子

示例 1：暂时错误，最大尝试次数 3 次

步骤	发生了什么	Delay
尝试 1	初始请求。服务返回 HTTP 503。	（无）
尝试 2	SDK 会随机等待（0, 50 毫秒）。使用 503 重试失败。	0—50 毫秒（平均约为 25 毫秒）
尝试 3	SDK 会随机等待（0, 100 毫秒）。重试成功。	0—100 毫秒（平均约为 50 毫秒）

两次重试的总增加延迟平均约为 75 毫秒。

示例 2：限制错误，最大尝试次数 3 次

步骤	发生了什么	Delay
尝试 1	初始请求。服务退货429. `Throttling`	（无）
尝试 2	SDK 会随机等待（0, 1,000 毫秒）。重试返回429。	0—1,000 毫秒（平均约为 500 毫秒）
尝试 3	SDK 会随机等待（0, 2,000 毫秒）。重试成功。	0—2,000 毫秒（平均约为 1,000 毫秒）

两次重试的总增加延迟平均约为 1,500 毫秒。

示例 3：瞬态错误，达到退避上限

基本延迟为 50 ms 时，上限之前的计算延迟为：

重试尝试	计算出的最大延迟	20 秒后上限
1	50 毫秒	50 毫秒
2	100 毫秒	100 毫秒
5	800 毫秒	800 毫秒
9	12,800 毫秒	12,800 毫秒
10	25,600 毫秒	200000 ms

对于暂时性错误，上限在第 10 次重试（第 11 次尝试）时生效。对于基准为 1,000 ms 的节流错误，上限将在第 6 次重试时生效。

> **Note**
>
> 默认情况下，最大尝试次数为 3 次（1 次初始请求 + 2 次重试），则永远不会达到退避上限。下表说明了如果增加幅度远`max_attempts`远超过默认值会发生什么。

为什么抖动很重要

随机乘数称为完全抖动。没有它，所有同时遇到错误的客户端都会同时重试，从而产生大量的重试流量（“雷鸣般的群体”问题）。完全抖动在整个退避窗口中均匀地分布重试次数，因此服务会收到稳定的请求流，而不是同步的峰值。

例如，假设 1,000 个客户同时收到 503。完全抖动会将他们的第一次重试次数均匀分布在 50 毫秒的时间段内，而不是将所有 1,000 次重试次数均匀分布在 50 毫秒。

Server-directed 重试时机

有些在错误响应中 Amazon Web Services 服务 包含`x-amz-retry-after`标题。标头值是以毫秒为单位的延迟。当存在此标头时，SDK 将使用服务器指定的延迟，限制为计算出的退避延迟的最小值和计算的退避延迟的最大值加上 5,000 ms。由于计算出的退避本身上限为 20 秒，因此服务器引导的最大有效延迟为 25 秒。SDK 不会对这个值应用抖动，因为服务应该会抖动它。这使服务能够在预期容量可用时准确地进行通信。

重试配额（令牌桶）

SDK 维护内部代币预算，用于跟踪成功请求与失败的比率。当故障普遍存在时，预算就会耗尽，SDK 会直接返回错误。您的应用程序会快速失败，而不是等待不太可能成功的重试。这还可以减少重试流量，帮助更快地解决服务中断问题。

重试配额的工作原理

代币预算一开始就满了。每次重试尝试都会扣除代币。重试成功后，SDK 会恢复该重试所消耗的令牌。当请求在第一次尝试时成功时（无需重试），SDK 会恢复 1 个令牌。当预算达到零时，SDK 会停止重试，并将错误直接返回到您的代码。

参数	值
预算容量	500 个代币
每次暂时（非限制）重试的成本	14 个代币
每次重试限制的费用	5 个代币
重试后成功恢复令牌	上次重试消耗量（14 或 5）
成功恢复令牌无需重试	1 个代币

瞬态重试的成本较高反映了它们的失败模式不同。诸如 500 和连接故障之类的暂时性错误通常表示存在服务范围的问题。在这种情况下，继续重试不太可能成功。它会增加您的通话延迟，占用客户资源，并可能延迟所有人的恢复。限制错误表明服务需要更多时间才能成功请求。为了提高成功的可能性，SDK 会在两次重试之间等待更长的时间。

配额何时会阻止重试

重试配额会随时跟踪代币，但只有在预算耗尽时才会阻止重试。在正常运行期间，几乎所有请求都会成功，并且预算仍然满额。配额对重试没有明显的影响。

成功的重试仅恢复其自身的代币成本（14 或 5 个令牌），而不会恢复同一个请求中较早失败的重试成本。例如，如果第一次重试失败，第二次重试成功，则预算净损失14个代币。当重试耗尽所有尝试但未成功时，预算消耗得最快，但是当请求需要多次重试才能成功时，预算也会逐渐耗尽。

默认情况下，最大尝试次数为 3 次，当超过大约 22% 的请求导致持续的暂时失败，或者超过大约 32% 的请求导致限制错误时，配额就会开始耗尽。低于这些比率时，成功的请求会比失败的重试耗尽预算的速度更快。

预算的起始余额为500个代币，提供了一个缓冲区，可以吸收短暂的失败。短暂的错误激增，即使是严重的错误，也不会阻止重试，除非它持续足够长的时间以耗尽缓冲区。

实际影响

- 故障率低：配额无效。预算保持在或接近容纳能力。
- 服务中断期间：如果您的请求中有很大一部分持续失败，则配额将耗尽，您的客户端会立即返回错误，而不是等待重试。这可以减少客户端延迟，释放线程和连接，并帮助服务更快地恢复。
- 恢复：当服务恢复并且请求再次开始成功时，成功的重试会恢复其全部代币消耗，首次尝试成功将恢复 1 个令牌。预算会逐渐充满，重试会自动恢复。
- 范围：代币预算通常限于单个 SDK 客户端实例。具体范围可能因 SDK 而异。它不在进程或主机之间共享。

Service-specific 行为

DynamoDB

DynamoDB 客户端使用针对 DynamoDB 的低延迟配置文件进行了优化的调整默认值：

设置	一般默认	DynamoDB 默认
瞬态（非节流）基本延迟	50 毫秒	25 毫秒
限制基本延迟	1,000 ms	1,000 ms
最大尝试次数	3	4

这些默认值适用于亚马逊 DynamoDB 和 DynamoDB Streams。

Long-polling 操作

某些 Amazon 操作使用长轮询。他们可以保持连接处于打开状态，等待工作到达。这些操作会受到特殊的重试处理：

- `SQS.ReceiveMessage`
- `SFN.GetActivityTask`
- `SWF.PollForActivityTask`
- `SWF.PollForDecisionTask`

特殊行为：当重试配额耗尽且重试被阻止（中的第 7 步请求失败时会发生什么）时，SDK 在将错误返回到您的代码之前仍会应用退避延迟。

这很重要，因为长轮询操作通常是在紧密循环中调用的。您的代码会调用`ReceiveMessage`、处理所有消息，然后立即`ReceiveMessage`再次调用。如果不强制退缩，代币预算耗尽将导致 SDK 毫不延迟地返回错误。然后，您的轮询循环将立即发送下一个请求，从而激增客户端 CPU 使用率并产生额外的流量。强制退避延迟打破了这个周期，使客户端资源使用量和轮询率在故障期间保持可控性。

Support by Amazon SDK 和工具

下表列出了每个 SDK 中更新后的重试行为的可用性。 SDK-specific 有关包括最低版本、之前和之后的默认值以及代码示例在内的详细信息，请参阅 GitHub 跟踪问题。

SDK	支持	GitHub 追踪问题
适用于 Java 2.x 的 SDK	是	追踪问题
适用于 Python (Boto3) 的 SDK	是	追踪问题

SDK	支持	GitHub 追踪问题
适用于.NET 4.x 的 SDK	是	追踪问题
适用于 PowerShell V5 的工具	是	追踪问题
适用于 JavaScript 3.x 的软件开发工具包	是	追踪问题
适用于 PHP 3.x 的 SDK	是	追踪问题
适用于 Kotlin 的 SDK	是	追踪问题
适用于 Rust 的 SDK	是	追踪问题
适用于 Swift 的 SDK	查看追踪问题	追踪问题
适用于 Ruby 3.x 的 SDK	查看追踪问题	追踪问题
适用于 Go V2 (1.x) 的 SDK	查看追踪问题	追踪问题
适用于 C++ 的 SDK	查看追踪问题	追踪问题
Amazon CLI v2	查看追踪问题	追踪问题

请求压缩

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

Amazon SDKs 而且，当向接收压缩负载的支持者发送请求时 Amazon Web Services 服务 ，工具可以自动压缩有效负载。在将有效负载发送到服务之前在客户端上对其进行压缩，可以减少向服务发送数据所需的请求总数和带宽，还可以减少由于服务对有效负载大小的限制而导致的失败请求。进行压缩时，SDK 或工具会选择服务和 SDK 都支持的编码算法。但是，当前可能的编码列表仅包含 gzip，但未来可能会扩展。

如果您的应用程序使用的是 Amazon，则请求压缩可能特别有用 CloudWatch。 CloudWatch 是一项监控和可观测性服务，它以日志、指标和事件的形式收集监控和操作数据。支持压缩的服务操作的一个示例是 CloudWatch的 PutMetricDataAPI 方法。

使用以下方法配置此功能：

disable_request_compression-共享 Amazon **config**文件设置, **AWS_DISABLE_REQUEST_COMPRESSION** - 环境变量, **aws.disableRequestCompression**-JVM 系统属性：仅限 Java/Kotlin

开启或关闭 SDK 或工具是否将在发送请求之前压缩有效负载。

默认值：`false`

有效值：

- **true** – 关闭请求压缩。
- **false** – 尽可能使用请求压缩。

request_min_compression_size_bytes-共享 Amazon **config**文件设置, **AWS_REQUEST_MIN_COMPRESSION_SIZE_BYTES** - 环境变量, **aws.requestMinCompressionSizeBytes**-JVM 系统属性：仅限 Java/Kotlin

设置 SDK 或工具应压缩的请求正文的最小大小（以字节为单位）。压缩后，小型有效载荷可能会变得更长，因此，将会有一个下限，使执行压缩变得有意义。该值包含首尾，大于或等于该值的请求大小将被压缩。

默认值：10240 字节

有效值：介于 0 到 10485760 字节（包含首尾）之间的整数值。

Support by Amazon SDKs and 工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	

