aws

用户指南

Amazon S3 on Outposts

API 版本 2006-03-01

Amazon S3 on Outposts: 用户指南

Table of Contents

什么是 Amazon S3 on Outposts？

AWS Outposts 是一项完全托管式服务，它为几乎任何数据中心、主机托管空间或本地部署设施提供相同的 AWS 基础架构、AWS 服务、API 和工具，以实现真正一致的混合体验。AWS Outposts 非常适合需要低延迟访问本地部署系统、本地数据处理、数据驻留以及迁移具有本地系统相互依赖项的应用程序的工作负载。有关更多信息，请参阅《AWS Outposts 用户指南》中的什么是 AWS Outposts？。

借助 Amazon S3 on Outposts，您可以在 Outposts 上创建 S3 桶并在本地中轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 OUTPOSTS，该存储类使用 Amazon S3 API，并且用于在 Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过虚拟私有云（VPC）使用访问点和端点连接与 Outposts 桶进行通信。

您可以像在 Amazon S3 中一样在 Outposts 存储桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。

S3 on Outposts 的工作原理

S3 on Outposts 是一种对象存储服务，可将数据以对象形式存储在 Outpost 上的存储桶中。对象 是指一个数据文件和描述该文件的任何元数据。存储桶是对象的容器。

要将数据存储在 S3 on Outposts 中，您需要先创建存储桶。创建存储桶时，您可以指定存储桶名称和将容纳该存储桶的 Outpost。要访问 S3 on Outposts 存储桶并执行对象操作，接下来要创建并配置访问点。您还必须创建端点以将请求路由到访问点。

访问点简化了在 S3 中存储数据的任何 AWS 服务或客户应用程序的数据访问。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行对象操作（如 `GetObject` 和 `PutObject`）。每个访问点都有不同的权限和网络控制。

您可以使用 AWS 管理控制台、AWS CLI、AWS SDK 或 REST API 创建和管理 S3 on Outposts 存储桶、访问点和端点。要上传和管理 S3 on Outposts 存储桶中的对象，您可以使用AWS CLI、AWS SDK 或 REST API。

Regions

在 AWS Outposts 调配期间，您或 AWS 将创建服务链路连接，将您的 Outpost 连接回您选择的 AWS 区域 或者用于存储桶操作和遥测的 Outposts 所属区域。Outposts 依赖于与父 AWS 区域 的连接。Outposts 机架不是为断开连接的操作或连接有限或没有连接的环境而设计的。有关更多信息，请参阅《AWS Outposts 用户指南》中的与 AWS 区域 的 Outpost 连接。

桶

存储桶是用于 S3 on Outposts 中存储的对象的容器。您可以在存储桶中存储任意数量的对象，并且每个 Outpost 每个账户中最多可以有 100 个存储桶。

创建存储桶时，您可以输入存储桶名称，然后选择存储桶将驻留的 Outpost。创建存储桶后，无法更改存储桶名称或将存储桶移到不同的 Outpost。存储桶名称必须遵循 Amazon S3 存储桶命名规则。在 S3 on Outposts 中，存储桶名称对于 Outposts 和 AWS 账户 是唯一的。S3 on Outposts 存储桶需要 `outpost-id`、`account-id` 以及用于识别它们的存储桶名称。

以下示例显示了用于 S3 on Outposts 存储桶的 Amazon Resource Name（ARN）格式。ARN 包括 Outpost 所属的区域、您的 Outpost 账户、Outpost ID 和存储桶名称。

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

每个对象都储存在一个存储桶中。您必须使用访问点访问 Outposts 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 ARN 或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 的访问点 ARN 格式，其中包括 `outpost-id`、`account-id` 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关存储桶的更多信息，请参阅 使用 S3 on Outposts 存储桶。

对象

对象是 S3 on Outposts 中存储的基础实体。对象由对象数据和元数据组成。元数据是一组描述对象的名称-值对。这些对值包括一些默认元数据（如上次修改日期）和标准 HTTP 元数据（如 `Content-Type`）。您还可以在存储对象时指定自定义元数据。存储桶中的对象将由键（或名称）进行唯一地标识。

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

密钥

对象密钥（或密钥名称）是指存储桶中对象的唯一标识符。存储桶内的每个对象都只能有一个键。存储桶和对象键的组合唯一标识各个对象。

以下示例显示了 S3 on Outposts 对象的 ARN 格式，其中包括 Outpost 所在区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID、存储桶名称和对象键：

