Application Auto Scaling

用户指南

Application Auto Scaling: 用户指南

Table of Contents

什么是 Application Auto Scaling？

Application Auto Scaling 是一项 Web 服务，可为开发人员和系统管理员提供一个解决方案，解决为超出 Amazon EC2 的各 AWS 服务自动扩展其可扩展资源的问题。Application Auto Scaling 允许您配置以下资源的自动扩展：

- Amazon ECS 服务
- Spot 队列请求
- Amazon EMR 集群
- AppStream 2.0 队列
- DynamoDB 表和全局二级索引
- Aurora 副本
- Amazon SageMaker 终端节点变体
- 由您自己的应用程序或服务提供的自定义资源。有关更多信息，请参阅 GitHub 存储库。

您有多种方式可用来使用 AWS 进行扩展。有关扩展 Amazon EC2 实例队列的信息，请参阅 Amazon EC2 Auto Scaling 用户指南。

您还可以使用 Application Auto Scaling 和 Amazon EC2 Auto Scaling 并结合使用 AWS Auto Scaling 来跨多个服务扩展资源。AWS Auto Scaling 可以通过结合预测扩展和动态扩展（分别为主动和被动方法）来更快地扩展您的 Amazon EC2 容量，以帮助您维护最佳可用性和性能。有关更多信息，请参阅 AWS Auto Scaling 用户指南。

Application Auto Scaling 的功能

Application Auto Scaling 可以让您根据您定义的条件自动扩展可扩展资源。

- 目标跟踪扩展 — 根据特定 CloudWatch 指标的目标值扩展资源。
- 步进扩展 — 根据一组扩展调整扩展资源，这些调整因警报违例大小而异。
- 计划扩展 — 根据日期和时间扩展资源。

开始使用

Application Auto Scaling 集成以下所有服务，以便您可以直接从想要扩展的资源的控制台调用 API 操作。如果您是首次接触 Application Auto Scaling 的用户，我们建议您参阅以下您感兴趣的服务的文档，以了解如何将它们与 Application Auto Scaling 集成。这些主题包含对主要使用 AWS 管理控制台来与 Application Auto Scaling 交互的用户而言尤其有帮助的信息。

- Amazon Elastic Container Service Developer Guide 中的服务自动扩展
- Amazon EC2 用户指南 中的 Spot 队组自动扩展
- Amazon EMR 管理指南 中的在 Amazon EMR 中使用自动扩展
- Amazon AppStream 2.0 开发人员指南 中的 AppStream 2.0 的队组自动扩展
- Amazon DynamoDB 开发人员指南 中的使用 DynamoDB 管理吞吐容量
- Amazon RDS 用户指南 中的将 Amazon Aurora Auto Scaling 用于 Aurora 副本
- Amazon SageMaker 开发人员指南 中的自动扩展 Amazon SageMaker 模型

要查看上面列出的任一 AWS 服务的区域可用性，请参阅区域表。

如果您希望使用命令行选项来指定 Application Auto Scaling API 操作，也可以选择使用此用户指南。要开始使用，请完成AWS CLI 使用入门 (p. 4)中的练习。在本教程中，我们向您介绍如何使用 AWS Command Line Interface (AWS CLI) 以编程方式访问 Application Auto Scaling。但是，如果在任何时候，您需要此用户指南中未包含的信息（包括您可以尝试的示例扩展策略），请参阅上述的服务文档。

访问 Application Auto Scaling

如果您已注册 AWS 账户，请通过登录 AWS 管理控制台访问 Application Auto Scaling。然后，打开服务控制台以查看“入门”一节中列出的服务之一。

您也可以使用 Application Auto Scaling API 访问 Application Auto Scaling。Application Auto Scaling 提供了查询 API。这些请求属于 HTTP 或 HTTPS 请求，需要使用 HTTP 动词 GET 或 POST 以及一个名为 `Action` 的查询参数。有关更多信息，请参阅 Application Auto Scaling API 参考 中的操作。

如果您倾向于使用特定语言的 API 而非通过 HTTP 或 HTTPS 提交请求来构建应用程序，AWS 为软件开发人员提供了库文件、示例代码、教程和其他资源。这些库文件提供可自动执行任务的基本功能，例如以加密方式对请求签名、重试请求和处理错误响应，因此您可以更轻松地上手。有关更多信息，请参阅 AWS SDKs and Tools。

如果倾向于使用命令行界面，您可使用以下选项：

AWS Command Line Interface (AWS CLI)

提供大量 AWS 产品的相关命令，同时被 Windows、macOS 和 Linux 支持。要了解其用法，请参阅 AWS Command Line Interface 用户指南。有关更多信息，请参阅 AWS CLI Command Reference 中的 application-autoscaling。

适用于 Windows PowerShell 的 AWS 工具

为在 PowerShell 环境中编写脚本的用户提供大量 AWS 产品的相关命令。要开始使用，请参阅 适用于 Windows PowerShell 的 AWS 工具 用户指南。有关更多信息，请参见 适用于 PowerShell 的 AWS 工具 Cmdlet Reference。

有关用于访问 AWS 的证书的信息，请参阅 Amazon Web Services 一般参考 中的 AWS 安全凭证。有关 Application Auto Scaling 的区域和终端节点的信息，请参阅 AWS General Reference 中的 AWS 区域和终端节点。

设置

在使用 Application Auto Scaling 配置自动扩展之前，请创建 AWS 账户、配置访问权限和设置 AWS Command Line Interface (AWS CLI)。

主题

注册 AWS

当您注册 AWS 时，您的 AWS 账户会自动注册 AWS 中的所有服务，包括 Application Auto Scaling。您只需为使用的服务付费。

如果您还没有 AWS 账户，则需要创建一个账户。如果您已有 AWS 账户，则可以跳过以下过程中创建账户的步骤，而转至步骤 3 来创建 IAM 用户。

注册 AWS

1. 打开 http://www.amazonaws.cn/，然后选择 Sign Up (注册)。
2. 按照屏幕上的说明进行操作。在注册过程中，您将接到一通电话，要求您使用电话键盘输入一个验证码。注册过程完成后，AWS 会向您发送一封确认电子邮件。
3. 创建 AWS Identity and Access Management (IAM) 管理员用户。有关说明，请参阅 IAM 用户指南 中的创建您的第一个 IAM 用户和组。

   Important

   本指南中的入门练习假定您拥有具有管理员权限的用户 (`adminuser`)。请按照以下过程在您的账户中创建 `adminuser`。
4. 确保您具有与您刚创建的 IAM 用户关联的访问密钥 ID 和秘密访问密钥。有关更多信息，请参阅 AWS Command Line Interface 用户指南 中的访问密钥和秘密访问密钥。

有关 IAM 的更多信息，请参阅下文：

- AWS Identity and Access Management (IAM)
- 入门
- IAM 用户指南

在 AWS 区域中使用 Application Auto Scaling

Application Auto Scaling 在多个 AWS 区域中提供。有关提供的区域的列表，请参阅 AWS General Reference 中的 AWS 区域和终端节点。利用全球 AWS 账户，您可以在大多数区域中使用资源。当利用中国区域的资源使用 Application Auto Scaling 时，请记住您必须拥有单独的 AWS（中国）账户。此外，Application Auto Scaling 的实现方式上存在一些差异。有关在中国区域中使用 Application Auto Scaling 的更多信息，请参阅中国的 Application Auto Scaling。

设置 AWS CLI

AWS Command Line Interface (AWS CLI) 是一款用于管理 AWS 服务的统一开发人员工具，包括 Application Auto Scaling。按照以下步骤下载和配置 AWS CLI。