SDK	支持	备注或更多信息
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	否	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

特定于服务的端点

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

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

使用以下方法配置此功能：

`endpoint_url`-共享 Amazon **`config`**文件设置, **`AWS_ENDPOINT_URL`** - 环境变量, **`aws.endpointUrl`**-JVM 系统属性：仅限 Java/Kotlin

直接在配置文件中指定或作为环境变量指定时，此设置将指定用于所有服务请求的端点。此端点会被任何已配置的特定服务端点覆盖。

您还可以在共享 Amazon `config`文件的某个`services`部分中使用此设置为特定服务设置自定义终端节点。有关 `services` 节的子节中要使用的所有服务标识符密钥的列表，请参阅特定于服务的端点的标识符。

默认值：none

有效值：包含端点架构和主机的 URL。URL 可以选择包含一个路径组件，该组件包括一个或多个路径段。

`AWS_ENDPOINT_URL_<SERVICE>` - 环境变量, **`aws.endpointUrl<ServiceName>`**-JVM 系统属性：仅限 Java/Kotlin

`AWS_ENDPOINT_URL_<SERVICE>`，其中`<SERVICE>`是标 Amazon Web Services 服务 识符，用于为特定服务设置自定义终端节点。有关特定于服务的所有环境变量的列表，请参阅特定于服务的端点的标识符。

此特定服务端点会覆盖 `AWS_ENDPOINT_URL` 中设置的任何全局端点。

默认值：none

有效值：包含端点架构和主机的 URL。URL 可以选择包含一个路径组件，该组件包括一个或多个路径段。

ignore_configured_endpoint_urls-共享 Amazon **config**文件设置, **AWS_IGNORE_CONFIGURED_ENDPOINT_URLS** - 环境变量, **aws.ignoreConfiguredEndpointUrls**-JVM 系统属性：仅限 Java/Kotlin

此设置用于忽略所有自定义端点配置。

请注意，无论此设置如何，都将使用代码中或服务客户端本身上设置的任何显式端点。例如，在`--endpoint-url`命令中包含命令行参数或将端点 URL 传递给客户端构造函数将始终生效。Amazon CLI

默认值：`false`

有效值：

- **true** - SDK 或工具不会从共享 `config` 文件或环境变量中读取任何用于设置端点 URL 的自定义配置选项。
- **false** - SDK 或工具使用共享 `config` 文件或环境变量中用户提供的任何可用端点。

使用环境变量来配置端点

要将所有服务的请求路由到自定义端点 URL，请设置 `AWS_ENDPOINT_URL` 全局环境变量。

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

要将针对特定终端节点 URL 的请求路由 Amazon Web Services 服务 到自定义终端节点 URL，请使用`AWS_ENDPOINT_URL_<SERVICE>`环境变量。 Amazon DynamoDB 有一`serviceId`个DynamoDB。对于此服务，端点 URL 环境变量为 `AWS_ENDPOINT_URL_DYNAMODB`。此端点优先于在 `AWS_ENDPOINT_URL` 中为此服务设置的全局端点。

```
export AWS_ENDPOINT_URL_DYNAMODB=http://localhost:5678
```

再举一个例子， Amazon Elastic Beanstalk 有一`serviceId`个Elastic Beanstalk。标 Amazon Web Services 服务 识符基于 API 模型，将所有空格`serviceId`替换为下划线，并将所有字母大写。为设置适用于此服务的端点，相应的环境变量为 `AWS_ENDPOINT_URL_ELASTIC_BEANSTALK`。有关特定于服务的所有环境变量的列表，请参阅特定于服务的端点的标识符。

```
export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:5567
```

使用共享 config 文件配置端点

在共享 config 文件中，endpoint_url 用于不同位置以实现不同的功能。

- endpoint_url 直接在 profile 中指定会使该端点成为全局端点。
- endpoint_url 嵌套在 services 部分中的服务标识符密钥下，使该端点仅适用于向该服务发出的请求。有关在共享 config 文件中定义 services 节的详细信息，请参阅 配置文件的格式。

以下示例使用 services 定义来配置用于 Amazon S3 的特定于服务的端点 URL 和用于其他所有服务的自定义全局端点：

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

[services s3-specific]
s3 =
  endpoint_url = https://play.min.io:9000
```

单个配置文件可以为多个服务配置端点。此示例说明如何在同一配置文件中为 Amazon S3 设置服务特定的终端节点 URLs 。 Amazon Elastic Beanstalk Amazon Elastic Beanstalk 有一serviceId个Elastic Beanstalk。标 Amazon Web Services 服务 识符基于 API 模型，将所有空格serviceId替换为下划线，并将所有字母小写。因此，服务标识符密钥变为 elastic_beanstalk 且已开始在线设置该服务 elastic_beanstalk = 。有关 services 节中要使用的所有服务标识符密钥的列表，请参阅特定于服务的端点的标识符。

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

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

“服务配置”节可以在多个配置文件中使用。例如，两个配置文件在更改其他配置文件属性时可以使用相同的 services 定义：

```
[services testing-s3]
s3 =
```

```
  endpoint_url = https://localhost:4567

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

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

使用基于角色的凭证在配置文件中配置端点

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

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

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

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

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

设置的优先级

该功能设置为可以同时使用，但每项服务只有一个值会优先使用。对于对给定的 API 调用 Amazon Web Services 服务，使用以下顺序来选择值：

1. 在代码中或服务客户端本身设置的任何显式设置均优先于其他任何设置。

- 对于 Amazon CLI，这是`--endpoint-url`命令行参数提供的值。对于 SDK，显式分配可以采用您在实例化 Amazon Web Services 服务 客户端或配置对象时设置的参数的形式。

2. 由特定于服务的环境变量提供的值，例如 `AWS_ENDPOINT_URL_DYNAMODB`。
3. `AWS_ENDPOINT_URL` 全局端点环境变量提供的值。
4. 该 `endpoint_url` 设置提供的值嵌套在共享 `config` 文件的 `services` 部分中的服务标识符密钥下。
5. 共享 `config` 文件的 `profile` 中直接指定的 `endpoint_url` 设置提供的值。
6. 最后使用相应 Amazon Web Services 服务 端点的所有默认端点 URL。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	是	
适用于 C++ 的 SDK	是	
适用于 Go V2 (1.x) 的 SDK	是	
适用于 Go 1.x (V1) 的 SDK	否	
适用于 Java 2.x 的 SDK	是	
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	是	
适用于.NET 4.x 的 SDK	是	

SDK	支持	备注或更多信息
适用于 .NET 3.x 的 SDK	是	
适用于 PHP 3.x 的 SDK	是	
适用于 Python (Boto3) 的 SDK	是	
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	是	
适用于 Swift 的 SDK	是	
适用于 PowerShell V5 的工具	是	
适用于 PowerShell V4 的工具	是	

特定于服务的端点的标识符

有关如何以及在何处使用下表中的标识符的信息，请参阅 特定于服务的端点。

serviceId	共享 Aconfig 文件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
AccessAnalyzer	ly	AWS_ENDPOINT_URL_ACCESSANALYZER
Account	ac	AWS_ENDPOINT_URL_ACCOUNT
ACM	ac	AWS_ENDPOINT_URL_ACM
ACM PCA	ac	AWS_ENDPOINT_URL_ACM_PCA
Alexa For Business	a_	AWS_ENDPOINT_URL_ALEXA_FOR_BUSINESS
amp	ar	AWS_ENDPOINT_URL_AMP
Amplify	ar	AWS_ENDPOINT_URL_AMPLIFY
AmplifyBackend	ar cl	AWS_ENDPOINT_URL_AMPLIFYBACKEND
AmplifyUIBuilder	ar bu	AWS_ENDPOINT_URL_AMPLIFYUIBUILDER
API Gateway	ap ay	AWS_ENDPOINT_URL_API_GATEWAY

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
ApiGatewayManagementApi	ap yr nt	AWS_ENDPOINT_URL_APIGATEWAYMANAGEMENTAPI
ApiGatewayV2	ap yv	AWS_ENDPOINT_URL_APIGATEWAYV2
AppConfig	ap	AWS_ENDPOINT_URL_APPCONFIG
AppConfigData	ap da	AWS_ENDPOINT_URL_APPCONFIGDATA
AppFabric	ap	AWS_ENDPOINT_URL_APPFABRIC
Appflow	ap	AWS_ENDPOINT_URL_APPFLOW
AppIntegrations	ap at	AWS_ENDPOINT_URL_APPINTEGRATIONS
Application Auto Scaling	ap on ca	AWS_ENDPOINT_URL_APPLICATION_AUTO_SCALING

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Application Insights	ap or ts	AWS_ENDPOINT_URL_APPLICATION_INSIGHTS
ApplicationCostProfiler	ap or f:	AWS_ENDPOINT_URL_APPLICATIONCOSTPROFILER
App Mesh	ap	AWS_ENDPOINT_URL_APP_MESH
AppRunner	ap	AWS_ENDPOINT_URL_APPRUNNER
AppStream	ap	AWS_ENDPOINT_URL_APPSTREAM
AppSync	ap	AWS_ENDPOINT_URL_APPSYNC
ARC Zonal Shift	a: _s	AWS_ENDPOINT_URL_ARC_ZONAL_SHIFT
Artifact	a:	AWS_ENDPOINT_URL_ARTIFACT
Athena	at	AWS_ENDPOINT_URL_ATHENA

serviceId	共享 Au ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
AuditManager	au ge	AWS_ENDPOINT_URL_AUDITMANAGER
Auto Scaling	au in	AWS_ENDPOINT_URL_AUTO_SCALING
Auto Scaling Plans	au in	AWS_ENDPOINT_URL_AUTO_SCALING_PLANS
b2bi	b2	AWS_ENDPOINT_URL_B2BI
Backup	ba	AWS_ENDPOINT_URL_BACKUP
Backup Gateway	ba te	AWS_ENDPOINT_URL_BACKUP_GATEWAY
BackupStorage	ba ra	AWS_ENDPOINT_URL_BACKUPSTORAGE
Batch	ba	AWS_ENDPOINT_URL_BATCH
BCM Data Exports	bc ex	AWS_ENDPOINT_URL_BCM_DATA_EXPORTS
Bedrock	be	AWS_ENDPOINT_URL_BEDROCK