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket1/object/myobject
```

有关对象键的更多信息，请参阅 使用 S3 on Outposts 对象。

S3 版本控制

您可以对 Outposts 桶使用 S3 版本控制功能，以将对象的多个变体保留在同一桶中。使用 S3 版本控制功能，您可以保留、检索和恢复存储桶中的各个版本。S3 版本控制功能可帮助您从用户意外操作和应用程序故障中恢复。

有关更多信息，请参阅 为 S3 on Outposts 桶管理 S3 版本控制。

版本 ID

在桶中启用 S3 版本控制时，S3 on Outposts 会为添加到桶中的每个对象生成唯一的版本 ID。启用版本控制时存在于存储桶中的对象的版本 ID 为`null`。如果使用其他操作修改这些（或任何其他）对象，例如 PutObject，则新对象将获得唯一的版本 ID。

有关更多信息，请参阅 为 S3 on Outposts 桶管理 S3 版本控制。

存储类别和加密

S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)。S3 Outposts 存储类仅可用于存储在 AWS Outposts 上存储桶中的对象。如果您尝试将其他 S3 存储类与 S3 on Outposts 一起使用，则 S3 on Outposts 将返回 `InvalidStorageClass` 错误消息。

默认情况下，存储在 S3 Outposts (OUTPOSTS) 存储类中的对象使用服务器端加密方式和由 Amazon S3 托管的加密密钥 (SSE-S3) 进行加密。有关更多信息，请参阅 S3 on Outposts 中的数据加密。

存储桶策略

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。

存储桶策略使用基于 JSON 的 IAM 策略语言，该语言是跨 AWS 的标准语言。您可以使用存储桶策略添加或拒绝存储桶中对象的权限。存储桶策略基于策略中的元素允许或拒绝请求。这些元素可以包括请求者、S3 on Outposts 操作、资源以及请求的特征或条件（例如，用于发出请求的 IP 地址）。例如，您可以创建一个存储桶策略，该策略授予跨账户将对象上传到 S3 on Outposts 存储桶的权限，同时确保存储桶拥有者对上传的对象拥有完全控制权。

在存储桶策略中，您可以在 ARN 和其他值上使用通配符 (*) 来授予对对象子集的权限。例如，您可以控制对以通用前缀或以给定扩展名结尾的对象组的访问，例如 `.html`。

S3 on Outposts 访问点

S3 on Outposts 访问点被命名为网络端点，其专用访问策略描述了如何使用该端点访问数据。访问点可简化对 S3 on Outposts 中的共享数据集的大规模数据访问管理。访问点附加到存储桶，您可以使用这些存储桶执行 S3 对象操作（如 `GetObject` 和 `PutObject`）。

当您为对象操作指定桶时，可以使用访问点 ARN 或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

访问点具有不同的权限和网络控制，S3 on Outposts 将它们应用于通过该访问点发出的任何请求。每个接入点强制实施自定义接入点策略，该策略与附加到底层存储桶的存储桶策略结合使用。

有关更多信息，请参阅 访问 S3 on Outposts 存储桶和对象。

S3 on Outposts 的功能

访问权限管理

S3 on Outposts 提供了用于审核和管理对存储桶和对象的访问的功能。默认情况下，S3 on Outposts 存储桶和对象都是私有的。您只能访问您创建的 S3 on Outposts 资源。

要授予支持您的特定使用案例的细粒度资源权限或审核 S3 on Outposts 资源的权限，您可以使用以下功能。

- S3 阻止公有访问 - 阻止对存储桶和对象的公有访问。对于 Outposts 上的存储桶，默认情况下始终启用 Block Public Access（阻止公共访问）。
- AWS Identity and Access Management（IAM） – IAM 是一种 Web 服务，可帮助您安全地控制对 AWS 资源（包括 S3 on Outposts 资源）的访问。借助 IAM，您可以集中管理控制用户可访问哪些 AWS 资源的权限。可以使用 IAM 来控制谁通过了身份验证（准许登录）并获得授权（具有相应权限）来使用资源。
- S3 on Outposts 访问点 – 管理对 S3 on Outposts 中的共享数据集的数据访问。访问点是具有专用访问策略的命名网络端点。访问点附加到存储桶，可用于执行对象操作（如 `GetObject` 和 `PutObject`）。
- 存储桶策略 - 使用基于 IAM 的策略语言为 S3 存储桶及其中的对象配置基于资源的权限。
- AWS Resource Access Manager (AWS RAM) – 跨您的组织或 AWS Organizations 中组织单位 (OU) 内的 AWS 账户 安全共享 S3 on Outposts 容量。

存储日志记录和监控

S3 on Outposts 提供日志记录和监控工具，您可以使用这些工具来监控和控制 S3 on Outposts 资源的使用情况。更多信息，请参阅监控工具。

- 针对 S3 on Outposts 的 Amazon CloudWatch 指标 – 跟踪资源的运行状况并了解容量可用性。
- 针对 S3 on Outposts 的 Amazon CloudWatch Events 事件 – 为任何 S3 on Outposts API 事件创建规则，以便通过所有受支持的 CloudWatch Events 目标接收通知，包括 Amazon Simple Queue Service (Amazon SQS)、Amazon Simple Notification Service (Amazon SNS) 和 AWS Lambda。
- S3 on Outposts 的 AWS CloudTrail 日志 – 记录用户、角色或 S3 on Outposts 中的 AWS 服务 执行的操作。CloudTrail 日志为您提供了 S3 存储桶级和对象级操作的详细 API 跟踪。

强一致性

S3 on Outposts 为所有 AWS 区域 内的 S3 on Outposts 存储桶中对象的 PUT 和 DELETE 请求提供了强大的先写后读一致性。这个行为既适用于新对象的写入，也适用于覆盖现有对象的 PUT 请求以及 DELETE 请求。此外，S3 on Outposts 对象标签和对象元数据（例如 HEAD 对象）具有严格的一致性。有关更多信息，请参阅《Amazon S3 用户指南》中的 Amazon S3 数据一致性模型。

相关服务

将数据加载到 S3 on Outposts 中之后，您可以将数据用于其他 AWS 服务。以下是您可能最常用使用的服务：

- Amazon Elastic Compute Cloud (Amazon EC2) – 在 AWS Cloud中提供安全可扩展的计算容量。使用 Amazon EC2 可减少前期的硬件投入，因此您能够快速开发和部署应用程序。您可以使用 Amazon EC2 启动所需数量的虚拟服务器，配置安全性和联网以及管理存储。
- Outposts 上的 Amazon Elastic Block Store (Amazon EBS) – 使用 Outposts 上的 Amazon EBS 本地快照将卷快照存储在 S3 on Outposts 本地。