设置 AWS CLI

1. 下载并配置 AWS CLI。有关说明，请参阅 AWS Command Line Interface 用户指南 中的以下主题：
   - 安装 AWS CLI
   - 配置 AWS CLI
2. 运行以下命令确认是否安装了 AWS CLI 的 Application Auto Scaling 命令。

```
aws application-autoscaling help
```

3. 在 AWS CLI 配置文件中为管理员用户添加一个命名配置文件。在执行 AWS CLI 命令时，您可以使用此配置文件。有关命名配置文件的更多信息，请参阅 AWS Command Line Interface 用户指南 中的命名配置文件。

```
aws configure --profile adminuser
```

   出现提示时，指定用于 Application Auto Scaling 的 IAM 用户的 AWS 访问密钥和 AWS 私有访问密钥。

```
aws_access_key_id = adminuser access key ID
aws_secret_access_key = adminuser secret access key
region = aws-region
default output format = json
```

   有关可用 AWS 区域的列表，请参阅 Amazon Web Services 一般参考 中的区域和终端节点。
4. 为了确认 AWS CLI 配置文件的配置正确无误，请在命令窗口中运行以下命令：

```
aws configure --profile adminuser
```

   如果您的配置文件已正确配置，您应该看到类似于以下内容的输出。

```
AWS Access Key ID [****************52FQ]:
AWS Secret Access Key [****************xgyZ]:
Default region name [us-east-1]:
Default output format [json]:
```

在设置 AWS 账户和 AWS CLI 后，您可以尝试下一个教程，在其中配置示例计划扩展操作。

AWS CLI 使用入门

在本教程中，您将使用 AWS CLI 探索 Application Auto Scaling。开始之前，请确保您已经有 AWS 账户并且已设置 AWS CLI。有关更多信息，请参阅 设置 (p. 3)。在本教程中，您将创建计划操作，以基于计划扩展可扩展的资源。通过计划的扩展，您可以指定一次性操作或重复性操作。

本教程中的练习假定您使用在设置 AWS CLI (p. 4)中设置的管理员凭证(`adminuser` 配置文件) 。如果您未提供此配置文件，系统会使用默认配置文件。请注意，如果要创建、更新、删除或列出 Application Auto Scaling 资源，您需要有权执行该操作，并且您需要有权访问相应的资源。有关更多信息，请参阅 Application Auto Scaling 的身份验证和访问控制 (p. 27)。

本教程中的 CLI 命令在 Linux 上进行了测试。要在 Microsoft Windows 中使用这些示例，请将换行符从反斜杠 (\) 改为插入符号 (^)。有关在 Windows 上使用 CLI 命令的信息，请参阅 AWS Command Line Interface 用户指南 中的为 AWS 命令行界面指定参数值。

> Note
>
> 使用本教程的过程中，您可能会产生 AWS 费用。请监控您的免费套餐使用情况并确保您了解所涉及的 AWS 费用。

目录

步骤 1：注册您的可扩展目标

首先，通过 Application Auto Scaling 将您的资源注册为可扩展的目标。可扩展目标是 Application Auto Scaling 可以扩大或缩小的资源。

您可以使用任何能处理 Application Auto Scaling 的资源 ，但对于这些示例，我们假设您要扩展名为 `my-table` 的 DynamoDB 表。如果您还没有 DynamoDB 表，您可以现在创建一个（Amazon DynamoDB 开发人员指南 中的步骤 1：创建 DynamoDB 表）。

要将 DynamoDB 全局二级索引或资源用于不同的服务，请相应地更新示例。在 `--service-namespace` 中指定其命名空间，在 `--scalable-dimension` 中指定其扩展维度，并在 `--resource-id` 中指定其资源 ID。有关每个选项的有效值的列表，请参阅 register-scalable-target。

向 Application Auto Scaling 注册您的可扩展目标

1. （可选）使用 describe-scalable-targets 命令来检查是否已注册了任何 DynamoDB 资源。这可帮助您验证是否注册 `my-table` 表。例如，如果您之前从 DynamoDB 控制台为此表配置了自动扩展功能，此表可能已向 Application Auto Scaling 注册了。

   ```
   aws application-autoscaling describe-scalable-targets \
     --service-namespace dynamodb \
     --profile adminuser
   ```

   如果没有现有的可扩展目标，则这是系统的响应。

   ```
   {
       "ScalableTargets": []
   }
   ```

2. 使用以下 register-scalable-target 命令来注册或更新名为 `my-table` 的 DynamoDB 表的写入容量。将最小所需容量设置为 5 个写入容量单位，将最大所需容量设置为 10 个写入容量单位。

   ```
   aws application-autoscaling register-scalable-target \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/my-table \
     --min-capacity 5 --max-capacity 10 \
     --profile adminuser
   ```

   如果此命令成功执行，将不会返回任何输出。

步骤 2：创建两个计划操作

Application Auto Scaling 可让您安排扩展操作应发生的时间。您可以指定可扩展目标、扩展计划、最小容量和最大容量。在指定的时间，Application Auto Scaling 会更新可扩展目标的最小值和最大值。如果当前容量超出此范围，这会导致一个扩展活动。

如果您决定创建扩展策略，计划更新最小和最大容量也会有所帮助。扩展策略允许基于当前资源利用率来动态扩展您的资源。扩展策略的一种常见保护措施是设置适当的最小和最大容量值。

在本练习中，我们将创建两个一次性操作来分别进行扩展和缩减。

创建和查看计划操作

1. 要创建第一个计划操作，请使用以下 put-scheduled-action 命令。

   `--schedule` 中的 at 命令安排操作在将来的指定日期和时间执行一次。小时采用世界标准时间 24 小时格式。将操作安排在当前时间开始约 5 分钟后发生。

   在指定的日期和时间，Application Auto Scaling 将更新 `MinCapacity` 和 `MaxCapacity` 的值。假设表当前有 5 个写入容量单位，Application Auto Scaling 扩展到 `MinCapacity`，以使表拥有 15-20 个写入容量单位的新所需范围。

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/my-table \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "at(2019-05-20T17:05:00)" \
     --scalable-target-action MinCapacity=15,MaxCapacity=20 \
     --profile adminuser
   ```

   如果此命令成功执行，将不会返回任何输出。

2. 要创建第二个计划操作供 Application Auto Scaling 来进行缩减，请使用以下 put-scheduled-action 命令。

   将操作安排在当前时间开始约 10 分钟后发生。

   在指定的日期和时间，Application Auto Scaling 将更新 `MinCapacity` 和 `MaxCapacity` 的值，并且缩减到 `MaxCapacity` 以将表恢复到初始 5 到 10 个写入容量单位的所需范围。

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/my-table \
     --scheduled-action-name my-second-scheduled-action \
     --schedule "at(2019-05-20T17:10:00)" \
     --scalable-target-action MinCapacity=5,MaxCapacity=10 \
     --profile adminuser
   ```

3. （可选）您可以使用以下 describe-scheduled-actions 命令获得指定服务命名空间的计划操作列表。

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace dynamodb \
     --profile adminuser
   ```

   下面是示例输出。

   ```
   {
       "ScheduledActions": [
   ```

```
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-first-scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
            "ScheduledActionName": "my-first-scheduled-action",
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-second-scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

步骤 3：查看扩展活动

在此步骤中，您将查看计划操作触发的扩展活动，并验证 DynamoDB 是否已更改表的写入容量。

查看扩展活动

1. 等到您选择的时间，使用以下 describe-scaling-activities 命令确认计划操作在正常运行。

```
aws application-autoscaling describe-scaling-activities \
  --service-namespace dynamodb \
  --profile adminuser
```