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Bedrock Agent	be ge	AWS_ENDPOINT_URL_BEDROCK_AGENT
Bedrock Agent Runtime	be ge in	AWS_ENDPOINT_URL_BEDROCK_AGENT_RUNTIME
Bedrock Runtime	be ur	AWS_ENDPOINT_URL_BEDROCK_RUNTIME
billingconductor	bi no	AWS_ENDPOINT_URL_BILLINGCONDUCTOR
Braket	br	AWS_ENDPOINT_URL_BRAKET
Budgets	bu	AWS_ENDPOINT_URL_BUDGETS
Cost Explorer	ce ox	AWS_ENDPOINT_URL_COST_EXPLORER
chatbot	ch	AWS_ENDPOINT_URL_CHATBOT
Chime	ch	AWS_ENDPOINT_URL_CHIME

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Chime SDK Identity	ch _:	AWS_ENDPOINT_URL_CHIME_SDK_IDENTITY
Chime SDK Media Pipelines	ch _r pe	AWS_ENDPOINT_URL_CHIME_SDK_MEDIA_PIPELINES
Chime SDK Meetings	ch _r	AWS_ENDPOINT_URL_CHIME_SDK_MEETINGS
Chime SDK Messaging	ch _r g	AWS_ENDPOINT_URL_CHIME_SDK_MESSAGING
Chime SDK Voice	ch _\	AWS_ENDPOINT_URL_CHIME_SDK_VOICE
CleanRooms	c: s	AWS_ENDPOINT_URL_CLEANROOMS
CleanRoomsML	c: sr	AWS_ENDPOINT_URL_CLEANROOMSML

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Cloud9	cl	AWS_ENDPOINT_URL_CLOUD9
CloudControl	cl rc	AWS_ENDPOINT_URL_CLOUDCONTROL
CloudDirectory	cl cf	AWS_ENDPOINT_URL_CLOUDDIRECTORY
CloudFormation	cl af	AWS_ENDPOINT_URL_CLOUDFORMATION
CloudFront	cl t	AWS_ENDPOINT_URL_CLOUDFRONT
CloudFront KeyValueStore	cl t_ es	AWS_ENDPOINT_URL_CLOUDFRONT_KEYVALUESTORE
CloudHSM	cl	AWS_ENDPOINT_URL_CLOUDHSM
CloudHSM V2	cl v2	AWS_ENDPOINT_URL_CLOUDHSM_V2
CloudSearch	cl ch	AWS_ENDPOINT_URL_CLOUDSEARCH

serviceId	共享 Aw **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
CloudSearch Domain	cl ch	AWS_ENDPOINT_URL_CLOUDSEARCH_DOMAIN
CloudTrail	cl l	AWS_ENDPOINT_URL_CLOUDTRAIL
CloudTrail Data	cl l_	AWS_ENDPOINT_URL_CLOUDTRAIL_DATA
CloudWatch	cl h	AWS_ENDPOINT_URL_CLOUDWATCH
codeartifact	co ac	AWS_ENDPOINT_URL_CODEARTIFACT
CodeBuild	co	AWS_ENDPOINT_URL_CODEBUILD
CodeCatalyst	co ys	AWS_ENDPOINT_URL_CODECATALYST
CodeCommit	co t	AWS_ENDPOINT_URL_CODECOMMIT

serviceId	共享 Ar co 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
CodeDeploy	co y	AWS_ENDPOINT_URL_CODEDEPLOY
CodeGuru Reviewer	co re	AWS_ENDPOINT_URL_CODEGURU_REVIEWER
CodeGuru Security	co se	AWS_ENDPOINT_URL_CODEGURU_SECURITY
CodeGuruProfiler	co ro	AWS_ENDPOINT_URL_CODEGURUPROFILER
CodePipeline	co in	AWS_ENDPOINT_URL_CODEPIPELINE
CodeStar	co	AWS_ENDPOINT_URL_CODESTAR
CodeStar connections	co co ns	AWS_ENDPOINT_URL_CODESTAR_CONNECTIONS
codestar notifications	co no ic	AWS_ENDPOINT_URL_CODESTAR_NOTIFICATIONS

serviceId	**共享 AWS config 文件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
Cognito Identity	cognito_identity	AWS_ENDPOINT_URL_COGNITO_IDENTITY
Cognito Identity Provider	cognito_identity_provider	AWS_ENDPOINT_URL_COGNITO_IDENTITY_PROVIDER
Cognito Sync	cognito_sync	AWS_ENDPOINT_URL_COGNITO_SYNC
Comprehend	comprehend	AWS_ENDPOINT_URL_COMPREHEND
ComprehendMedical	comprehendmedical	AWS_ENDPOINT_URL_COMPREHENDMEDICAL
Compute Optimizer	compute_optimizer	AWS_ENDPOINT_URL_COMPUTE_OPTIMIZER
Config Service	config_service	AWS_ENDPOINT_URL_CONFIG_SERVICE
Connect	connect	AWS_ENDPOINT_URL_CONNECT

serviceId	**共享 Ar ce 件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
Connect Contact Lens	co on ns	AWS_ENDPOINT_URL_CONNECT_CONTACT_LENS
ConnectCampaigns	co mp	AWS_ENDPOINT_URL_CONNECTCAMPAIGNS
ConnectCases	co se	AWS_ENDPOINT_URL_CONNECTCASES
ConnectParticipant	co rt	AWS_ENDPOINT_URL_CONNECTPARTICIPANT
ControlTower	co we	AWS_ENDPOINT_URL_CONTROLTOWER
Cost Optimization Hub	co mi hu	AWS_ENDPOINT_URL_COST_OPTIMIZATION_HUB
Cost and Usage Report Service	co us oi ce	AWS_ENDPOINT_URL_COST_AND_USAGE_REPO RT_SERVICE

serviceId	**共享 Ar ce 件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
Customer Profiles	cu p:	AWS_ENDPOINT_URL_CUSTOMER_PROFILES
DataBrew	da	AWS_ENDPOINT_URL_DATABREW
DataExchange	da ng	AWS_ENDPOINT_URL_DATAEXCHANGE
Data Pipeline	da l:	AWS_ENDPOINT_URL_DATA_PIPELINE
DataSync	da	AWS_ENDPOINT_URL_DATASYNC
DataZone	da	AWS_ENDPOINT_URL_DATAZONE
DAX	da	AWS_ENDPOINT_URL_DAX
Detective	de	AWS_ENDPOINT_URL_DETECTIVE
Device Farm	de rn	AWS_ENDPOINT_URL_DEVICE_FARM
DevOps Guru	de ru	AWS_ENDPOINT_URL_DEVOPS_GURU

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Direct Connect	d: nı	AWS_ENDPOINT_URL_DIRECT_CONNECT
Application Discovery Service	ap or e: ce	AWS_ENDPOINT_URL_APPLICATION_DISCOVERY_SERVICE
DLM	d:	AWS_ENDPOINT_URL_DLM
Database Migration Service	da m: _s	AWS_ENDPOINT_URL_DATABASE_MIGRATION_SERVICE
DocDB	dc	AWS_ENDPOINT_URL_DOCDB
DocDB Elastic	dc st	AWS_ENDPOINT_URL_DOCDB_ELASTIC
drs	d:	AWS_ENDPOINT_URL_DRS
Directory Service	d: _s	AWS_ENDPOINT_URL_DIRECTORY_SERVICE
DynamoDB	dy	AWS_ENDPOINT_URL_DYNAMODB

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
DynamoDB Streams	dy st	AWS_ENDPOINT_URL_DYNAMODB_STREAMS
EBS	eb	AWS_ENDPOINT_URL_EBS
EC2	ec	AWS_ENDPOINT_URL_EC2
EC2 Instance Connect	ec nc ct	AWS_ENDPOINT_URL_EC2_INSTANCE_CONNECT
ECR	ec	AWS_ENDPOINT_URL_ECR
ECR PUBLIC	ec c	AWS_ENDPOINT_URL_ECR_PUBLIC
ECS	ec	AWS_ENDPOINT_URL_ECS
EFS	ef	AWS_ENDPOINT_URL_EFS
EKS	ek	AWS_ENDPOINT_URL_EKS
EKS Auth	ek	AWS_ENDPOINT_URL_EKS_AUTH
Elastic Inference	el nf	AWS_ENDPOINT_URL_ELASTIC_INFERENCE

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
ElastiCache	el he	AWS_ENDPOINT_URL_ELASTICACHE
Elastic Beanstalk	el ea	AWS_ENDPOINT_URL_ELASTIC_BEANSTALK
Elastic Transcoder	el ra	AWS_ENDPOINT_URL_ELASTIC_TRANSCODER
Elastic Load Balancing	el oa c:	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING
Elastic Load Balancing v2	el oa c:	AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING_V2
EMR	em	AWS_ENDPOINT_URL_EMR
EMR containers	em in	AWS_ENDPOINT_URL_EMR_CONTAINERS
EMR Serverless	em r:	AWS_ENDPOINT_URL_EMR_SERVERLESS

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
EntityResolution	er o:	AWS_ENDPOINT_URL_ENTITYRESOLUTION
Elasticsearch Service	e: a: i	AWS_ENDPOINT_URL_ELASTICSEARCH_SERVICE
EventBridge	ev ge	AWS_ENDPOINT_URL_EVENTBRIDGE
Evidently	ev	AWS_ENDPOINT_URL_EVIDENTLY
finspace	f:	AWS_ENDPOINT_URL_FINSPACE
finspace data	f: da	AWS_ENDPOINT_URL_FINSPACE_DATA
Firehose	f:	AWS_ENDPOINT_URL_FIREHOSE
fis	f:	AWS_ENDPOINT_URL_FIS
FMS	fr	AWS_ENDPOINT_URL_FMS
forecast	fc	AWS_ENDPOINT_URL_FORECAST

serviceId	**共享 Ar ce 件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
forecastquery	fo ue	AWS_ENDPOINT_URL_FORECASTQUERY
FraudDetector	fr ct	AWS_ENDPOINT_URL_FRAUDDETECTOR
FreeTier	fr	AWS_ENDPOINT_URL_FREETIER
FSx	fs	AWS_ENDPOINT_URL_FSX
GameLift	ga	AWS_ENDPOINT_URL_GAMELIFT
Glacier	gl	AWS_ENDPOINT_URL_GLACIER
Global Accelerator	gl ce	AWS_ENDPOINT_URL_GLOBAL_ACCELERATOR
Glue	gl	AWS_ENDPOINT_URL_GLUE
grafana	gr	AWS_ENDPOINT_URL_GRAFANA
Greengrass	gr s	AWS_ENDPOINT_URL_GREENGRASS