- Outposts 上的 Amazon Relational Database Service (Amazon RDS) – 使用 Amazon RDS 本地备份将您的 Amazon RDS 备份存储在您的 Outpost 本地。
- AWS DataSync – 在 Outposts 和 AWS 区域 之间自动传输数据，同时选择要传输的内容、何时传输以及要使用多少网络带宽。S3 on Outposts 已与 AWS DataSync 集成。对于需要高吞吐量本地处理的本地应用程序，S3 on Outposts 提供了本地对象存储，以最大限度地减少网络变化造成的数据传输和缓冲区，同时让您能够轻松地在 Outposts 和 AWS 区域 之间传输数据。

访问 S3 on Outposts

您可以通过以下任何方式使用 S3 on Outpost：

AWS 管理控制台

控制台是基于 Web 的用户界面，用于管理 S3 on Outposts 和 AWS 资源。如果您已注册 AWS 账户，可以通过登录 AWS 管理控制台 并从 AWS 管理控制台 主页中选择 S3 来访问 S3 on Outposts。然后，从左侧导航窗格中选择 Outposts buckets（Outposts 存储桶）。

AWS Command Line Interface

可以使用 AWS 命令行工具，在系统的命令行中发出命令来执行 AWS （包括 S3）任务。

AWS Command Line Interface (AWS CLI) 针对大量 AWS 服务 提供了相关命令。AWS CLI 在 Windows、macOS 和 Linux 上受支持。要开始使用，请参阅《AWS Command Line Interface 用户指南》。有关您可用于 S3 on Outposts 的命令的更多信息，请参阅《AWS CLI 命令参考》中的 s3api、s3control 和 s3outposts。

AWS 软件开发工具包

AWS 提供的 SDK（软件开发工具包）包含各种编程语言和平台（Java、Python、Ruby、.NET、iOS、Android 等）的库和示例代码。AWS SDK 提供便捷的方式来创建对 S3 on Outposts 和 AWS 的编程访问。由于 S3 on Outposts 与 Amazon S3 使用相同的 SDK，因此 S3 on Outposts 使用相同的 S3 API、自动化和工具提供一致的体验。

S3 on Outposts 是一项 REST 服务。您可以使用 AWS SDK 库向 S3 on Outposts 发送请求，这些库包装底层 REST API 并简化编程任务。例如，SDK 负责计算签名、加密签名请求、管理错误和自动重试请求等任务。有关 AWS SDK 的信息（包括如何下载及安装），请参阅在 AWS 上构建所用的工具。

支付 S3 on Outposts 的费用

您可以购买各种 AWS Outposts 机架配置，这些配置具有 Amazon EC2 实例类型、Amazon EBS 通用固态硬盘 (SSD) 卷 (gp2) 和 S3 on Outposts 的组合。定价包括交付、安装、基础设施服务维护以及软件修补程序和升级。

有关更多信息，请参阅 AWS Outposts 机架定价。

后续步骤

有关使用 S3 on Outposts 的更多信息，请参阅以下主题：

- 设置 Outpost
- Amazon S3 on Outposts 与 Amazon S3 有何不同？
- 开始使用 Amazon S3 on Outposts
- S3 on Outposts 的网络
- 使用 S3 on Outposts 存储桶
- 使用 S3 on Outposts 对象
- S3 on Outposts 中的安全性
- 管理 S3 on Outposts 存储

- 使用 Amazon S3 on Outposts 进行开发

设置 Outpost

要开始使用 Amazon S3 on Outposts，您需要在您的设施中部署具有 Amazon S3 容量的 Outpost。有关订购 Outpost 和 S3 容量的选项的信息，请参阅 AWS Outposts。要检查 Outposts 在其上是否具有 S3 容量，可以使用 ListOutpostsWithS3 API 调用。有关规格以及要了解 S3 on Outposts 与 Amazon S3 的不同之处，请参阅 Amazon S3 on Outposts 与 Amazon S3 有何不同？

有关更多信息，请参阅以下主题。

主题

订购新 Outpost

如果您需要订购具有 S3 容量的新 Outpost，请参阅 AWS Outposts 机架定价以了解适用于 Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Elastic Block Store (Amazon EBS) 和 Amazon S3 的容量选项。

选择配置后，按照《AWS Outposts 用户指南》中创建 Outpost 和订购 Outpost 容量中的步骤操作。

Amazon S3 on Outposts 与 Amazon S3 有何不同？

Amazon S3 on Outposts 向您的本地 AWS Outposts 环境提供对象存储。使用 S3 on Outposts，可通过将数据保持在本地应用程序附近，帮助您满足本地处理、数据驻留和苛刻的性能需求。由于使用 Amazon S3 API 和功能，S3 on Outposts 可以轻松存储、保护、标记、报告和控制对 Outposts 数据的访问，并将 AWS 基础设施扩展到您的本地设施，以获得一致的混合体验。

有关 S3 on Outposts 的独特之处的更多信息，请参阅以下主题。

主题

S3 on Outposts 规范

- Outposts 存储桶的最大大小为 50 TB。
- Outposts 存储桶中对象的最大大小为 5 TB。
- 每个 AWS 账户 的最大 Outpost 存储桶数为 100。
- Outpost 存储桶只能使用访问点和终端节点进行访问。
- 每个 Outposts 存储桶的最大访问点数为 10。
- 访问点策略的大小限制为 20 KB。
- Outpost 拥有者可以使用 AWS Resource Access Manager 在 AWS Organizations 中管理您组织内的访问权限。需要访问 Outpost 的所有账户必须与 AWS Organizations 中的所有者账户位于同一组织内。
- S3 on Outposts 存储桶拥有者账户始终是存储桶中所有对象的所有者。
- 只有 S3 on Outposts 存储桶拥有者账户可以对存储桶执行操作。
- 对象大小限制与 Amazon S3 一致。
- 在 S3 on Outposts 上存储的所有对象都存储在 OUTPOSTS 存储类中。

- 原定设置情况下，存储在 OUTPOSTS 存储类中的所有对象都使用具有 Amazon S3 托管式加密密钥 (SSE-S3) 的服务器端加密进行存储。您也可以明确选择使用具有客户提供的加密密钥 (SSE-C) 的服务器端加密来存储对象。
- 如果没有足够的空间在您的 Outpost 上存储对象，API 会返回容量不足异常 (ICE)。

S3 on Outposts 支持的 API 操作

有关 S3 on Outposts 支持的 API 操作的列表，请参阅Amazon S3 on Outposts API 操作。