以下是第一个计划操作在计划操作正在执行时的示例输出。

扩展活动按创建日期排序，首先返回最新的扩展活动。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Waiting for
 change to be fulfilled by dynamodb.",
```

```
            "StatusCode": "InProgress"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
 20",
            "StatusCode": "Successful"
        }
    ]
}
```

以下是两个计划操作都运行完成后的示例输出。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
```

```
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
 20",
            "StatusCode": "Successful"
        }
    ]
}
```

2. 成功运行计划操作后，请转至 DynamoDB 控制台并选择要处理的表。在容量选项卡下查看写入容量单位。在第二个扩展操作运行后，写入容量单位应已从 15 变为 10。

   您还可以通过 AWS CLI 查看此信息。

   请使用 DynamoDB describe-table 命令验证表的当前写入容量。

```
aws dynamodb describe-table --table-name my-table \
  --query "Table.[TableName,TableStatus,ProvisionedThroughput]" \
  --profile adminuser
```

   下面是示例输出。

```
[
    "my-table",
    "ACTIVE",
    {
        "NumberOfDecreasesToday": 1,
        "WriteCapacityUnits": 10,
        "LastIncreaseDateTime": 1561574133.264,
        "ReadCapacityUnits": 5,
        "LastDecreaseDateTime": 1561574435.607
    }
]
```

步骤 4：后续步骤

现在您已经熟悉了 Application Auto Scaling 及其部分功能，请考虑执行以下操作：

- 如果您想尝试通过重复性计划进行扩展，请参阅Application Auto Scaling 的计划扩展 (p. 21)中的示例。
- 如果您要尝试根据资源利用率的变化进行动态扩展（例如：使用 `DynamoDBWriteCapacityUtilization` 指标），请按照 Application Auto Scaling 的目标跟踪扩展策略 (p. 11) 中的步骤操作。

第 5 步：清除

当您完成入门练习后，可以按照如下步骤清除关联的资源。

删除计划的操作

下面的 delete-scheduled-action 命令将删除指定的计划操作。如果您要将此计划操作保留供将来使用，您可以跳过此操作。

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/my-table \
```

```
  --scheduled-action-name my-second-scheduled-action \
  --profile adminuser
```

撤消可扩展目标的注册

使用如下 deregister-scalable-target 命令来取消注册可扩展目标。如果您有任何您创建的扩展策略或尚未删除的计划操作，这条命令会将它们删除。如果您要将此可扩展目标保留供将来使用，您可以跳过此操作。

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/my-table \
  --profile adminuser
```

删除 DynamoDB 表

使用以下 delete-table 命令删除本教程中使用的表。如果您要保留该表供将来使用，可以跳过此步骤。

```
aws dynamodb delete-table --table-name my-table \
  --profile adminuser
```

Application Auto Scaling 的目标跟踪扩展策略

在使用目标跟踪扩展策略时，可以选择一个扩展指标并设置一个目标值。Application Auto Scaling 创建和管理触发扩展策略的 CloudWatch 警报，并根据指标和目标值计算扩展调整。扩展策略根据需要增加或减少容量，将指标保持在指定的目标值或接近指定的目标值。除了将指标保持在目标值附近以外，目标跟踪扩展策略还会对由于负载模式变化而造成的指标变化进行调整。

您可以具有一个或多个目标跟踪扩展策略和/或一个或多个步进扩展策略。Application Auto Scaling 基于为扩展和缩减提供了最大容量的策略进行扩展。这样可以更灵活地覆盖多种场景，并确保始终有足够的容量来处理应用程序工作负载。

限制

- 无法为 Amazon EMR 集群或 AppStream 2.0 队列创建目标跟踪扩展策略。
- 您无法使用 Application Auto Scaling 创建或更新用于 AWS Auto Scaling 服务的扩展策略。有关 AWS Auto Scaling 控制台、CLI 或 API 操作的信息，请参阅 AWS Auto Scaling 文档。
- 您可以使用 Application Auto Scaling 基于预定义或 CloudWatch 自定义指标来应用目标跟踪扩展策略。不过，并非所有服务都可让您通过控制台管理自定义指标。要查看某个服务是否支持控制台中的自定义指标，请参阅该服务的文档。

注意事项

请注意以下事项：

- 并非所有指标都适用于目标跟踪。当指定自定义指标时，这可能很重要。指标必须是有效的使用率指标并且描述可扩展目标的繁忙程度。指标值必须根据可扩展目标的容量按比例增加或减少，以便指标数据可用于按比例扩展可扩展目标。
- 只要可能，您应按照具有 1 分钟频率的指标进行扩展，因为这可以确保更快地响应利用率变化。
- 目标跟踪扩展策略假设它应该在指定指标高于目标值时执行扩展。因此，不能使用目标跟踪扩展策略在指定指标低于目标值时向外扩展。
- 例如，如果因网络问题导致指定的指标数据不足，则目标跟踪扩展策略不会执行扩展。它不会执行向内扩展，因为它不会将数据不足解读为使用率低。
- 您可能会看到目标值与实际指标数据点之间存在差距。这是因为 Application Auto Scaling 在确定要添加或删除多少容量时将始终通过向上或向下舍入保守地进行操作，以免添加的容量不足或删除的容量过多。但是，对于具有小容量的可扩展目标，实际指标数据点可能看起来与目标值差距很大。
- 对于容量更高的可扩展目标，添加或删除容量将缩小目标值与实际指标数据点之间的差距。
- 为了确保应用程序可用性，Application Auto Scaling 针对指标尽快按比例扩展，但会逐渐缩减。
- 一个可扩展目标可以具有多个目标跟踪扩展策略，前提是它们分别使用不同的指标。Application Auto Scaling 的目的是始终优先考虑可用性，因此其行为会有所不同，具体取决于目标跟踪策略是否已准备好扩展或缩减。如果任何目标跟踪策略已准备好进行扩展，它将扩展可扩展目标，但仅在所有目标跟踪策略（启用了缩减部分）准备好缩减时才执行缩减。
- 您可以禁用目标跟踪扩展策略的缩减部分。利用此功能，您可以灵活地使用与用于向外扩展的方法不同的向内扩展方法。例如，您可以使用不同的扩展策略类型进行缩减，同时使用目标跟踪扩展策略进行扩展。
- 不要编辑或删除为目标跟踪扩展策略配置的 CloudWatch 警报。当您删除扩展策略时，将自动删除与目标跟踪扩展策略关联的 CloudWatch 警报。

冷却时间

向外扩展冷却时间 是一个向外扩展活动完成后、另一个向外扩展活动开始之前的时间量（秒）。尽管此冷却时间有效，但启动向外扩展事件所添加的容量将计算为下一向外扩展事件所需容量的一部分。旨在持续 (但不过度) 扩大。

向内扩展冷却时间 是活动中一次扩展完成之后、活动中另一次扩展开始之前的时间量（秒）。此冷却时间用于阻止后续向内扩展事件，直至冷却时间到期。旨在谨慎地缩小以保护您的应用程序的可用性。但是，如果事件中另一个警报在扩展之后的冷却时间内触发了向外扩展策略，Application Auto Scaling 将立即向外扩展您的可扩展目标。

注册可扩展目标

您必须注册可扩展目标，然后才能创建扩展策略。使用 register-scalable-target 命令注册新的可扩展目标。

以下示例使用 Application Auto Scaling 注册 Spot 队列请求。Application Auto Scaling 可以扩展 Spot 队列中的实例数量，最少 2 个实例，最多 10 个实例。

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

> **Note**
>
> 当您在控制台中配置扩展策略时，会自动向 Application Auto Scaling 将资源注册为可扩展目标。有关更多信息，请参阅开始使用 (p. 1)部分中的文档。