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
GreengrassV2	gr sv	AWS_ENDPOINT_URL_GREENGRASSV2
GroundStation	gr ti	AWS_ENDPOINT_URL_GROUNDSTATION
GuardDuty	gu	AWS_ENDPOINT_URL_GUARDDUTY
Health	he	AWS_ENDPOINT_URL_HEALTH
HealthLake	he e	AWS_ENDPOINT_URL_HEALTHLAKE
Honeycode	ho	AWS_ENDPOINT_URL_HONEYCODE
IAM	ia	AWS_ENDPOINT_URL_IAM
identitystore	id to	AWS_ENDPOINT_URL_IDENTITYSTORE
imagebuilder	im de	AWS_ENDPOINT_URL_IMAGEBUILDER

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
ImportExport	ir o:	AWS_ENDPOINT_URL_IMPORTEXPORT
Inspector	ir	AWS_ENDPOINT_URL_INSPECTOR
Inspector Scan	ir _s	AWS_ENDPOINT_URL_INSPECTOR_SCAN
Inspector2	ir 2	AWS_ENDPOINT_URL_INSPECTOR2
InternetMonitor	ir or	AWS_ENDPOINT_URL_INTERNETMONITOR
IoT	io	AWS_ENDPOINT_URL_IOT
IoT Data Plane	io p:	AWS_ENDPOINT_URL_IOT_DATA_PLANE
IoT Jobs Data Plane	io da e	AWS_ENDPOINT_URL_IOT_JOBS_DATA_PLANE

serviceId	**共享 Ar ce 件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
IoT 1Click Devices Service	io k_ _s	AWS_ENDPOINT_URL_IOT_1CLICK_DEVICES_ SERVICE
IoT 1Click Projects	io k_ s	AWS_ENDPOINT_URL_IOT_1CLICK_PROJECTS
IoTAnalytics	io io	AWS_ENDPOINT_URL_IOTANALYTICS
IotDeviceAdvisor	io ad	AWS_ENDPOINT_URL_IOTDEVICEADVISOR
IoT Events	io s	AWS_ENDPOINT_URL_IOT_EVENTS
IoT Events Data	io s_	AWS_ENDPOINT_URL_IOT_EVENTS_DATA
IoTFleetHub	io ub	AWS_ENDPOINT_URL_IOTFLEETHUB
IoTFleetWise	io is	AWS_ENDPOINT_URL_IOTFLEETWISE

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
IoTSecureTunneling	io tu	AWS_ENDPOINT_URL_IOTSECURETUNNELING
IoTSiteWise	io se	AWS_ENDPOINT_URL_IOTSITEWISE
IoTThingsGraph	io gr	AWS_ENDPOINT_URL_IOTTHINGSGRAPH
IoTTwinMaker	io ke	AWS_ENDPOINT_URL_IOTTWINMAKER
IoT Wireless	io es	AWS_ENDPOINT_URL_IOT_WIRELESS
ivs	iv	AWS_ENDPOINT_URL_IVS
IVS RealTime	iv in	AWS_ENDPOINT_URL_IVS_REALTIME
ivschat	iv	AWS_ENDPOINT_URL_IVSCHAT
Kafka	ka	AWS_ENDPOINT_URL_KAFKA

serviceId	共享 Au config 文件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
KafkaConnect	ka ec	AWS_ENDPOINT_URL_KAFKACONNECT
kendra	ke	AWS_ENDPOINT_URL_KENDRA
Kendra Ranking	ke nl	AWS_ENDPOINT_URL_KENDRA_RANKING
Keyspaces	ke	AWS_ENDPOINT_URL_KEYSPACES
Kinesis	ki	AWS_ENDPOINT_URL_KINESIS
Kinesis Video Archived Media	ki id iv a	AWS_ENDPOINT_URL_KINESIS_VIDEO_ARCHIVED_MEDIA
Kinesis Video Media	ki id a	AWS_ENDPOINT_URL_KINESIS_VIDEO_MEDIA
Kinesis Video Signaling	ki id a:	AWS_ENDPOINT_URL_KINESIS_VIDEO_SIGNALING

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Kinesis Video WebRTC Storage	ki ic te e	AWS_ENDPOINT_URL_KINESIS_VIDEO_WEBRTC_STORAGE
Kinesis Analytics	ki na	AWS_ENDPOINT_URL_KINESIS_ANALYTICS
Kinesis Analytics V2	ki na v2	AWS_ENDPOINT_URL_KINESIS_ANALYTICS_V2
Kinesis Video	ki ic	AWS_ENDPOINT_URL_KINESIS_VIDEO
KMS	km	AWS_ENDPOINT_URL_KMS
LakeFormation	la ti	AWS_ENDPOINT_URL_LAKEFORMATION
Lambda	la	AWS_ENDPOINT_URL_LAMBDA
Launch Wizard	la za	AWS_ENDPOINT_URL_LAUNCH_WIZARD

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Lex Model Building Service	le _l _s	AWS_ENDPOINT_URL_LEX_MODEL_BUILDING_ SERVICE
Lex Runtime Service	le me e	AWS_ENDPOINT_URL_LEX_RUNTIME_SERVICE
Lex Models V2	le s_	AWS_ENDPOINT_URL_LEX_MODELS_V2
Lex Runtime V2	le me	AWS_ENDPOINT_URL_LEX_RUNTIME_V2
License Manager	li ar	AWS_ENDPOINT_URL_LICENSE_MANAGER
License Manager Linux Subscriptions	li ar nu r:	AWS_ENDPOINT_URL_LICENSE_MANAGER_LIN UX_SUBSCRIPTIONS

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
License Manager User Subscriptions	l: ar e: ip	AWS_ENDPOINT_URL_LICENSE_MANAGER_USER_SUBSCRIPTIONS
Lightsail	l:	AWS_ENDPOINT_URL_LIGHTSAIL
Location	l(	AWS_ENDPOINT_URL_LOCATION
CloudWatch Logs	c: h_	AWS_ENDPOINT_URL_CLOUDWATCH_LOGS
LookoutEquipment	l(u:	AWS_ENDPOINT_URL_LOOKOUTEQUIPMENT
LookoutMetrics	l(t:	AWS_ENDPOINT_URL_LOOKOUTMETRICS
LookoutVision	l(s:	AWS_ENDPOINT_URL_LOOKOUTVISION
m2	m:	AWS_ENDPOINT_URL_M2

serviceId	共享 Ar ce 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Machine Learning	ma ea	AWS_ENDPOINT_URL_MACHINE_LEARNING
Macie2	ma	AWS_ENDPOINT_URL_MACIE2
ManagedBlockchain	ma oc	AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN
ManagedBlockchain Query	ma oc qu	AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN_QUERY
Marketplace Agreement	ma ce er	AWS_ENDPOINT_URL_MARKETPLACE_AGREEMENT
Marketplace Catalog	ma ce g	AWS_ENDPOINT_URL_MARKETPLACE_CATALOG
Marketplace Deployment	ma ce me	AWS_ENDPOINT_URL_MARKETPLACE_DEPLOYMENT

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Marketplace Entitlement Service	ma ce en v:	AWS_ENDPOINT_URL_MARKETPLACE_ENTITLEMENT_SERVICE
Marketplace Commerce Analytics	ma ce ce ic	AWS_ENDPOINT_URL_MARKETPLACE_COMMERCE_ANALYTICS
MediaConnect	me ec	AWS_ENDPOINT_URL_MEDIACONNECT
MediaConvert	me e:	AWS_ENDPOINT_URL_MEDIACONVERT
MediaLive	me	AWS_ENDPOINT_URL_MEDIALIVE
MediaPackage	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGE
MediaPackage Vod	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGE_VOD

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
MediaPackageV2	me ag	AWS_ENDPOINT_URL_MEDIAPACKAGEV2
MediaStore	me e	AWS_ENDPOINT_URL_MEDIASTORE
MediaStore Data	me e_	AWS_ENDPOINT_URL_MEDIASTORE_DATA
MediaTailor	me o:	AWS_ENDPOINT_URL_MEDIATAILOR
Medical Imaging	me ma	AWS_ENDPOINT_URL_MEDICAL_IMAGING
MemoryDB	me	AWS_ENDPOINT_URL_MEMORYDB
Marketplace Metering	ma ce ng	AWS_ENDPOINT_URL_MARKETPLACE_METERING
Migration Hub	m: _h	AWS_ENDPOINT_URL_MIGRATION_HUB
mgn	mg	AWS_ENDPOINT_URL_MGN

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Migration Hub Refactor Spaces	m _I ct es	AWS_ENDPOINT_URL_MIGRATION_HUB_REFAC TOR_SPACES
MigrationHub Config	m hu g	AWS_ENDPOINT_URL_MIGRATIONHUB_CONFIG
MigrationHubOrches trator	m hu t:	AWS_ENDPOINT_URL_MIGRATIONHUBORCHESTRATOR
MigrationHubStrategy	m hu gy	AWS_ENDPOINT_URL_MIGRATIONHUBSTRATEGY
Mobile	mo	AWS_ENDPOINT_URL_MOBILE
mq	mo	AWS_ENDPOINT_URL_MQ
MTurk	mt	AWS_ENDPOINT_URL_MTURK
MWAA	mw	AWS_ENDPOINT_URL_MWAA

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Neptune	ne	AWS_ENDPOINT_URL_NEPTUNE
Neptune Graph	ne ra	AWS_ENDPOINT_URL_NEPTUNE_GRAPH
neptunedata	ne ta	AWS_ENDPOINT_URL_NEPTUNEDATA
Network Firewall	ne i:	AWS_ENDPOINT_URL_NETWORK_FIREWALL
NetworkManager	ne na	AWS_ENDPOINT_URL_NETWORKMANAGER
NetworkMonitor	ne n:	AWS_ENDPOINT_URL_NETWORKMONITOR
nimble	n:	AWS_ENDPOINT_URL_NIMBLE
OAM	oa	AWS_ENDPOINT_URL_OAM
Omics	or	AWS_ENDPOINT_URL_OMICS
OpenSearch	op h	AWS_ENDPOINT_URL_OPENSEARCH