S3 on Outposts 支持的 Amazon S3 AWS CLI 命令

Amazon S3 on Outposts 目前支持以下 Amazon S3 AWS CLI 命令。有关更多信息，请参阅《AWS CLI Command Reference》中的 Available Commands。

- `cp`、`mv` 和 `sync`，在同一个 Outposts 存储桶中，或者在本地环境与 Outposts 存储桶之间。
- `ls`
- `presign`
- `rm`

S3 on Outposts 不支持的 Amazon S3 功能

Amazon S3 on Outposts 目前不支持以下 Amazon S3 功能。想要使用它们的任何尝试都将遭到拒绝。

- 有条件请求
- 访问控制列表 (ACL)
- 跨源资源共享 (CORS)
- S3 批量操作
- S3 清单报告
- 更改默认存储桶加密
- 公有存储桶
- 多重验证 (MFA) 删除
- S3 生命周期转换（除了对象删除和停止未完成的分段上传以外）
- S3 对象锁定依法保留

- 对象锁定保留
- 具有 AWS Key Management Service（AWS KMS）密钥的服务器端加密（SSE-KMS）
- S3 复制时间控制 (S3 RTC)
- Amazon CloudWatch 请求指标
- 指标配置
- Transfer Acceleration
- S3 事件通知
- 申请方付款存储桶
- S3 Select
- AWS Lambda 事件
- Server access logging (服务器访问日志记录)
- HTTP POST 请求
- SOAP
- 网站访问

S3 on Outposts 网络要求

- 要将请求路由到 S3 on Outposts 访问点，您必须创建 S3 on Outposts 终端节点并进行配置。以下限制适用于 S3 on Outposts 的终端节点：
  - Outpost 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的终端节点，每个 Outpost 最多可以有 100 个终端节点。
  - 可以将多个访问点映射到同一终端节点。
  - 只能将终端节点添加到其 CIDR 块在以下 CIDR 范围的子空间内的 VPC 中：
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
- 只能从具有不重叠 CIDR 块的 VPC 中创建到 Outpost 的终端节点。
- 只能从对应的 Outposts 子网内创建终端节点。
- 用于创建终端节点的子网必须包含四个 IP 地址，以供 S3 on Outposts 使用。
- 如果您指定客户拥有的 IP 地址池（CoIP 池），则它必须包含四个 IP 地址，以供 S3 on Outposts 使用。

- 每个 VPC 每个 Outpost 只能创建一个终端节点。

开始使用 Amazon S3 on Outposts

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。

通过 Amazon S3 on Outposts，您可以像在 Amazon S3 上一样在 AWS Outposts 上使用 Amazon S3 API 和功能，如对象存储、访问策略、加密和标记。有关 S3 on Outposts 的信息，请参阅什么是 Amazon S3 on Outposts？

主题

- 通过 AWS 管理控制台开始使用
- 通过 AWS CLI 和适用于 Java 的 SDK 开始使用

通过 AWS 管理控制台开始使用

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

要通过控制台开始使用 S3 on Outposts，请参阅以下主题。要通过 AWS CLI 或适用于 Java 的 AWS SDK 开始使用，请参阅通过 AWS CLI 和适用于 Java 的 SDK 开始使用。

主题

- 创建存储桶、访问点和端点
- 后续步骤

创建存储桶、访问点和端点

以下过程介绍了如何在 S3 on Outposts 中创建第一个存储桶。使用控制台创建存储桶时，您还可以创建一个访问点以及与该存储桶关联的端点，以便您可以立即开始在存储桶中存储对象。

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择创建 Outposts 存储桶。
4. 对于 Bucket name（存储桶名称）中，输入符合域名系统 (DNS) 标准的存储桶名称。

   存储桶名称必须满足以下要求：

   - 在 AWS 账户、Outpost 以及 Outpost 所属的 AWS 区域内保持唯一性。
   - 长度为 3-63 个字符。
   - 不包含大写字符。
   - 以小写字母或数字开头。

   创建存储桶后，便无法再更改其名称。有关命名存储桶的信息，请参阅《Amazon S3 用户指南》中的通用存储桶命名规则。

   > **Important**
   >
   > 避免在存储桶名称中包含敏感信息，如账号。存储桶名称会显示在指向存储桶中的对象的 URL 中。

5. 对于 Outpost，选择要放置存储桶的 Outpost。
6. 在 Bucket Versioning（桶版本控制）下，将 S3 on Outposts 桶的 S3 版本控制状态设置为以下选项之一：

   - Disable（禁用）（原定设置）- 桶保持未版本控制状态。
   - Enable（启用）- 为桶中的对象启用 S3 版本控制。添加到桶的所有对象都将收到唯一的版本 ID。

   有关 S3 版本控制的更多信息，请参阅为 S3 on Outposts 桶管理 S3 版本控制。
7. （可选）添加要与 Outposts 存储桶关联的任何 optional tags（可选标签）。您可以使用标签来跟踪单个项目或项目组的标准，或使用成本分配标签标注您的存储桶。

原定设置情况下，存储在 Outposts 存储桶中的所有对象都使用具有 Amazon S3 托管式加密密钥 (SSE-S3) 的服务器端加密进行存储。您也可以明确选择使用具有客户提供的加密密钥 (SSE-C) 的服务器端加密来存储对象。要更改加密类型，您必须使用 REST API、AWS Command Line Interface (AWS CLI) 或 AWS 软件开发工具包。

8. 在 Outposts access point settings（Outposts 访问点设置）部分，输入访问点名称。

   S3 on Outposts 访问点可简化对 S3 on Outposts 中的共享数据集的大规模数据访问管理。访问点是附加到 Outposts 存储桶的已命名网络端点，您可以使用这些存储桶执行 S3 对象操作。有关更多信息，请参阅 接入点。

   在此区域和 Outpost 的账户中，接入点名称必须唯一，并符合接入点限制和局限性。

9. 为此 Amazon S3 on Outposts 访问点选择 VPC。

   如果您没有 VPC，请选择 Create VPC（创建 VPC）。有关更多信息，请参阅《Amazon S3 用户指南》中的创建限制到虚拟私有云（VPC）的接入点。

   通过 Virtual Private Cloud (VPC)，您可以将 AWS 资源启动到您定义的虚拟网络中。这个虚拟网络与您在自己的数据中心中运行的传统网络极其相似，并会为您提供使用的可扩展基础设施的优势 AWS

10. （对于现有 VPC 为可选）为端点选择 Endpoint subnet（端点子网）。