创建目标跟踪扩展策略

您可以创建目标跟踪扩展策略，来指示 Application Auto Scaling 在应用程序上的负载发生更改时自动横向扩展或缩减。

以下是将 CPU 平均使用率保持在 40% 的示例目标跟踪配置。将此配置保存在名为 `config.json` 的文件中。

```
{
  "TargetValue": 40.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
    }
}
```

或者，您可以通过创建自定义指标规范并为 CloudWatch 中的每个参数添加值来自定义用于扩展的指标。以下是一个示例目标跟踪配置，它将指定指标的平均利用率保持在 40%。

```
{
   "TargetValue":40.0,
   "CustomizedMetricSpecification":{
      "MetricName":"MyUtilizationMetric",
      "Namespace":"MyNamespace",
      "Dimensions":[
         {
```

```
            "Name":"MyOptionalMetricDimensionName",
            "Value":"MyOptionalMetricDimensionValue"
          }
      ],
      "Statistic":"Average",
      "Unit":"Percent"
   }
}
```

使用以下 put-scaling-policy 命令以及您创建的 config.json 文件，创建一个名为 cpu40-target-tracking-scaling-policy 的扩展策略。

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu40-target-tracking-scaling-policy --policy-type TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json
```

下面是示例输出。它包含代表您创建的两个 CloudWatch 警报的 ARN 和名称。

```
{
    "PolicyARN": "arn:aws-cn:autoscaling:region:account-id:scalingPolicy:policy-
id:resource/ec2/spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE:policyName/
cpu40-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws-cn:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-
a60f-3b36d653feca",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws-cn:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-
a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

描述目标跟踪扩展策略

您可使用以下 describe-scaling-policies 命令描述指定服务命名空间的所有扩展策略。

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2
```

您可使用 --query 参数筛选结果以仅显示目标跟踪扩展策略。此 query 语法仅在 Linux 或 macOS 上有效。在 Windows 上，请将单引号更改为双引号。

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 \
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

下面是示例输出。

```
[
```

```
    {
        "PolicyARN": "PolicyARN",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
            },
            "TargetValue": 40.0
        },
        "PolicyName": "cpu40-target-tracking-scaling-policy",
        "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "ServiceNamespace": "ec2",
        "PolicyType": "TargetTrackingScaling",
        "ResourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "Alarms": [
            {
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca",
                "AlarmARN": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
            },
            {
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
                "AlarmARN": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
            }
        ],
        "CreationTime": 1515021724.807
    }
]
```

删除目标跟踪扩展策略

在使用完目标跟踪扩展策略后，您可以使用 delete-scaling-policy 命令删除该策略。

以下命令将删除指定 Spot 队组请求的目标跟踪扩展策略。它还将删除 Application Auto Scaling 代表您创建的 CloudWatch 警报。

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu40-target-tracking-scaling-policy
```

Application Auto Scaling 的步进扩展策略

借助分步扩展，您可以为触发扩展流程的 CloudWatch 警报选择扩展指标和阈值，并定义当阈值违反指定数量的评估期时应如何扩展您的可扩展目标。

步进扩展策略根据一组扩展调整增加或减少可扩展目标的当前容量，这些调整称为步进调整。这些调整将根据警报违例规模发生变化。

在扩展活动启动后，该策略继续响应其他警报，甚至在进行扩展活动时也是如此。因此，在收到警报消息时，Application Auto Scaling 将评估触发的任何警报。

如果您的扩展指标是与可扩展目标的容量成比例增加或减少的利用率指标，我们建议您使用目标跟踪扩展策略。有关更多信息，请参阅 Application Auto Scaling 的目标跟踪扩展策略 (p. 11)。您仍然可以选择使用目标跟踪扩展与步进扩展来实现更高级的扩展策略配置。例如，如果需要，您可以在利用率达到特定级别时配置更积极的响应。

限制

- 您无法为 DynamoDB 表和全局辅助索引创建步进扩展策略。

扩展调整类型

在执行步进扩展策略时，将使用该策略中指定的扩展调整更改可扩展目标的当前容量。扩展调整无法将可扩展目标的容量更改为高于最大容量或低于最小容量。

对于步进扩展策略，Application Auto Scaling 支持以下调整类型：

- ChangeInCapacity — 将可扩展目标的当前容量增加或减少指定的值。正值将增加容量，负值将减小容量。

  示例：如果当前容量为 3 且调整为 5%，则 Application Auto Scaling 将为容量增加 5（总量为 8）。
- ExactCapacity — 将可扩展目标的当前容量更改为指定的值。为该调整类型指定一个正值。

  示例：如果当前容量为 3 且调整为 5，则 Application Auto Scaling 会将容量更改为 5。
- PercentChangeInCapacity — 将可扩展目标的当前容量增加或减少指定的百分比。正值将增加容量，负值将减小容量。如果得出的值不是整数，Application Auto Scaling 将进行舍入，如下所示：
  - 大于 1 的值向下取整。例如，`12.7` 取整为 `12`。
  - 0 和 1 之间的值舍入到 1。例如，`.67` 取整为 `1`。
  - 0 和 -1 之间的值舍入到 -1。例如，`-.58` 取整为 `-1`。
  - 小于 -1 的值向上取整。例如，`-6.67` 取整为 `-6`。

  示例：如果当前容量为 10 且调整值为 10%，则 Application Auto Scaling 将在容量中增加 1，总数为 11。

对于 PercentChangeInCapacity，您还可以指定最小扩展量（使用 `MinAdjustmentMagnitude` 参数）。例如，假定您创建一个增加 25% 的策略，并指定最小扩展量为 2。如果可扩展目标的容量为 4 并执行该扩展策略，4 的 25% 为 1。不过，由于您指定最小扩展量为 2，Application Auto Scaling 将增加 2。

步进调整

创建分步扩展策略时，将添加一个或多个分步调整，让您可以根据警报的严重程度进行扩展。每个分步调整指定以下内容：

- 指标值的下限
- 指标值的上限
- 要扩展的数量（基于扩展调整类型）

策略分步调整有一些规则：

- 分步调整范围不能重叠或有间隙。
- 只有一个分步调整可以有空下限 (负无穷)。如果一个分步调整有负下限，则必须有一个分步调整有空下限。
- 只有一个分步调整可以有空上限 (正无穷)。如果一个分步调整有正上限，则必须有一个分步调整有空上限。
- 同一分步调整中的上限和下限不能为空。
- 如果指标值高于违例阈值，则含下限而不含上限。如果指标值低于违例阈值，则不含下限而含上限。

CloudWatch 根据与您的 CloudWatch 警报关联的指标的统计信息聚合指标数据点。当突破警报阈值时，将触发相应的扩展策略。 Application Auto Scaling 将指定的聚合类型应用于 CloudWatch 中最新的指标数据点（与原始指标数据相对）。它将此聚合指标值与步进调整定义的上限和下限进行比较，以确定执行哪个步进调整。

您可以指定相对于违例阈值的上限和下限。例如，假定警报的违例阈值为 50，扩展调整类型为 `PercentChangeInCapacity`。还有具有以下步进调整的扩大和缩小策略：

Scale out policy			
Lower bound	Upper bound	Adjustment	Metric value
0	10	0	50 <= 值 < 60
10	20	10	60 <= 值 < 70
20	null	30	70 <= 值 < +无穷
Scale in policy			
Lower bound	Upper bound	Adjustment	Metric value
-10	0	0	40 < 值 <= 50
-20	-10	-10	30 < 值 <= 40
null	-20	-30	- 无穷 < 值 <= 30