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
OpenSearchServerless	op hs ss	AWS_ENDPOINT_URL_OPENSEARCHSERVERLESS
OpsWorks	op	AWS_ENDPOINT_URL_OPSWORKS
OpsWorksCM	op m	AWS_ENDPOINT_URL_OPSWORKSCM
Organizations	or id	AWS_ENDPOINT_URL_ORGANIZATIONS
OSIS	os	AWS_ENDPOINT_URL_OSIS
Outposts	ou	AWS_ENDPOINT_URL_OUTPOSTS
p8data	p8	AWS_ENDPOINT_URL_P8DATA
p8data	p8	AWS_ENDPOINT_URL_P8DATA
Panorama	pa	AWS_ENDPOINT_URL_PANORAMA
Payment Cryptography	pa ry hy	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
Payment Cryptography Data	pa ry hy	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY_DATA
Pca Connector Ad	pc ct	AWS_ENDPOINT_URL_PCA_CONNECTOR_AD
Personalize	pe ze	AWS_ENDPOINT_URL_PERSONALIZE
Personalize Events	pe ze	AWS_ENDPOINT_URL_PERSONALIZE_EVENTS
Personalize Runtime	pe ze e	AWS_ENDPOINT_URL_PERSONALIZE_RUNTIME
PI	pi	AWS_ENDPOINT_URL_PI
Pinpoint	pi	AWS_ENDPOINT_URL_PINPOINT
Pinpoint Email	pi em	AWS_ENDPOINT_URL_PINPOINT_EMAIL

serviceId	**共享 Ar ce 件的服务标识密钥**	**AWS_ENDPOINT_URL_<SERVICE>　环境变量**
Pinpoint SMS Voice	p: sr	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE
Pinpoint SMS Voice V2	p: sr _\	AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE_V2
Pipes	p:	AWS_ENDPOINT_URL_PIPES
Polly	p(	AWS_ENDPOINT_URL_POLLY
Pricing	p:	AWS_ENDPOINT_URL_PRICING
PrivateNetworks	p: tv	AWS_ENDPOINT_URL_PRIVATENETWORKS
Proton	p:	AWS_ENDPOINT_URL_PROTON
QBusiness	qb	AWS_ENDPOINT_URL_QBUSINESS
QConnect	q(	AWS_ENDPOINT_URL_QCONNECT
QLDB	q]	AWS_ENDPOINT_URL_QLDB

serviceId	共享 Ar ce 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
QLDB Session	q: ic	AWS_ENDPOINT_URL_QLDB_SESSION
QuickSight	qu t	AWS_ENDPOINT_URL_QUICKSIGHT
RAM	ra	AWS_ENDPOINT_URL_RAM
rbin	rb	AWS_ENDPOINT_URL_RBIN
RDS	rd	AWS_ENDPOINT_URL_RDS
RDS Data	rd	AWS_ENDPOINT_URL_RDS_DATA
Redshift	re	AWS_ENDPOINT_URL_REDSHIFT
Redshift Data	re da	AWS_ENDPOINT_URL_REDSHIFT_DATA
Redshift Serverless	re se s	AWS_ENDPOINT_URL_REDSHIFT_SERVERLESS
Rekognition	re or	AWS_ENDPOINT_URL_REKOGNITION

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
repostspace	re ce	AWS_ENDPOINT_URL_REPOSTSPACE
resiliencehub	re eh	AWS_ENDPOINT_URL_RESILIENCEHUB
Resource Explorer 2	re ex 2	AWS_ENDPOINT_URL_RESOURCE_EXPLORER_2
Resource Groups	re gr	AWS_ENDPOINT_URL_RESOURCE_GROUPS
Resource Groups Tagging API	re gr gg	AWS_ENDPOINT_URL_RESOURCE_GROUPS_TAGGING_API
RoboMaker	ro	AWS_ENDPOINT_URL_ROBOMAKER
RolesAnywhere	ro he	AWS_ENDPOINT_URL_ROLESANYWHERE
Route 53	ro	AWS_ENDPOINT_URL_ROUTE_53

serviceId	共享 Ar co 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Route53 Recovery Cluster	rc ec lu	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CLUSTER
Route53 Recovery Control Config	rc ec or nt	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CONTROL_CONFIG
Route53 Recovery Readiness	rc ec ea	AWS_ENDPOINT_URL_ROUTE53_RECOVERY_READINESS
Route 53 Domains	rc do	AWS_ENDPOINT_URL_ROUTE_53_DOMAINS
Route53Resolver	rc so	AWS_ENDPOINT_URL_ROUTE53RESOLVER
RUM	ru	AWS_ENDPOINT_URL_RUM
S3	s3	AWS_ENDPOINT_URL_S3
S3 Control	s3 l	AWS_ENDPOINT_URL_S3_CONTROL

serviceId	共享 Au ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
S3Outposts	s3 s	AWS_ENDPOINT_URL_S3OUTPOSTS
SageMaker	sa	AWS_ENDPOINT_URL_SAGEMAKER
SageMaker A2I Runtime	sa _a in	AWS_ENDPOINT_URL_SAGEMAKER_A2I_RUNTIME
Sagemaker Edge	sa _e	AWS_ENDPOINT_URL_SAGEMAKER_EDGE
SageMaker FeatureStore Runtime	sa _f to in	AWS_ENDPOINT_URL_SAGEMAKER_FEATUREST ORE_RUNTIME
SageMaker Geospatial	sa _g a	AWS_ENDPOINT_URL_SAGEMAKER_GEOSPATIAL
SageMaker Metrics	sa _m	AWS_ENDPOINT_URL_SAGEMAKER_METRICS

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
SageMaker Runtime	sa _	AWS_ENDPOINT_URL_SAGEMAKER_RUNTIME
savingsplans	sa ar	AWS_ENDPOINT_URL_SAVINGSPLANS
Scheduler	sc	AWS_ENDPOINT_URL_SCHEDULER
schemas	sc	AWS_ENDPOINT_URL_SCHEMAS
SimpleDB	sd	AWS_ENDPOINT_URL_SIMPLEDB
Secrets Manager	se ar	AWS_ENDPOINT_URL_SECRETS_MANAGER
SecurityHub	se ub	AWS_ENDPOINT_URL_SECURITYHUB
SecurityLake	se al	AWS_ENDPOINT_URL_SECURITYLAKE

serviceId	共享 Ar **co** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
ServerlessApplicationRepository	se sa io to	AWS_ENDPOINT_URL_SERVERLESSAPPLICATIONREPOSITORY
Service Quotas	se uo	AWS_ENDPOINT_URL_SERVICE_QUOTAS
Service Catalog	se at	AWS_ENDPOINT_URL_SERVICE_CATALOG
Service Catalog AppRegistry	se at pi	AWS_ENDPOINT_URL_SERVICE_CATALOG_APPREGISTRY
ServiceDiscovery	se sc	AWS_ENDPOINT_URL_SERVICEDISCOVERY
SES	se	AWS_ENDPOINT_URL_SES
SESv2	se	AWS_ENDPOINT_URL_SESV2
Shield	sh	AWS_ENDPOINT_URL_SHIELD

serviceId	共享 Ar **ce** 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
signer	s:	AWS_ENDPOINT_URL_SIGNER
SimSpaceWeaver	s: ea	AWS_ENDPOINT_URL_SIMSPACEWEAVER
SMS	sr	AWS_ENDPOINT_URL_SMS
Snow Device Management	sr ce me	AWS_ENDPOINT_URL_SNOW_DEVICE_MANAGEMENT
Snowball	sr	AWS_ENDPOINT_URL_SNOWBALL
SNS	sr	AWS_ENDPOINT_URL_SNS
SQS	sc	AWS_ENDPOINT_URL_SQS
SSM	ss	AWS_ENDPOINT_URL_SSM
SSM Contacts	ss ct	AWS_ENDPOINT_URL_SSM_CONTACTS
SSM Incidents	ss er	AWS_ENDPOINT_URL_SSM_INCIDENTS
Ssm Sap	ss	AWS_ENDPOINT_URL_SSM_SAP

serviceId	共享 Ac 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
SSO	ss	AWS_ENDPOINT_URL_SSO
SSO Admin	ss	AWS_ENDPOINT_URL_SSO_ADMIN
SSO OIDC	ss	AWS_ENDPOINT_URL_SSO_OIDC
SFN	st	AWS_ENDPOINT_URL_SFN
Storage Gateway	st at	AWS_ENDPOINT_URL_STORAGE_GATEWAY
STS	st	AWS_ENDPOINT_URL_STS
SupplyChain	su in	AWS_ENDPOINT_URL_SUPPLYCHAIN
Support	su	AWS_ENDPOINT_URL_SUPPORT
Support App	su pp	AWS_ENDPOINT_URL_SUPPORT_APP
SWF	sw	AWS_ENDPOINT_URL_SWF
synthetics	sy s	AWS_ENDPOINT_URL_SYNTHETICS

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
Textract	te	AWS_ENDPOINT_URL_TEXTRACT
Timestream InfluxDB	t: m_ b	AWS_ENDPOINT_URL_TIMESTREAM_INFLUXDB
Timestream Query	t: m_	AWS_ENDPOINT_URL_TIMESTREAM_QUERY
Timestream Write	t: m_	AWS_ENDPOINT_URL_TIMESTREAM_WRITE
tnb	tr	AWS_ENDPOINT_URL_TNB
Transcribe	t: e	AWS_ENDPOINT_URL_TRANSCRIBE
Transfer	t:	AWS_ENDPOINT_URL_TRANSFER
Translate	t:	AWS_ENDPOINT_URL_TRANSLATE
TrustedAdvisor	t: v:	AWS_ENDPOINT_URL_TRUSTEDADVISOR

serviceId	共享 Ar ce 件的服务标识密钥	AWS_ENDPOINT_URL_<SERVICE> 环境变量
VerifiedPermissions	ve es	AWS_ENDPOINT_URL_VERIFIEDPERMISSIONS
Voice ID	vo	AWS_ENDPOINT_URL_VOICE_ID
VPC Lattice	vp ce	AWS_ENDPOINT_URL_VPC_LATTICE
WAF	wa	AWS_ENDPOINT_URL_WAF
WAF Regional	wa na	AWS_ENDPOINT_URL_WAF_REGIONAL
WAFV2	wa	AWS_ENDPOINT_URL_WAFV2
WellArchitected	we te	AWS_ENDPOINT_URL_WELLARCHITECTED
Wisdom	wi	AWS_ENDPOINT_URL_WISDOM
WorkDocs	wo	AWS_ENDPOINT_URL_WORKDOCS
WorkLink	wo	AWS_ENDPOINT_URL_WORKLINK
WorkMail	wo	AWS_ENDPOINT_URL_WORKMAIL

serviceId	共享 Ar ce 件的服务标识密钥	**AWS_ENDPOINT_URL_<SERVICE>** 环境变量
WorkMailMessageFlow	wo es w	AWS_ENDPOINT_URL_WORKMAILMESSAGEFLOW
WorkSpaces	wo s	AWS_ENDPOINT_URL_WORKSPACES
WorkSpaces Thin Client	wo s_ ie	AWS_ENDPOINT_URL_WORKSPACES_THIN_CLIENT
WorkSpaces Web	wo s_	AWS_ENDPOINT_URL_WORKSPACES_WEB
XRay	x:	AWS_ENDPOINT_URL_XRAY