    子网是您的 VPC 内的 IP 地址范围。如果您没有所需的子网，请选择 Create subnet（创建子网）。有关更多信息，请参阅 S3 on Outposts 的网络。

11. （对于现有 VPC 为可选）为端点选择 Endpoint security group（端点安全组）。

    安全组充当 VPC 的虚拟防火墙，以控制入站和出站流量。

12. （对于现有 VPC 为可选）选择 Endpoint access type（端点访问类型）：

    - Private（私密）– 要与 VPC 一起使用。
    - Customer owned IP（客户拥有的 IP）– 要与本地部署网络中的客户拥有的 IP 池一起使用。

13. （可选）指定 Outpost access point policy（Outpost 访问点策略）。控制台会自动显示访问点的 Amazon Resource Name (ARN)，您可以在策略中使用该名称。

14. 选择创建 Outposts 存储桶。

> **Note**
>
> 创建 Outpost 端点并准备好桶以供使用，最多可能需要 5 分钟的时间。要配置其他存储桶设置，请选择 View details（查看详细信息）。

后续步骤

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

创建 S3 on Outposts 存储桶、访问点和端点之后，您可以使用 AWS CLI 或适用于 Java 的 SDK，将对象上传到存储桶。有关更多信息，请参阅 将对象上传到 S3 on Outposts 存储桶。

通过 AWS CLI 和适用于 Java 的 SDK 开始使用

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅 什么是 Amazon S3 on Outposts？

要开始使用 S3 on Outposts，您必须创建存储桶、访问点和端点。然后，您可以将对象上传到存储桶。以下示例显示如何通过 AWS CLI 和适用于 Java 的 SDK 开始使用 S3 on Outposts。要通过控制台开始使用，请参阅通过 AWS 管理控制台开始使用。

主题

- 步骤 1：创建存储桶
- 步骤 2：创建访问点
- 步骤 3：创建端点

步骤 1：创建存储桶

以下 AWS CLI 和适用于 Java 的 SDK 示例显示如何创建 S3 on Outposts 存储桶。

AWS CLI

Example

以下示例使用 AWS CLI 创建 S3 on Outposts 存储桶 (`s3-outposts:CreateBucket`)。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

SDK for Java

Example

有关如何使用适用于 Java 的 AWS SDK 创建 S3 Outposts 存储桶的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsBucket.java。

步骤 2：创建访问点

要访问 Amazon S3 on Outposts 存储桶，您必须创建和配置访问点。这些示例显示如何使用 AWS CLI 和适用于 Java 的 SDK 创建访问点。

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

AWS CLI

Example

以下 AWS CLI 示例为 Outposts 存储桶创建访问点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

SDK for Java

Example

有关如何使用适用于 Java 的 AWS SDK 为 S3 Outposts 存储桶创建接入点的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsAccessPoint.java。

步骤 3：创建端点

要将请求路由到 Amazon S3 on Outposts 访问点，您必须创建 S3 on Outposts 端点并进行配置。为了创建端点，您需要使用服务链接建立到 Outposts 主区域的活跃连接。Outposts 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的端点。有关端点配额的更多信息，请参阅S3 on Outposts 网络要求。您必须创建一个端点，才能访问 Outposts 桶并执行对象操作。有关更多信息，请参阅 端点。

这些示例显示如何使用 AWS CLI 和适用于 Java 的 SDK 创建端点。有关创建和管理端点所需的权限的更多信息，请参阅 S3 on Outposts 端点的权限。

AWS CLI

Example

以下 AWS CLI 示例使用 VPC 资源访问类型，为 Outpost 创建端点。VPC 派生自子网。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

以下 AWS CLI 示例使用客户拥有的 IP 地址池（CoIP 池）访问类型为 Outpost 创建端点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

SDK for Java

Example

有关如何使用适用于 Java 的 AWS SDK 为 S3 Outpost 创建端点的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsEndPoint.java。

步骤 4：将对象上传到 S3 on Outposts 存储桶

要上传对象，请参阅将对象上传到 S3 on Outposts 存储桶。

S3 on Outposts 的网络

您可以使用 Amazon S3 on Outposts 为需要本地数据访问、数据处理和数据驻留的应用程序存储和检索本地对象。本节介绍了访问 S3 on Outposts 的网络要求。

主题

选择网络访问类型

您可以从 VPC 内部或本地网络访问 S3 on Outposts。您通过使用访问点和终端节点连接与 Outposts 存储桶进行通信。此连接将在 AWS 网络内的 VPC 和 S3 on Outposts 存储桶之间保持流量。创建终端节点时，您必须将终端节点访问类型指定为 `Private`（对于 VPC 路由）或 `CustomerOwnedIp`（对于客户拥有的 IP 地址池 [CoIP 池]）。

- `Private`（对于 VPC 路由）– 如果您不指定访问类型，S3 on Outposts 将默认使用 `Private`。使用 `Private` 访问类型，VPC 中的实例无需公有 IP 地址便可与 Outpost 中的资源进行通信。您可以从 VPC 内使用 S3 on Outposts。这种类型的端点可以通过直接 VPC 路由从本地网络访问。有关更多信息，请参阅《AWS Outposts 用户指南》中的本地网关路由表。
- `CustomerOwnedIp`（对于 CoIP 池）– 如果您不默认使用 `Private` 访问类型并选择 `CustomerOwnedIp`，则必须指定 IP 地址范围。您可以通过此访问类型从本地网络和 VPC 内使用 S3 on Outposts。访问 VPC 内 Outposts 的 S3 时，您的本地网关的带宽将限制流量。

访问 S3 on Outposts 存储桶和对象

要访问 S3 on Outposts 存储桶和对象，您必须满足以下条件：