您的可扩展目标的当前容量和所需容量均为 10。在聚合指标值大于 40 并小于 60 时，将保留当前容量和所需容量。

如果指标值增加到 60，Application Auto Scaling 会将组的所需容量增加 1 以达到 11。这基于向外扩展策略的第二个步进调整（增加 10 的 10%）。在增加新容量后，Application Auto Scaling 将当前容量增加到 11。如果即使在增加该容量后指标值仍增加到 70，Application Auto Scaling 会将目标容量增加 3 以达到 14。这基于向外扩展策略的第三个步进调整（增加 11 的 30%，即 3.3，向下舍入到 3）。

如果指标值达到 40，根据缩小策略的第二个步进调整 (减去 14 的 10%，即 1.4，向下舍入到 1)，Application Auto Scaling 将目标容量减少 1 以达到 13。如果在减少该容量后指标值仍降到 30，根据缩小策略的第三个步进调整 (减去 13 的 30%，即 3.9，向下舍入到 3)，Application Auto Scaling 将目标容量减小 3 以达到 10。

冷却时间

冷却时间 是指，在扩展活动完成后，以前的触发器相关扩展活动可能影响将来的扩展事件的时间（秒）。

尽管冷却时间有效，但启动向外扩展事件所添加的容量将计算为下一向外扩展事件所需容量的一部分。旨在持续 (但不过度) 扩大。例如，如果警报触发一个步进扩展策略以将容量增加 2，将成功完成扩展活动并开始计算冷却时间。如果警报在冷却时间内再次触，但进行了 3 这样更大幅度的步进调整，以前增加的 2 将被视为当前容量的一部分。因此，仅在容量中增加 1。

对于缩小策略，冷却时间用于阻止后续缩小事件，直至冷却时间到期。旨在谨慎地缩小以保护您的应用程序的可用性。但是，如果事件中另一个警报在扩展之后的冷却时间内触发了向外扩展策略，Application Auto Scaling 将立即向外扩展您的可扩展目标。

注册可扩展目标

您必须注册可扩展目标，然后才能创建扩展策略。使用 register-scalable-target 命令注册新的可扩展目标。

以下示例向 Application Auto Scaling 注册一个 Amazon ECS 服务。Application Auto Scaling 可以扩展任务的数量，最少 2 个任务，最多 10 个任务。

```
aws application-autoscaling register-scalable-target --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/sample-app-service \
--min-capacity 2 --max-capacity 10
```

Note

当您在控制台中配置扩展策略时，会自动向 Application Auto Scaling 将资源注册为可扩展目标。有关更多信息，请参阅开始使用 (p. 1)部分中的文档。

使用 AWS CLI 配置分步扩展策略

您可以创建分步扩展策略，来指示 Application Auto Scaling 当该应用程序上的负载发生变化时该怎么做。

下面是调整类型为 `ChangeInCapacity` 的示例分步配置，其基于以下分步调节增加可扩展目标的容量（假设 CloudWatch 警报阈值为 70%）：

- 当指标值大于或等于 70% 但小于 85% 时，将容量增加 1
- 当指标值大于或等于 85% 但小于 95% 时，将容量增加 2
- 当指标值大于或等于 95% 时，将容量增加 3

将此配置保存在名为 `config.json` 的文件中。

```
{
  "AdjustmentType": "ChangeInCapacity",
  "MetricAggregationType": "Average",
```

```
  "Cooldown": 60,
  "StepAdjustments": [
    {
      "MetricIntervalLowerBound": 0,
      "MetricIntervalUpperBound": 15,
      "ScalingAdjustment": 1
    },
    {
      "MetricIntervalLowerBound": 15,
      "MetricIntervalUpperBound": 25,
      "ScalingAdjustment": 2
    },
    {
      "MetricIntervalLowerBound": 25,
      "ScalingAdjustment": 3
    }
  ]
}
```

可以使用以下 put-scaling-policy 命令以及您创建的 config.json 文件创建一个名为 my-step-scaling-policy 的扩展策略：

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

输出包括作为策略唯一名称的 ARN。您需要使用它来创建 CloudWatch 警报。

```
{
    "PolicyARN": "arn:aws-cn:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/default/sample-app-service:policyName/my-step-scaling-policy"
}
```

最后，使用以下 CloudWatch put-metric-alarm 命令创建要与分步扩展策略结合使用的警报。在本示例中，您将根据平均 CPU 利用率发出警报。如果警报在至少两个连续 60 秒的评估期间达到 70% 的阈值，则它将被配置为处于 ALARM 状态。要指定其他 CloudWatch 指标或使用您自己的自定义指标，请在 --metric-name 中指定其名称并在 --namespace 中指定其命名空间。

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/default/sample-app-service \
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average \
  --period 60 --evaluation-periods 2 --threshold 70 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=ClusterName,Value=default,Name=ServiceName,Value=sample-app-service" \
  --alarm-actions PolicyARN
```

描述步进扩展策略

您可使用以下 describe-scaling-policies 命令描述指定服务命名空间的所有扩展策略。

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

您可以使用 --query 参数将结果筛选为仅步进扩展策略。此 query 语法仅在 Linux 或 macOS 上有效。在 Windows 上，请将单引号更改为双引号。

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs \
  --query 'ScalingPolicies[?PolicyType==`StepScaling`]'
```

下面是示例输出。

```
[
    {
        "PolicyARN": "PolicyARN",
        "StepScalingPolicyConfiguration": {
            "MetricAggregationType": "Average",
            "Cooldown": 60,
            "StepAdjustments": [
                {
                    "MetricIntervalLowerBound": 0.0,
                    "MetricIntervalUpperBound": 15.0,
                    "ScalingAdjustment": 1
                },
                {
                    "MetricIntervalLowerBound": 15.0,
                    "MetricIntervalUpperBound": 25.0,
                    "ScalingAdjustment": 2
                },
                {
                    "MetricIntervalLowerBound": 25.0,
                    "ScalingAdjustment": 3
                }
            ],
            "AdjustmentType": "ChangeInCapacity"
        },
        "PolicyType": "StepScaling",
        "ResourceId": "service/default/sample-app-service",
        "ServiceNamespace": "ecs",
        "Alarms": [
            {
                "AlarmName": "Step-Scaling-AlarmHigh-ECS:service/default/sample-app-
service",
                "AlarmARN": "arn:aws-cn:cloudwatch:region:012345678910:alarm:Step-Scaling-
AlarmHigh-ECS:service/default/sample-app-service"
            }
        ],
        "PolicyName": "my-step-scaling-policy",
        "ScalableDimension": "ecs:service:DesiredCount",
        "CreationTime": 1515024099.901
    }
]
```

删除步进扩展策略

当您不再需要某个步进扩展策略时，可将其删除。要删除的扩展策略和 CloudWatch 警报，请完成以下任务。

删除您的扩展策略

使用以下 delete-scaling-policy 命令：

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --policy-name my-step-scaling-policy
```

删除 CloudWatch 警报

使用 delete-alarms 命令。您可以一次删除一个或多个警报。例如，使用以下命令可删除 `Step-Scaling-AlarmHigh-ECS:service/default/sample-app-service` 和 `Step-Scaling-AlarmLow-ECS:service/default/sample-app-service` 警报。

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-ECS:service/default/
sample-app-service Step-Scaling-AlarmLow-ECS:service/default/sample-app-service
```

Application Auto Scaling 的计划扩展

按计划扩展使您可以按照可预测的负载变化来设置您自己的扩展计划。例如，您的 Web 应用程序的流量会在每周的星期三开始增加，并在星期四保持高流量状态，然后在星期五开始下降。您可以配置 Application Auto Scaling 在星期三增加容量并在星期五减少容量。