智能配置默认值

> **Note**
>
> 如需了解设置页面布局或解释后面的 Support b y Amazon SDKs 和 tools 表格的帮助，请参阅了解本指南的设置页面。

借助智能配置默认值功能， Amazon SDKs 可以为其他配置设置提供预定义的、优化的默认值。

使用以下方法配置此功能：

`defaults_mode`-共享 Amazon **`config`**文件设置, **`AWS_DEFAULTS_MODE`** - 环境变量, **`aws.defaultsMode`**-JVM 系统属性：仅限 Java/Kotlin

使用此设置，您可以选择与您的应用程序架构相匹配的模式，然后为您的应用程序提供经过优化的默认值。如果 S Amazon DK 设置明确设置了值，则该值始终优先。如果 S Amazon DK 设置没有明确设置值，并且`defaults_mode`不等于旧版，则此功能可以为针对您的应用程序优化的各种设置提供不同的默认值。设置可能包括以下内容：HTTP 通信设置、重试行为、服务区域端点设置，可能还包括任何与 SDK 相关的配置。使用此功能的客户可以获得针对常见使用场景量身定制的新配置默认值。由于提供的默认值可能会随着最佳实践的发展而改变，因此如果您的 `defaults_mode` 不等于 `legacy`，我们建议您在升级 SDK 时对您的应用程序进行测试。

默认值：`legacy`

注意：will 的新主要 SDKs 版本默认为`standard`。

有效值：

- `legacy` – 提供默认设置，这些设置因 SDK 而异，并且在建立 `defaults_mode` 之前就已存在。
- `standard` – 提供最新的推荐默认值，这些默认值在大多数情况下都应该可以安全运行。
- `in-region`— 基于标准模式构建，包括为 Amazon Web Services 服务 从标准模式内部调用的应用程序量身定制的优化 Amazon Web Services 区域。
- `cross-region`— 基于标准模式构建，包括为调用不同区域的应用程序量身定制 Amazon Web Services 服务 的优化。
- `mobile` – 基于标准模式构建，包括为移动应用程序量身定制的优化。
- `auto` – 基于标准模式构建，包括实验功能。SDK 会尝试发现运行时系统环境以自动确定适当的设置。自动检测是基于启发式的，无法提供 100% 的准确性。如果无法确定运行时系统环境，则使用 `standard` 模式。自动检测功能可能会查询实例元数据，这可能会带来延迟。如果启动延迟对您的应用程序而言至关重要，我们建议您改为选择显式 `defaults_mode` 延迟。

在 `config` 文件中设置此值的示例：

```
[default]
defaults_mode = standard
```

以下参数可能会根据 defaults_mode 的选项进行优化：

- retryMode – 指定 SDK 如何尝试重试。请参阅重试行为。
- stsRegionalEndpoints— 指定 SDK 如何确定用于与 Amazon Security Token Service (Amazon STS) 通信的 Amazon Web Services 服务 端点。请参阅Amazon STS 区域终端节点。
- s3UsEast1RegionalEndpoints— 指定软件开发工具包如何确定用于与该us-east-1区域的 Amazon S3 通信的 Amazon 服务终端节点。
- connectTimeoutInMillis – 在套接字上进行初始连接尝试后，超时之前的时长。如果客户端没有收到连接握手完成的消息，则客户端会放弃操作并使其失败。
- tlsNegotiationTimeoutInMillis – 从发送 CLIENT HELLO 消息到客户端和服务器完全协商密码并交换密钥，TLS 握手可能花费的最大时长。

每个设置的默认值会根据为应用程序选择的 defaults_mode 而变化。这些值目前设置如下（可能会发生变化）：

参数	**standard** 模式	**in-region** 模式	**cross-region** 模式	**mobile** 模式
retryMode	standard	standard	standard	standard
stsRegionalEndpoints	regional	regional	regional	regional
s3UsEast1RegionalEndpoints	regional	regional	regional	regional
connectTimeoutInMillis	3100	1100	3100	30000
tlsNegotiationTimeoutInMillis	3100	1100	3100	30000

例如，如果您选择的defaults_mode是standard，则将为retry_mode分配standard的值（来自有效的retry_mode选项），将为stsRegionalEndpoints分配regional的值（来自有效stsRegionalEndpoints选项）。

Support Amazon SDKs by 和工具

以下内容 SDKs 支持本主题中描述的功能和设置。所有部分例外情况均已注明。 适用于 Java 的 Amazon SDK 和 适用于 Kotlin 的 Amazon SDK 唯一支持任何 JVM 系统属性设置。

SDK	支持	备注或更多信息
Amazon CLI v2	否	
适用于 C++ 的 SDK	是	参数未优化：stsRegionalEndpoints 、s3UsEast1RegionalEndpoints 、tlsNegotiationTimeoutInMillis 。
适用于 Go V2 (1.x) 的 SDK	是	参数未优化：retryMode 、stsRegionalEndpoints 、s3UsEast1RegionalEndpoints 。
适用于 Go 1.x（V1）的 SDK	否	
适用于 Java 2.x 的 SDK	是	参数未优化：stsRegionalEndpoints 。
适用于 Java 1.x 的 SDK	否	
适用于 JavaScript 3.x 的软件开发工具包	是	参数未优化：stsRegionalEndpoints 、s3UsEast1RegionalEndpoints 、tlsNegotiationTimeoutInMill

SDK	支持	备注或更多信息
		is 。connectTimeoutInMillis 被称为 connectionTimeout 。
适用于 JavaScript 2.x 的 SDK	否	
适用于 Kotlin 的 SDK	否	
适用于.NET 4.x 的 SDK	是	参数未优化：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。
适用于 .NET 3.x 的 SDK	是	参数未优化：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。
适用于 PHP 3.x 的 SDK	是	参数未优化：tlsNegotiationTimeoutInMillis 。
适用于 Python (Boto3) 的 SDK	是	参数未优化：tlsNegotiationTimeoutInMillis 。
适用于 Ruby 3.x 的 SDK	是	
适用于 Rust 的 SDK	否	
适用于 Swift 的 SDK	否	

SDK	支持	备注或更多信息
适用于 PowerShell V5 的工具	是	参数未优化：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。
适用于 PowerShell V4 的工具	是	参数未优化：connectTimeoutInMillis 、tlsNegotiationTimeoutInMillis 。

Amazon SDK 和工具安全设置

本节介绍所有 Amazon SDK 通用的一些安全设置。但是，要全面了解各个 Amazon SDK 的安全性，请参阅相关 SD Amazon K 开发人员指南中专门的安全章节：

- Amazon 适用于 C++ 的 SDK 安全章节
- Amazon SDK for Go Security 章节
- Amazon 适用于 Java 的 SDK 安全章节
- Amazon JavaScript 安全版 SDK 章节
- Amazon 适用于 Kotlin 的 SDK 安全章节
- Amazon 适用于.NET 的 SDK 安全章节
- Amazon 适用于 PHP 的 SDK 安全章节
- Amazon Python SDK 安全章节
- Amazon 适用于 Ruby 的 SDK 安全章节
- Amazon Rust SDK 安全章节
- Amazon Swift SDK 安全章节

主题

- 启用混合后量子 TLS

启用混合后量子 TLS

Amazon SDK 和工具的加密功能和配置因语言和运行时而异。目前， Amazon SDK 或工具可以通过三种方式提供 PQ TLS 支持：

主题

- 默认情况下启用 PQ TLS 的软件开发工具包
- Opt-in PQ TLS 支持
- 依赖系统 OpenSSL 的软件开发工具包
- Amazon 不打算支持 PQ TLS 的软件开发工具包和工具

默认情况下启用 PQ TLS 的软件开发工具包

> **Note**
>
> 从 6 开始Nov-2025， Amazon 适用于 macOS 和 Windows 的 SDK 及其底层 CRT 库使用系统库来实现 TLS，因此这些平台上的 PQ TLS 功能通常取决于系统级支持。

Amazon 适用于 Go 的 SDK

Amazon 适用于 Go 的 SDK 使用 Golang 自己的标准库提供的 TLS 实现。从 v1.24 开始，Golang 支持并更喜欢 PQ TLS，因此 Amazon SDK for Go 用户只需将 Golang 升级到 v1.24 即可启用 PQ TLS。

Amazon 适用于 JavaScript （浏览器）的 SDK

JavaScript （浏览器）的 Amazon SDK 使用浏览器的 TLS 堆栈，因此，如果浏览器运行时支持和首选 PQ TLS，SDK 将协商 PQ TLS。Firefox 在 v132.0 中推出了对 PQ TLS 的支持。Chrome 宣布在 v131 中支持 PQ TLS。Edge 支持台式机版 v120 中的可选加入 PQ TLS，安卓版 140 版支持选择加入 PQ

Amazon 适用于 Node.js

从 Node.js v22.20 (LTS) 和 v24.9.0 开始，静态 Node.js 链接和捆绑了 OpenSSL 3.5。这意味着 PQ TLS 在这些版本和后续版本中默认处于启用状态，并且首选。

Amazon 适用于 Kotlin 的 SDK

从 1.5.78 版本开始，Kotlin SDK 支持并更喜欢在 Linux 上使用 PQ TLS。由于 Amazon 适用于 Kotlin CRT-based 客户端的 SDK 依赖于 macOS 和 Windows 上的 TLS 系统库，因此对 PQ TLS 的支持将取决于这些底层系统库。

Amazon 适用于 Rust 的 SDK

适用于 Rust 的 Amazon SDK 为每个服务客户端分发不同的软件包（在 Rust 生态系统中被称为 “板条箱”）。它们都在统一的 GitHub 存储库中进行管理，但每个服务客户端都遵循自己的版本和发布节奏。合并后的 SDK 于 8/29 /25 发布了 PQ TLS 首选项，因此在该日期之后发布的任何单个服务客户端版本都将默认支持并首选 PQ TLS。