- VPC 的访问点
- 同一 VPC 的终端节点
- 您的 Outpost 与 AWS 区域之间的主动连接。有关如何将您的 Outpost 连接到区域的更多信息，请参阅 AWSOutposts 用户指南 中的复制到 Outpost AWS 区域。

有关访问 S3 on Outposts 中的存储桶和对象的更多信息，请参阅使用 S3 on Outposts 存储桶和使用 S3 on Outposts 对象。

跨账户弹性网络接口

S3 on Outposts 终端节点是具有适当 Amazon Resource Name (ARN) 的命名资源。创建这些终端节点后，AWS Outposts 会设置多个跨账户弹性网络接口。S3 on Outposts 跨账户弹性网络接口与其他网络接口相似，但有一个例外：S3 on Outposts 将跨账户弹性网络接口与 Amazon EC2 实例相关联。

S3 on Outposts 域名系统（DNS）通过跨账户弹性网络接口对请求进行负载平衡。S3 on Outposts 在您的AWS账户中创建跨帐户弹性网络接口，它可从 Amazon EC2 控制台的 Network interfaces (网络接口) 窗格中查看。

对于使用 CoIP 池访问类型的终端节点，S3 on Outposts 将从配置的 CoIP 池中分配 IP 地址并将其与跨账户弹性网络接口相关联。

使用 S3 on Outposts 存储桶

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 存储桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您可以像在 Amazon S3 中一样在 Outpost 存储桶上使用相同的 API 和功能，包括访问策略、加密和标记。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

您通过 Virtual Private Cloud (VPC) 使用访问点和终端节点连接与 Outpost 存储桶进行通信。要访问 S3 on Outposts 存储桶和对象，您必须具有 VPC 的访问点和同一 VPC 的终端节点。有关更多信息，请参阅 S3 on Outposts 的网络。

桶

在 S3 on Outposts 中，存储桶名称对于 Outposts 是唯一的，并且需要 Outpost 所属区域的 AWS 区域代码，AWS 账户 ID、Outpost ID 和用于识别它们的存储桶名称。

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

有关更多信息，请参阅 S3 on Outposts 的资源 ARN。

接入点

Amazon S3 on Outposts 支持将纯 Virtual Private Cloud (VPC) 接入点作为访问 Outposts 存储桶的唯一方式。

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

以下示例显示用于 S3 on Outposts 访问点的 ARN 格式。访问点 ARN 包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称。

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

端点

要将请求路由到 S3 on Outposts 访问点，您必须创建 S3 on Outposts 终端节点并进行配置。借助 S3 on Outposts 终端节点，您可以将 VPC 私密连接到您的 Outpost 存储桶。S3 on Outposts 终端节点是指向 S3 on Outposts 存储桶的入口点的虚拟统一资源标识符（URI）。它们是水平扩展、冗余和高度可用的 VPC 组件。

Outpost 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的终端节点，每个 Outpost 最多可以有 100 个终端节点。您必须创建这些终端节点，才能访问 Outpost 存储桶并执行对象操作。创建这些终端节点还通过允许相同的操作在 S3 和 S3 on Outposts 中工作，使 API 模型和行为相同。

S3 on Outposts 上的 API 操作

要管理 Outpost 存储桶 API 操作，S3 on Outposts 将托管一个与 Amazon S3 终端节点不同的单独终端节点。这个终端节点是 `s3-outposts.region.amazonaws.com`。

要使用 Amazon S3 API 操作，您必须使用正确的 ARN 格式，对存储桶和对象进行签署。您必须向 API 操作传递 ARN，以便 Amazon S3 能够确定请求是针对 Amazon S3 (`s3-control.region.amazonaws.com`) 还是针对 S3 on Outposts (`s3-outposts.region.amazonaws.com`)。根据 ARN 格式，然后 S3 可以适当地签署和路由请求。

无论何时将请求发送到 Amazon S3 控制平面，此 SDK 都会从 ARN 中提取组件，并包含一个附加标头 `x-amz-outpost-id`，其 *outpost-id* 值提取自 ARN。ARN 中的服务名称将用于在将请求路由到 S3 on Outposts 终端节点之前对请求进行签名。此行为适用于所有由 `s3control` 客户端处理的所有 API 操作。

下表列出了用于 Amazon S3 on Outposts 的扩展 API 操作及其相对于 Amazon S3 的更改。

API	S3 on Outposts 参数值	
`CreateBucket`	存储桶名称为 ARN，Outpost ID	
`ListRegionalBuckets`	Outpost ID	
`DeleteBucket`	存储桶名称为 ARN	
`DeleteBucketLifecycleConfiguration`	存储桶名称为 ARN	

API	S3 on Outposts 参数值	
GetBucketLifecycle Configuration	存储桶名称为 ARN	
PutBucketLifecycle Configuration	存储桶名称为 ARN	
GetBucketPolicy	存储桶名称为 ARN	
PutBucketPolicy	存储桶名称为 ARN	
DeleteBucketPolicy	存储桶名称为 ARN	
GetBucketTagging	存储桶名称为 ARN	
PutBucketTagging	存储桶名称为 ARN	
DeleteBucketTagging	存储桶名称为 ARN	
CreateAccessPoint	访问点名称为 ARN	
DeleteAccessPoint	访问点名称为 ARN	
GetAccessPoint	访问点名称为 ARN	
GetAccessPoint	访问点名称为 ARN	
ListAccessPoints	访问点名称为 ARN	
PutAccessPointPolicy	访问点名称为 ARN	
GetAccessPointPolicy	访问点名称为 ARN	
DeleteAccessPointPolicy	访问点名称为 ARN	

创建和管理 S3 on Outposts 存储桶

有关创建和管理 S3 on Outposts 存储桶的更多信息，请参阅以下主题。

创建 S3 on Outposts 存储桶