要使用计划的扩展，请创建指示 Application Auto Scaling 在特定时间执行扩展活动的计划的操作。创建计划的操作时，请指定可扩展目标、扩展活动执行的时间以及最小和最大容量。在指定时间，Application Auto Scaling 将基于新的容量值进行扩展。

Application Auto Scaling 将保证同一可扩展目标的计划操作的执行顺序，但不保证跨可扩展目标的计划操作的执行顺序。

Note

为简洁起见，本主题中的示例说明了用于 Amazon ECS、DynamoDB、AppStream 2.0、Spot 队列和自定义资源的 CLI 命令。要指定不同的可扩展目标，请在 `--service-namespace` 中指定其命名空间，在 `--scalable-dimension` 中指定其可扩展维度，并在 `--resource-id` 中指定其资源 ID。

注册可扩展目标

您必须注册可扩展目标，然后才能创建计划的操作。使用 register-scalable-target 命令注册新的可扩展目标。

以下示例使用 Application Auto Scaling 注册 Spot 队列请求。Application Auto Scaling 可以扩展 Spot 队列中的实例数量，最少 2 个实例，最多 10 个实例。

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

使用 AWS CLI 创建或更新计划的操作

您可以使用 put-scheduled-action 命令创建并更新仅扩展一次或按重复计划扩展的已计划的操作。当您指定新容量时，可指定最小容量和/或最大容量。

计划一次性计划的操作

您可以指定一次性计划以在特定日期和时间（采用 UTC 表示）自动扩展您的可扩展目标。

Example 示例：仅向内扩展一次

在为 `--schedule` 指定的日期和时间，如果为 `MaxCapacity` 指定的值低于当前容量，则 Application Auto Scaling 将向内扩展至 `MaxCapacity`。

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
```

```
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/web-app \
  --scheduled-action-name my-one-time-action \
  --schedule "at(2019-01-31T17:00:00)" \
  --scalable-target-action MaxCapacity=10
```

Example 示例：仅向外扩展一次

在为 `--schedule` 指定的日期和时间，如果为 `MinCapacity` 指定的值高于当前容量，则 Application Auto Scaling 将向外扩展至 `MinCapacity`。

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name my-one-time-action \
  --schedule "at(2019-03-31T22:00:00)" \
  --scalable-target-action MinCapacity=3
```

`custom-resource-id.txt` 文件为您的自定义资源指定 API 网关 终端节点。有关配置自定义资源的更多信息，请参阅我们的 GitHub 存储库。文件的内容可能如下所示：

Example

```
https://example.execute-api.region.amazonaws.com/prod/scalableTargetDimensions/1-23456789
```

按照定期计划来计划操作

您可以使用 put-scheduled-action 命令的 `--schedule` 选项为计划的操作指定重复时间表。Application Auto Scaling 支持计划表达式的 cron 和 rate 格式。有关更多信息，请参阅 Amazon CloudWatch Events 用户指南中的 Cron 表达式和 Rate 表达式。

Example 示例：使用 cron 表达式按照重复时间表扩展

对于指定的计划（每天中午 12:00 (UTC)），如果为 `MinCapacity` 指定的值高于当前容量，则 Application Auto Scaling 将向外扩展至 `MinCapacity`。如果为 `MaxCapacity` 指定的值低于当前容量，则 Application Auto Scaling 将向内扩展到 `MaxCapacity`。

```
aws application-autoscaling put-scheduled-action --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/my-table \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 12 * * ? *)" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Example 示例：使用 rate 表达式按照重复时间表扩展

对于指定的计划（每小时），如果为 `MinCapacity` 指定的值高于当前容量，则 Application Auto Scaling 将向外扩展至 `MinCapacity`。如果为 `MaxCapacity` 指定的值低于当前容量，则 Application Auto Scaling 将向内扩展到 `MaxCapacity`。

```
aws application-autoscaling put-scheduled-action --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(1 hour)" \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

描述计划的操作

您可使用以下 describe-scheduled-actions 命令描述指定服务命名空间的所有计划的操作。

```
aws application-autoscaling describe-scheduled-actions --service-namespace ecs
```

下面是示例输出。

```
{
    "ScheduledActions": [
        {
            "ScheduledActionARN": "<arn>",
            "ServiceNamespace": "ecs",
            "CreationTime": 1515026382.218,
            "ScalableDimension": "ecs:service:DesiredCount",
            "Schedule": "at(2018-01-31T17:00:00)",
            "ScalableTargetAction": {
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-one-time-action",
            "ResourceId": "service/default/web-app"
        }
    ]
}
```

删除计划的操作

在使用完计划的操作后，您可以使用 delete-scheduled-action 命令将其删除。

以下命令将删除指定的可扩展目标的指定计划操作。

```
aws application-autoscaling delete-scheduled-action --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --scheduled-action-name my-spot-fleet-action
```

暂停和恢复 Application Auto Scaling 的扩展

本主题说明如何暂停然后恢复应用程序中可扩展目标的一个或多个扩展活动。暂停-恢复功能用于临时暂停由您的扩展策略和计划操作触发的扩展活动。暂停-恢复功能非常有用，例如，当您更改或调查配置问题时，不希望自动扩展潜在产生干扰。您可以保留您的扩展策略和计划操作，在您准备就绪时，可以恢复扩展活动。

扩展活动

Application Auto Scaling 支持将以下扩展活动置于暂停状态：

- 由扩展策略触发的所有缩减活动。
- 由扩展策略触发的所有横向扩展活动。
- 涉及计划操作的所有扩展活动。

以下描述说明了暂停各个扩展活动时会发生什么。每个扩展活动都可以单独暂停和恢复。根据暂停扩展活动的原因，您可能需要一起暂停多个扩展活动。

`DynamicScalingInSuspended`

- 在触发目标跟踪扩展策略或分步扩展策略时，Application Auto Scaling 不会删除容量。这使您可以暂时禁用与扩展策略关联的缩减活动，而不删除扩展策略或其关联的 CloudWatch 警报。当您恢复缩减时，Application Auto Scaling 会评估具有当前违反的警报阈值的策略。

`DynamicScalingOutSuspended`

- 在触发目标跟踪扩展策略或分步扩展策略时，Application Auto Scaling 不会增加容量。这使您可以暂时禁用与扩展策略关联的横向扩展活动，而不删除扩展策略或其关联的 CloudWatch 警报。当您恢复横向扩展时，Application Auto Scaling 会评估具有当前违反的警报阈值的策略。

`ScheduledScalingSuspended`

- 在暂停期间，Application Auto Scaling 不启动计划要运行的扩展操作。当您恢复计划的扩展时，Application Auto Scaling 仅评估尚未经过执行时间的计划操作。

使用 AWS CLI 暂停和恢复扩展活动

您可以暂停和恢复 Application Auto Scaling 可扩展目标的单个或所有扩展活动。

> **Note**
>
> 为简洁起见，这些示例说明了如何暂停和恢复 DynamoDB 表的扩展。要指定不同的可扩展目标，请在 `--service-namespace` 中指定其命名空间，在 `--scalable-dimension` 中指定其可扩展维度，并在 `--resource-id` 中指定其资源 ID。

暂停扩展活动

打开一个命令行窗口，然后使用 register-scalable-target 命令和 `--suspendedstate` 选项，如下所示。

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

要仅暂停某个扩展策略触发的缩减活动，请在 config.json 中指定以下内容。

```
{
    "DynamicScalingInSuspended":true
}
```