您可以通过导航到相关的 crates.io 版本网址（例如，这里是）并查找 29-之后发布的第一个版本来确定支持特定服务客户端 Amazon Web Services 服务抵扣金的 PQ TLS 的最低版本。Aug-25默认情况下，在 29-之后发布的任何服务客户端版本都Aug-25将启用 PQ TLS 并处于首选状态。

Opt-in PQ TLS 支持

Amazon SDK for C++

默认情况下，C++ 开发工具包使用平台原生客户端，例如 libcurl 和。 WinHttpLibcurl 通常依赖系统 OpenSSL 来实现 TLS，因此只有在系统 OpenSSL 等于 v3.5 时，才会默认启用 PQ TLS。你可以在 C++ SDK v1.11.673 或更高版本中覆盖此默认值，然后选择加入默认支持和启用 PQ TLS AwsCrtHttpClient 的。

关于构建 Opt-In PQ TLS 的注意事项您可以使用此脚本获取 SDK 的 CRT 依赖项。这里和这里都描述了从源代码构建 SDK，但请注意，你可能需要一些额外的 CMake 标志：

```
-DUSE_CRT_HTTP_CLIENT=ON \
-DUSE_TLS_V1_2=OFF \
-DUSE_TLS_V1_3=ON \
-DUSE_OPENSSL=OFF \
```

Amazon 适用于 Java 的 SDK

从 v2 开始， Amazon 适用于 Java 的 SDK 提供了一个 Amazon 公共运行时 (Amazon CRT) HTTP 客户端，可以将其配置为执行 PQ TLS。从 v2.35.11 开始，无论在哪里使用 PQ TLS，都会默认 AwsCrtHttpClient 启用并首选 PQ TLS。

依赖系统 OpenSSL 的软件开发工具包

有几个 Amazon SDK 和工具依赖于系统的 TLS libcrypto/libssl 库。最常用的系统库是 OpenSSL。在 3.5 版本中启用了 OpenSSL 的 PQ TLS 支持，因此为 PQ TLS 配置这些软件开发工具包和工具的最简单方法是在至少安装了 OpenSSL 3.5 的操作系统发行版上使用它。

你也可以将 Docker 容器配置为使用 OpenSSL 3.5 在任何支持 Docker 的系统上启用 PQ TLS。有关为 Python 进行此设置的示例，请参阅 Python 中的 T Post-quantum LS。

Amazon CLI

从 v2.34.54 开始，适用于 Linux 的 CL Amazon I 安装程序捆绑了 OpenSSL 3.5.6，因此，在 Linux 上，该版本和后续版本默认启用并首选 PQ TLS。 Amazon Linux 上的 CLI 用户可以通过升级到 CL Amazon I v2.34.54 或更高版本来启用 PQ TLS。

对于 macOS，请通过 Homebrew 安装 Amazon CLI，并确保你的 Op Homebrew-vended enSSL 已升级到 3.5+ 版本。你可以用 “brew install openssl @3 .6” 来做到这一点，然后用 “brew list | grep openssl” 进行验证。

有关验证安装的分步说明，请参阅 github 存储库和随附的博客文章。

Amazon 适用于 PHP 的 SDK

Amazon 适用于 PHP 的 SDK 依赖于系统 libssl/libcrypto。要使用 PQ TLS，请在至少安装了 OpenSSL 3.5 的操作系统发行版上使用此 SDK。

Amazon 适用于 Python 的 SDK（Boto3）

Amazon 适用于 Python 的 SDK (Boto3) 依赖于 TLS 的 Python 安装所关联的 OpenSSL 库。行为因平台而异：

Windows and macOS (python.org installer)

适用于 Windows (.exe) 和 macOS (.pkg) 的官方 python.org 安装程序将他们自己的 OpenSSL 库捆绑在一起。从 Python 3.14.6 开始，捆绑的 OpenSSL 3.5.7 默认支持并首选 PQ TLS。无需进行其他配置。

macOS (Homebrew)

由于 Homebrew 的 p ython @3 .14 链接到 Homebrew 的 openssl @3 共享库，因此从 Homebrew OpenSSL 3.5+ 开始，默认情况下支持和首选 PQ TLS。你可以通过以下方式验证你的 OpenSSL 版本：

```
python3.14 -c "import ssl; print(ssl.OPENSSL_VERSION)"
```

如果你的版本低于 3.5，请使用 brew 升级 openssl @3 进行升级。

Linux

在 Linux 上，Python 会动态链接到系统的共享 libssl。安装 OpenSSL 3.5+ 是必要的，但还不够，因为系统的加密策略还必须在其默认 TLS 配置中包括后量子组。

对于亚马逊 Linux 2023（AL2023.12 或更高版本），请使用以下命令启用 PQ TLS：

```
sudo update-crypto-policies --set DEFAULT:PQ
```

有关更多信息，请参阅在 AL2023 上启用 Post-Quantum 加密 (PQC)。AL2023

对于其他 Linux 发行版，请参阅您的发行版中有关配置默认 TLS 组的文档。

您可以通过检查 TLS 握手中的 X25519MLKEM768 密钥交换来验证 PQ TLS 是否正常工作。

Amazon 适用于 Ruby 的 SDK

Amazon 适用于 Ruby 的 SDK 依赖于系统 libssl/libcrypto。要使用 PQ TLS，请在至少安装了 OpenSSL 3.5 的操作系统发行版上使用此 SDK。

Amazon 适用于 .NET 的 SDK

在 Linux 上，适用于.NET 的 Amazon SDK 依赖于系统 libssl/libcrypto。要使用 PQ TLS，请在至少安装了 OpenSSL 3.5 的操作系统发行版上使用此 SDK。在 Windows 和 macOS 上，PQ TLS 从 .NET 10 和 Windows 11 开始可用。在 macOS 上，可以通过选择加入 Apple 来启用 TLS 1.3 支持（PQ TLS 的先决条件），如下所述。 Network.framework 假设.NET 版本最低为 10，则应启用 PQ TLS。

Amazon 不打算支持 PQ TLS 的软件开发工具包和工具

目前没有计划支持以下语言 SDK 和工具：

- Amazon SAP 软件开发工具包
- Amazon 适用于 Swift
- Amazon 适用于 Windows 的工具 PowerShell

Amazon 通用运行时 (CRT) 库

Amazon 通用运行时 (CRT) 库是 SDK 的基础库。CRT 是一个由独立程序包组成的模块化系列，用 C 语言编写。每个程序包都为不同的所需功能提供了良好的性能和最小的占用空间。这些功能在所有 SDK 中都是通用和共享的，可提供更好的代码重用、优化和准确性。程序包是：

- awslabs/aws-c-auth: Amazon 客户端身份验证（标准凭据提供程序和签名 (sigv4)）
- awslabs/aws-c-cal：加密原始类型、哈希（MD5、SHA256、SHA256 HMAC）、签名者、AES
- awslabs/aws-c-common: 基本数据结构、 threading/synchronization 原始类型、缓冲区管理、stdlib 相关函数
- awslabs/aws-c-compression: 压缩算法（Huffman encoding/decoding）
- awslabs/aws-c-event-stream: 事件流消息处理（标头、前奏、有效负载 crc/trailer）、通过事件流实现远程过程调用 (RPC)
- awslabs/aws-c-http: C99 实施 HTTP/1.1 和规范 HTTP/2
- awslabs/aws-c-io: 套接字（TCP、UDP）、DNS、管道、事件循环、通道、 SSL/TLS
- awslabs/aws-c-iot: C99 实现 Amazon 物联网云服务与设备集成
- awslabs/aws-c-mqtt：适用于物联网 (IoT) 的标准轻量级消息传输协议
- awslabs/aws-c-s3：C99 库实现与 Amazon S3 服务的通信，旨在最大限度地增加高带宽 Amazon EC2 实例的吞吐量
- awslabs/aws-c-sdkutils: 用于解析和管理 Amazon 配置文件的实用程序库
- awslabs/aws-checksums: Cross-platform 硬件加速的 crc32c 和 CRC32，可回退到高效的软件实现
- awslabs/aws-lc: General-purpose Amazon 密码学团队根据谷歌 BoringSSL Amazon 项目和 OpenSSL 项目的代码为其客户维护的密码库
- awslabs/s2n: C99 TLS/SSL 协议的实施，这些协议的设计既小又快，将安全作为优先事项

CRT 可通过除 Go 和 Rust 之外的所有 SDK 获得。

CRT 依赖关系

CRT 库构成了一个由关系和依赖关系组成的复杂网络。如果您需要直接从源代码构建 CRT，了解这些关系会很有帮助。但是，大多数用户通过其语言软件开发工具包（例如适用于 C++ 的 SDK 或 Java 的 Amazon SDK）或他们的语言物联网设备 Amazon 软件开发工具包（例如适用于 C++ 的 IoT SDK 或适

用于 Java 的 Amazon lo Amazon T SDK) 来访问 CRT 功能。在下图中，“语言 CRT 绑定” 框指的是封装特定语言 SDK 的 CRT 库的程序包。这是格式为 `aws-crt-*` 的程序包的集合，其中“*”是 SDK 语言（例如 `aws-crt-cpp` 或 `aws-crt-java`）。

下图概述了 CRT 库的分层依赖关系。
CRT 依赖关系图显示了各个 CRT 库是如何相互关联的。

Amazon SDK 和工具维护政策

概述

本文档概述了 Amazon 软件开发套件 (SDK) 和工具（包括移动和物联网软件开发工具包）的维护政策及其底层依赖关系。 Amazon 定期向 Amazon SDK 和工具提供更新，其中可能包含对新增或更新 Amazon 的 API、新功能、增强功能、错误修复、安全补丁或文档更新的支持。更新还可以解决依赖关系、语言运行时和操作系统的变化。 Amazon SDK 版本发布给包管理器（例如 Maven NuGet、、PyPI），并作为源代码提供。 GitHub

我们建议用户及时了解 SDK 版本，以了解其最新功能、安全更新和底层依赖项。不建议继续使用不受支持的 SDK 版本，但是否继续使用由用户自行决定。

版本控制

S Amazon DK 发布版本的形式是， X.Y.Z 其中 X 代表主要版本。增加 SDK 的主版本表明该 SDK 进行了重大而实质性的更改，以支持该语言中的新习语和模式。当公共接口（例如类、方法、类型等）、行为或语义发生变化时，就会引入主要版本。应用程序需要更新才能使用最新的 SDK 版本。请务必根据 Amazon提供的升级指南谨慎更新主要版本。