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅 什么是 Amazon S3 on Outposts？

> **Note**
>
> 创建存储桶的 AWS 账户 拥有该存储桶，也是唯一可以向其提交操作的账户。存储桶具有配置属性，如 Outpost、标签、默认加密和访问点设置。访问点设置包括用于访问存储桶中对象的 Virtual Private Cloud (VPC) 和访问点策略以及其他元数据。有关更多信息，请参阅 S3 on Outposts 规范。
>
> 如果您想创建一个桶，该桶使用 AWS PrivateLink 通过虚拟私有云（VPC）中的接口 VPC 端点提供桶和端点管理访问，请参阅适用于 S3 on Outposts 的 AWS PrivateLink。

以下示例显示如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 创建 S3 on Outposts 存储桶。

使用 S3 控制台

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择创建 Outposts 存储桶。
4. 对于 Bucket name（存储桶名称）中，输入符合域名系统 (DNS) 标准的存储桶名称。

   存储桶名称必须满足以下要求：

   - 在 AWS 账户、Outpost 以及 Outpost 所属的 AWS 区域内保持唯一性。
   - 长度为 3-63 个字符。
   - 不包含大写字符。

- 以小写字母或数字开头。

  创建存储桶后，便无法再更改其名称。有关命名存储桶的信息，请参阅《Amazon S3 用户指南》中的通用存储桶命名规则。

  > **Important**
  >
  > 避免在存储桶名称中包含敏感信息，如账号。存储桶名称会显示在指向存储桶中的对象的 URL 中。

5. 对于 Outpost，选择要放置存储桶的 Outpost。
6. 在 Bucket Versioning（桶版本控制）下，将 S3 on Outposts 桶的 S3 版本控制状态设置为以下选项之一：

   - Disable（禁用）（原定设置）- 桶保持未版本控制状态。
   - Enable（启用）- 为桶中的对象启用 S3 版本控制。添加到桶的所有对象都将收到唯一的版本 ID。

   有关 S3 版本控制的更多信息，请参阅为 S3 on Outposts 桶管理 S3 版本控制。
7. （可选）添加要与 Outposts 存储桶关联的任何 optional tags（可选标签）。您可以使用标签来跟踪单个项目或项目组的标准，或使用成本分配标签标注您的存储桶。

   原定设置情况下，存储在 Outposts 存储桶中的所有对象都使用具有 Amazon S3 托管式加密密钥 (SSE-S3) 的服务器端加密进行存储。您也可以明确选择使用具有客户提供的加密密钥 (SSE-C) 的服务器端加密来存储对象。要更改加密类型，您必须使用 REST API、AWS Command Line Interface (AWS CLI) 或 AWS 软件开发工具包。
8. 在 Outposts access point settings（Outposts 访问点设置）部分，输入访问点名称。

   S3 on Outposts 访问点可简化对 S3 on Outposts 中的共享数据集的大规模数据访问管理。访问点是附加到 Outposts 存储桶的已命名网络端点，您可以使用这些存储桶执行 S3 对象操作。有关更多信息，请参阅 接入点。

   在此区域和 Outpost 的账户中，接入点名称必须唯一，并符合接入点限制和局限性。
9. 为此 Amazon S3 on Outposts 访问点选择 VPC。

   如果您没有 VPC，请选择 Create VPC（创建 VPC）。有关更多信息，请参阅《Amazon S3 用户指南》中的创建限制到虚拟私有云（VPC）的接入点。

通过 Virtual Private Cloud (VPC)，您可以将 AWS 资源启动到您定义的虚拟网络中。这个虚拟网络与您在自己的数据中心中运行的传统网络极其相似，并会为您提供使用的可扩展基础设施的优势 AWS

10. （对于现有 VPC 为可选）为端点选择 Endpoint subnet（端点子网）。

    子网是您的 VPC 内的 IP 地址范围。如果您没有所需的子网，请选择 Create subnet（创建子网）。有关更多信息，请参阅 S3 on Outposts 的网络。

11. （对于现有 VPC 为可选）为端点选择 Endpoint security group（端点安全组）。

    安全组充当 VPC 的虚拟防火墙，以控制入站和出站流量。

12. （对于现有 VPC 为可选）选择 Endpoint access type（端点访问类型）：

    - Private（私密）– 要与 VPC 一起使用。
    - Customer owned IP（客户拥有的 IP）– 要与本地部署网络中的客户拥有的 IP 池一起使用。

13. （可选）指定 Outpost access point policy（Outpost 访问点策略）。控制台会自动显示访问点的 Amazon Resource Name (ARN)，您可以在策略中使用该名称。

14. 选择创建 Outposts 存储桶。

> **Note**
>
> 创建 Outpost 端点并准备好桶以供使用，最多可能需要 5 分钟的时间。要配置其他存储桶设置，请选择 View details（查看详细信息）。

使用 AWS CLI

Example

以下示例使用 AWS CLI 创建 S3 on Outposts 存储桶 (`s3-outposts:CreateBucket`)。要运行此命令，请将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

使用适用于 Java 的 AWS 软件开发工具包

Example

有关如何使用适用于 Java 的 AWS SDK 创建 S3 Outposts 存储桶的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsBucket.java。

为 S3 on Outposts 存储桶添加标签

您可以为 Amazon S3 on Outposts 存储桶添加标签，以跟踪各个项目或项目组的存储成本和其他标准。

> **Note**
>
> 创建存储桶的 AWS 账户拥有该存储桶，也是唯一可以更改其标签的账户。

使用 S3 控制台

1. 登录到AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择您要编辑其标签的 Outposts 存储桶。
4. 选择属性选项卡。
5. 在 Tags（标签）下，选择 Edit（编辑）。
6. 选择 Add new tag（添加新标签），然后输入 Key（键）和可选的 Value（值）。

   添加您想要与 Outposts 存储桶关联的任何标签，以跟踪单个项目或项目组的其他标准。
7. 选择 Save changes（保存更改）。

使用 AWS CLI

以下 AWS CLI 示例使用当前文件夹中指定标签的 JSON 文档 (*tagging.json*) 将标记配置应用于 S3 on Outposts 存储桶。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --tagging file://tagging.json
```