要仅暂停某个扩展策略触发的横向扩展活动，请在 config.json 中指定以下内容。

```
{
    "DynamicScalingOutSuspended":true
}
```

要仅暂停涉及计划操作的扩展活动，请在 config.json 中指定以下内容。

```
{
    "ScheduledScalingSuspended":true
}
```

暂停所有扩展活动

将 register-scalable-target 命令与 `--suspendedstate` 选项一起使用，如下所示。

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

此示例假定文件 config.json 包含以下 JSON 格式的参数。

```
{
    "DynamicScalingInSuspended":true,
    "DynamicScalingOutSuspended":true,
    "ScheduledScalingSuspended":true
}
```

查看暂停的扩展活动

使用 describe-scalable-targets 命令可确定可扩展目标处于暂停状态的扩展活动。

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

下面是示例输出。

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "dynamodb",
            "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
            "ResourceId": "table/my-table",
```

```
            "MinCapacity": 1,
            "MaxCapacity": 20,
            "SuspendedState": {
                "DynamicScalingOutSuspended": true,
                "DynamicScalingInSuspended": true,
                "ScheduledScalingSuspended": true
            },
            "CreationTime": 1558125758.957,
            "RoleARN": "arn:aws-cn:iam::123456789012:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable"
        }
    ]
}
```

恢复扩展活动

当您准备好恢复扩展活动时，可以使用 register-scalable-target 命令恢复它。

以下示例命令恢复指定的可扩展目标的所有扩展活动。

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

此示例假定文件 config.json 包含以下 JSON 格式的参数。

```
{
    "DynamicScalingInSuspended":false,
    "DynamicScalingOutSuspended":false,
    "ScheduledScalingSuspended":false
}
```

Application Auto Scaling 的身份验证和访问控制

访问和 Application Auto Scaling 需要可供 AWS 用来验证您的请求的凭证。这些凭证必须具有权限才能执行 Application Auto Scaling 操作，例如配置扩展策略。

本主题提供了详细信息来说明如何使用 AWS Identity and Access Management (IAM) 控制哪些用户可执行 Application Auto Scaling 操作，从而对您的资源进行保护。

默认情况下，全新的 IAM 用户没有执行任何操作的权限。要授予调用 Application Auto Scaling 操作的权限，您可以将 IAM 策略附加到需要获得权限的 IAM 用户或组。

在策略中指定操作

Application Auto Scaling 提供了一组可在 IAM 策略中指定的操作。有关更多信息，请参阅 Application Auto Scaling API 参考 中的操作。

要指定单个策略，您可以在操作名称中使用以下前缀：`application-autoscaling:`。例如：

```
"Action": "application-autoscaling:DescribeScalingActivities"
```

支持通配符。例如，您可以使用 `application-autoscaling:*` 指定所有 Application Auto Scaling 操作。

```
"Action": "application-autoscaling:*"
```

您还可以使用 `Describe*` 指定名称以 `Describe` 开头的所有操作。

```
"Action": "application-autoscaling:Describe*"
```

除了调用 Application Auto Scaling 操作的权限之外，用户还需要创建服务相关角色的权限。

当用户调用 `RegisterScalableTarget` 时，如果服务相关角色不存在，Application Auto Scaling 会在您的账户中创建该角色。此服务相关角色向 Application Auto Scaling 授予权限，以便它能代表您调用其他服务。

为使自动角色创建操作成功，用户必须具有 `iam:CreateServiceLinkedRole` 操作的权限。

```
"Action": "iam:CreateServiceLinkedRole"
```

有关更多信息，请参阅 Application Auto Scaling 的服务相关角色 (p. 30)。

指定资源

Application Auto Scaling 没有可用作 IAM 策略声明的 `Resource` 元素的服务定义的资源。因此，IAM 策略中没有可供您使用的 Amazon 资源名称 (ARN)。要控制对 Application Auto Scaling 操作的访问，请在编写 IAM 策略时始终使用 * (星号) 作为资源。

在策略中指定条件

当您授予权限时，可使用 IAM 策略语言来指定规定策略何时生效的条件。例如，您可能希望策略仅在特定日期后应用。要表示条件，请使用预定义的条件键。

有关每个 AWS 服务支持的上下文键的列表以及 AWS 范围的策略键的列表，请参阅 IAM 用户指南 中的 AWS 服务的操作、资源和条件键和 AWS 全局条件上下文键。

Application Auto Scaling 未提供额外的条件键。

示例策略

要为可扩展资源配置步进扩展策略或目标跟踪策略，用户必须具有相应的权限以使用以下示例策略中的操作：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScalingPolicy",
              "application-autoscaling:DescribeScalingPolicies",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScalingPolicy",
              "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*"
        }
    ]
}
```