SDK 主要版本的生命周期

主要 SDK 和工具版本的生命周期由 5 个阶段组成，概述如下。

- 开发者预览版（第 0 阶段）- 在此阶段，不支持 SDK，不应在生产环境中使用，并且仅用于抢先体验和反馈目的。未来版本可能会引入重大变更。一旦 Amazon 确定某个版本为稳定产品，它就可以将其标记为候选版本。除非出现重大错误，否则候选版本已准备好发布，并且将获得全力 Amazon 支持。
- 正式发布 (GA)（第 1 阶段）- 在此阶段，完全支持 SDK。 Amazon 将提供常规的 SDK 版本，其中包括对新服务的支持、现有服务的 API 更新以及错误和安全修复。对于工具， Amazon 将提供包含新功能更新和错误修复的常规版本。 Amazon 将支持 GA 版本的 SDK 至少 24 个月。
- 维护公告（第 2 阶段）- Amazon 将在 SDK 进入维护模式前至少 6 个月发布公告。在此期间，SDK 将继续得到全面支持。通常，维护模式是在下一个主要版本过渡到 GA 的同时宣布的。
- 维护（第 3 阶段）- 在维护模式期间， Amazon 将 SDK 版本限制为仅解决关键错误修复和安全问题。SDK 不会收到新服务或现有服务的 API 更新，也不会更新以支持新区域。除非另有说明，否则维护模式的默认持续时间为 12 个月。

- End-of-Support（第 4 阶段）- 当 SDK 达到终止支持时，它将不再接收更新或版本。之前发布的版本将继续通过公共包管理器提供，并且代码将保持不变 GitHub。 GitHub 存储库可能已存档。用户可自行决定是否使用已终止支持的 SDK。我们建议用户升级到新的主要版本。

下图直观地说明了 SDK 主要版本的生命周期。请注意，下面显示的时间表仅供参考，不具约束力。
维护政策时间表

依赖项生命周期

大多数 Amazon SDK 都有底层依赖关系，例如语言运行时、操作系统或第三方库和框架。这些依赖项通常与语言社区或拥有该特定组件的供应商有关。每个社区或供应商都会发布自己的产品终止支持时间表。

以下术语用于对底层第三方依赖项进行分类：

- 操作系统 (OS)：示例包括 Amazon Linux AMI、Amazon Linux 2、Windows 2008、Windows 2012、Windows 2016 等。
- 语言运行时系统：示例包括 Java 7、Java 8、Java 11、.NET Core、.NET Standard、.NET PCL 等。
- 第三方库/框架：示例包括 OpenSSL、.NET Framework 4.5、Java EE 等。

我们的政策是在社区或供应商终止对 SDK 依赖项的支持后至少 6 个月内继续支持 SDK 依赖项。但是，此策略可能会因具体的依赖项而有所不同。

> **Note**
>
> Amazon 保留在不增加主要 SDK 版本的情况下停止对底层依赖项的支持的权利

沟通方式

维护公告将通过多种方式传达：

- 我们会向受影响的账户发送一封电子邮件公告，宣布我们计划终止对特定 SDK 版本的支持。该电子邮件将概述终止支持的路径，指定活动时间表，并提供升级指导。
- Amazon SDK 文档（例如 API 参考文档、用户指南、SDK 产品营销页面和 GitHub 自述文件）已更新，以指明活动时间表并提供有关升级受影响应用程序的指导。

- 发布了一篇 Amazon 博客文章，概述了终止支持的路径，并重申了活动时间表。
- 在 SDK 中添加了弃用警告，概述了终止支持的路径并链接到 SDK 文档。

要查看软件开发工具包和 Amazon 工具的可用主要版本列表以及它们在维护生命周期中所处的位置，请参阅版本生命周期。

Amazon SDKs 和 “工具” 版本生命周期

下表显示了可用的 Amazon 软件开发套件 (SDK) 主要版本列表，以及它们在维护生命周期中所处的位置以及相关的时间表。有关主要版本 Amazon SDKs 和 Tools 的生命周期及其底层依赖项的详细信息，请参阅维护政策。

SDK	主要版本	当前阶段	公开发行日期	注意
Amazon CLI	1.x	维护公告	9/2/2013	详情和日期请查看公告
Amazon CLI	2.x	公开发行	2020 年 10 月 2 日	
适用于 C++ 的 SDK	1.x	公开发行	9/2/2015	
适用于 Go V2 的 SDK	V2 1.x	公开发行	1/19/2021	
适用于 Go 的 SDK	1.x	停止支持	11/19/2015	
适用于 Java 的 SDK	1.x	停止支持	2010 年 3 月 25 日	
适用于 Java 的 SDK	2.x	公开发行	11/20/2018	
适用于 JavaScript	1.x	停止支持	5/6/2013	
适用于 JavaScript	2.x	停止支持	6/19/2014	
适用于 JavaScript	3.x	公开发行	12/15/2020	

SDK	主要版本	当前阶段	公开发行日期	注意
适用于 Kotlin 的 SDK	1.x	公开发行	11/27/2023	
适用于 .NET 的 SDK	1.x	停止支持	11/2009	
适用于 .NET 的 SDK	2.x	停止支持	11/8/2013	
适用于 .NET 的 SDK	3.x	公开发行	7/28/2015	
适用于 .NET 的 SDK	4.x	公开发行	2025 年 4 月 28 日	
适用于 PHP 的 SDK	2.x	停止支持	11/2/2012	
适用于 PHP 的 SDK	3.x	公开发行	5/27/2015	
适用于 Python (Boto2) 的 SDK	1.x	停止支持	7/13/2011	
适用于 Python (Boto3) 的 SDK	1.x	公开发行	2015 年 6 月 22 日	
适用于 Python (Botocore) 的 SDK	1.x	公开发行	2015 年 6 月 22 日	
适用于 Ruby 的 SDK	1.x	停止支持	7/14/2011	
适用于 Ruby 的 SDK	2.x	停止支持	2015 年 2 月 15 日	

SDK	主要版本	当前阶段	公开发行日期	注意
适用于 Ruby 的 SDK	3.x	公开发行	8/29/2017	
适用于 Rust 的 SDK	1.x	公开发行	11/27/2023	
适用于 Swift 的 SDK	1.x	公开发行	9/17/2024	
用于 PowerShell	2.x	停止支持	11/8/2013	
用于 PowerShell	3.x	停止支持	7/29/2015	
用于 PowerShell	4.x	公开发行	11/21/2019	
用于 PowerShell	5.x	公开发行	2025 年 6 月 23 日	

正在寻找未提及的 SDK 或工具？ 例如 SDKs，加密 SDKs、物联网设备和移动 SDKs设备不包含在本指南中。要查找有关其他工具的文档，请参阅 Tools to Build on Amazon。

Amazon SDKs 和《工具参考指南》的文档历史记录

下表介绍了《和工具参考指南》的重要新增内容Amazon SDKs 和更新。如需有关此文档的更新通知，您可以订阅 RSS 源。

变更	说明	日期
增加新 S3 Express One Zone 设置	增加有关新 S3 Express One Zone 设置的内容，该设置用来禁用会话身份验证。	2025 年 10 月 13 日
增加新的身份验证决策树	增加有关新决策树功能的内容，该功能用来帮助在选项之间做出身份验证决策。	2025 年 9 月 23 日
增加新身份验证方案功能	增加有关新身份验证方案功能的内容。 Amazon STS 区域终端节点的更新。	2025 年 8 月 18 日
添加新版本的 “工具” PowerShell	添加最新版本的 Tools 以 PowerShell 支持所有设置参考与 Amazon SDKs 表格的兼容性。增加了有关“主机前缀注入功能”的内容。	2025 年 6 月 23 日
页面标题更新	其他标题、表格标题、摘要和 SEO 更新。	2025 年 3 月 5 日
页面标题更新	更新了内容，以使用更具描述性的标题。	2025 年 2 月 24 日
在“设置参考”中增加了 Swift SDK	将 Swift SDK 支持添加到所有设置参考与 Amazon SDKs 表格的兼容性。	2024 年 9 月 17 日
适用于 Java 的 SDK 1.x 系统属性	添加有关 适用于 Java 的 Amazon SDK 1.x 支持的 JVM 系统配置设置的详细信息。	2024 年 5 月 30 日

设置更新	增加了 JVM 系统配置设置。	2024 年 3 月 27 日
兼容性表更新	更新了有关 SDK 支持兼容性的内容，更新了有关 IAM Identity Center 操作过程的内容。	2024 年 2 月 20 日
容器凭证更新。IMDS 更新。	正在添加对 Amazon EKS 的支持。正在添加设置以禁用 IMDSv1 回退。	2023 年 12 月 29 日
请求压缩	正在为请求压缩特征添加设置。	2023 年 12 月 27 日
兼容性表	SDK 和工具特征的兼容性表已更新，包括 SDK for Kotlin、SDK for Rust 和 Amazon Tools for PowerShell。	2023 年 12 月 10 日
身份验证更新	更新了 SDKs 和工具支持的身份验证方法。	2023 年 7 月 1 日
IAM 最佳实践更新	更新了指南，使其符合 IAM 最佳实践。有关更多信息，请参阅 IAM 安全最佳实践。	2023 年 2 月 27 日
SSO 更新	更新了新 SSO 令牌配置的 SSO 凭证。	2022 年 11 月 19 日
设置更新	更新了常规配置和 Amazon S3 多区域接入点的支持表。	2022 年 11 月 17 日
设置更新	更新了 IMDS 客户端和 IMDS 凭证的明确程度。更新了环境变量。	2022 年 11 月 4 日
更新欢迎页面	宣布亚马逊 CodeWhisperer。	2022 年 9 月 22 日

更改单点登录的服务名称	为了反映 Amazon SSO 而进行的更新现在被称为。 Amazon IAM Identity Center	2022 年 7 月 26 日
设置更新	小幅更新配置文件详细信息和支持的设置。	2022 年 6 月 15 日
更新	大幅更新了本指南几乎所有部分。	2022 年 2 月 1 日
初始版本	本指南的第一版已向公众发布。	2020 年 3 月 13 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。