要为可扩展资源配置计划扩展，用户必须具有相应的权限以使用以下示例策略中的操作：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScheduledAction",
              "application-autoscaling:DescribeScheduledActions",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScheduledAction",
              "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*"
        }
    ]
}
```

其他 IAM 权限

对于用户将配置扩展策略的每种类型的资源，用户必须具有额外的权限。您还可以在 IAM 策略声明的 `Action` 元素中指定以下操作。

AppStream 2.0 队列

- `appstream:DescribeFleets`
- `appstream:UpdateFleet`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

DynamoDB 表和全局二级索引

- `dynamodb:DescribeTable`
- `dynamodb:UpdateTable`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

Amazon EC2 Spot 队列请求

- `ec2:DescribeSpotFleetRequests`
- `ec2:ModifySpotFleetRequest`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

Amazon ECS 服务

- `ecs:DescribeServices`
- `ecs:UpdateServices`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

Amazon EMR 集群

- `elasticmapreduce:ModifyInstanceGroups`
- `elasticmapreduce:ListInstanceGroups`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

Aurora 数据库集群

- `rds:AddTagsToResource`

- rds:CreateDBInstance
- rds:DeleteDBInstance
- rds:DescribeDBClusters
- rds:DescribeDBInstances
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

Amazon SageMaker 终端节点

- sagemaker:DescribeEndpoint
- sagemaker:DescribeEndpointConfig
- sagemaker:UpdateEndpointWeightsAndCapacities
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

自定义资源

- execute-api:Invoke
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

Application Auto Scaling 的服务相关角色

Application Auto Scaling 使用服务相关角色获取代表您调用其他 AWS 服务所需的权限。服务相关的角色是一种与 AWS 服务直接关联的独特类型的 AWS Identity and Access Management (IAM) 角色。

服务相关角色提供了一种将权限委托给 AWS 服务的安全方式，因为只有相关服务才能代入服务相关角色。有关更多信息，请参阅 IAM 用户指南 中的使用服务相关角色。

只有在首先删除角色的相关资源后，才能删除角色。这将保护您的资源，因为您不会无意中删除对资源的访问权限。

服务相关角色授予的权限

Application Auto Scaling 使用以下服务相关角色代表您管理扩展操作。对于每个可扩展资源类型，都有一个服务相关角色。在每种情况下，该服务相关角色都是一个包含所有必要权限的预定义角色。每个服务相关角色信任指定的服务委托人来代入该角色。

AWSServiceRoleForApplicationAutoScaling_AppStreamFleet

操作:

- appstream:DescribeFleets
- appstream:UpdateFleet
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms

- cloudwatch:PutMetricAlarm

服务委托人：appstream.application-autoscaling.amazonaws.com

AWSServiceRoleForApplicationAutoScaling_DynamoDBTable

操作:

- dynamodb:DescribeTable
- dynamodb:UpdateTable
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

服务委托人：dynamodb.application-autoscaling.amazonaws.com

AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest

操作:

- ec2:DescribeSpotFleetRequests
- ec2:ModifySpotFleetRequest
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

服务委托人：ec2.application-autoscaling.amazonaws.com

AWSServiceRoleForApplicationAutoScaling_ECSService

操作:

- ecs:DescribeServices
- ecs:UpdateService
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms
- cloudwatch:PutMetricAlarm

服务委托人：ecs.application-autoscaling.amazonaws.com

AWSServiceRoleForApplicationAutoScaling_RDSCluster

操作:

- rds:AddTagsToResource
- rds:CreateDBInstance
- rds:DeleteDBInstance
- rds:DescribeDBClusters
- rds:DescribeDBInstance
- cloudwatch:DeleteAlarms
- cloudwatch:DescribeAlarms

- `cloudwatch:PutMetricAlarm`

服务委托人：`rds.application-autoscaling.amazonaws.com`

`AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`

操作:

- `sagemaker:DescribeEndpoint`
- `sagemaker:DescribeEndpointConfig`
- `sagemaker:UpdateEndpointWeightsAndCapacities`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

服务委托人：`sagemaker.application-autoscaling.amazonaws.com`

`AWSServiceRoleForApplicationAutoScaling_CustomResource`

操作:

- `execute-api:Invoke`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`

服务委托人：`custom-resource.application-autoscaling.amazonaws.com`

您必须配置权限以允许 IAM 实体 (例如，用户、组或角色) 创建、编辑或删除服务相关角色。有关更多信息，请参阅 IAM 用户指南 中的使用服务相关角色。

创建服务相关角色（自动）

在大多数情况下，您无需手动创建服务相关角色。Application Auto Scaling 将在您调用 `RegisterScalableTarget` 时为您创建相应的服务相关角色。例如，如果您已为某个 Amazon ECS 服务设置自动扩展，则 Application Auto Scaling 会创建 `AWSServiceRoleForApplicationAutoScaling_ECSService` 角色。

> **Important**
>
> 调用 `RegisterScalableTarget` 操作的 IAM 用户必须具有适当的 IAM 权限才能创建服务相关角色。否则，自动创建操作将失败。有关更多信息，请参阅 IAM 用户指南 中的服务相关角色权限或本指南中有关所需的用户权限 (p. 27)的信息。

创建服务相关角色（手动）

您可以使用 IAM 控制台、AWS CLI 或 IAM API 创建服务相关角色。有关更多信息，请参阅 IAM 用户指南 中的创建服务相关角色。

编辑服务相关角色

对于 Application Auto Scaling 创建的服务相关角色，您只能编辑其描述。有关更多信息，请参阅 IAM 用户指南中的编辑服务相关角色。

删除服务相关角色

如果您不再将 Application Auto Scaling 用于某种类型的可扩展资源，我们建议您删除相应的服务相关角色。只有在首先删除相关可扩展资源后，才能删除服务相关角色。有关更多信息，请参阅有关可扩展资源的文档。例如，要删除 ECS 服务，请参阅 Amazon Elastic Container Service Developer Guide 中的删除服务。

要删除服务相关角色，您可以使用 IAM 控制台、IAM CLI 或 IAM API。有关更多信息，请参阅 IAM 用户指南中的删除服务相关角色。

在删除某个服务相关角色后，当您调用 `RegisterScalableTarget` 时，Application Auto Scaling 将重新创建该角色。

Application Auto Scaling 服务相关角色的受支持区域

Application Auto Scaling 支持在服务可用的所有区域中使用服务相关角色。有关更多信息，请参阅 AWS Regions and Endpoints。

Application Auto Scaling 限制

您的 AWS 账户存在以下与 Application Auto Scaling 相关的限制。要请求提高限制，请使用Application Auto Scaling限制表单。

每账户每区域的默认限制

项目	默认限制	备注
每个资源类型的最大可扩展目标数	Amazon DynamoDB：3000 所有其他资源类型：500	确保在有关提高限制的请求中指定资源的类型，例如 Amazon ECS 或 DynamoDB。
每个可扩展目标的最大扩展策略数	50	这包含步进扩展策略和目标跟踪策略。
每个可扩展目标的最大计划操作数	200	
每个步进扩展策略的最大步进调整数	20	

有关 AWS 其他服务的服务限制的信息，请参阅 Amazon Web Services 一般参考 中的 AWS 服务限制。

文档历史记录

下表介绍了自 2018 年 1 月以来对 Application Auto Scaling 文档的重要补充。如需对此文档更新的通知，您可以订阅 RSS 源。

update-history-change	update-history-description	update-history-date
暂停和恢复扩展 (p. 35)	增加了对暂停和恢复扩展的支持。有关更多信息，请参阅 Application Auto Scaling 的暂停和恢复扩展。	August 29, 2019
新章节 (p. 35)	Application Auto Scaling 文档中增加了设置章节。对整个用户指南进行了少量改进和修复。	June 28, 2019
指南更改 (p. 35)	改进了 Application Auto Scaling 文档中的计划的扩展、步进扩展策略和目标跟踪扩展策略部分。	March 11, 2019
添加对自定义资源的支持 (p. 35)	使用 Application Auto Scaling 扩展由您自己的应用程序或服务提供的自定义资源。有关更多信息，请参阅我们的 GitHub 存储库。	July 9, 2018
增加对 Amazon SageMaker 终端节点变体的支持 (p. 35)	使用 Application Auto Scaling 扩展为变体预配置的终端节点实例数。	February 28, 2018

下表介绍了 2018 年 1 月之前对 Application Auto Scaling 文档的一些重要更改。

更改	描述	日期
添加对 Aurora 副本的支持	使用 Application Auto Scaling 扩展所需计数。有关更多信息，请参阅 Amazon RDS 用户指南 中的将 Amazon Aurora Auto Scaling 与 Aurora 副本结合使用。	2017 年 11 月 17 日
添加对计划扩展的支持	使用计划的扩展在特定预设时间或按照特定预设间隔扩展资源。有关更多信息，请参阅计划的 Application Auto Scaling 扩展。	2017 年 11 月 8 日
添加对目标跟踪扩展策略的支持	使用目标跟踪扩展策略通过几个简单步骤为您的应用程序设置动态扩展。有关更多信息，请参阅 Application Auto Scaling 的目标跟踪扩展策略。	2017 年 12 月 7 日

更改	描述	日期
添加对 DynamoDB 表和全局二级索引的预置读取和写入容量的支持	使用 Application Auto Scaling 扩展预置的吞吐容量。有关更多信息，请参阅 Amazon DynamoDB 开发人员指南 中的使用 DynamoDB Auto Scaling 管理吞吐容量。	2017 年 6 月 14 日
添加对 AppStream 2.0 队列的支持	使用 Application Auto Scaling 扩展队列的大小。有关更多信息，请参阅 Amazon AppStream 2.0 开发人员指南 中的 AppStream 2.0 的队组 Auto Scaling。	2017 年 3 月 23 日
添加对 Amazon EMR 集群的支持	使用 Application Auto Scaling 扩展核心节点和任务节点。有关更多信息，请参阅 Amazon EMR 管理指南 中的在 Amazon EMR 中使用 Automatic Scaling。	2016 年 11 月 18 日
增加对 Spot 队列的支持	使用 Application Auto Scaling 扩展目标容量。有关更多信息，请参阅 Amazon EC2 用户指南（适用于 Linux 实例） 中的 Spot 队组的 Automatic Scaling。	2016 年 9 月 1 日
添加对 Amazon ECS 服务的支持	使用 Application Auto Scaling 扩展所需计数。有关更多信息，请参阅 Amazon Elastic Container Service Developer Guide 中的 Auto Scaling 服务。	2016 年 8 月 9 日