tagging.json

```
{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

以下 AWS CLI 示例直接从命令行将标签配置应用于 S3 on Outposts 存储桶。

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --tagging 'TagSet=[{Key=organization,Value=marketing}]'
```

有关此命令的更多信息，请参阅《AWS CLI 参考》中的 put-bucket-tagging。

使用存储桶策略管理对 Amazon S3 on Outposts 存储桶的访问

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。有关更多信息，请参阅存储桶策略。

您可以更新存储桶策略来管理对 Amazon S3 on Outposts 存储桶的访问。有关更多信息，请参阅以下主题。

主题

- 为 Amazon S3 on Outposts 存储桶添加或编辑存储桶策略
- 查看 Amazon S3 on Outposts 存储桶策略
- 删除 Amazon S3 on Outposts 存储桶策略
- 存储桶策略示例

为 Amazon S3 on Outposts 存储桶添加或编辑存储桶策略

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。有关更多信息，请参阅存储桶策略。

以下主题显示如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 或 适用于 Java 的 AWS SDK 更新 Amazon S3 on Outposts 存储桶策略。

使用 S3 控制台

创建或编辑存储桶策略

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择您要编辑其权限的 Outposts 存储桶。
4. 选择权限选项卡。
5. 在 Outposts bucket policy（Outposts 存储桶策略）部分，要创建或编辑新策略，请选择 Edit（编辑）。

   现在，您可以添加或编辑 S3 on Outposts 存储桶策略。有关更多信息，请参阅 使用 S3 on Outposts 设置 IAM。

使用 AWS CLI

以下 AWS CLI 示例在 Outpost 存储桶上放置策略。

1. 将以下存储桶策略保存到 JSON 文件中。在本示例中，文件命名为 `policy1.json`。将 *`user input placeholders`* 替换为您自己的信息。

   JSON

```
{
   "Version":"2012-10-17",
   "Id":"testBucketPolicy",
   "Statement":[
```

```
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:root"
         },
         "Action":[
            "s3-outposts:GetObject",
            "s3-outposts:PutObject",
            "s3-outposts:DeleteObject",
            "s3-outposts:ListBucket"
         ],
         "Resource":"arn:aws:s3-outposts:us-
east-1:123456789012:outpost/op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket"
      }
   ]
}
```

2. 将 JSON 文件作为 put-bucket-policy CLI 命令的一部分提交。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control put-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --policy file://policy1.json
```

使用适用于 Java 的 AWS 软件开发工具包

以下 SDK for Java 示例在 Outpost 存储桶上放置策略。

```
import com.amazonaws.services.s3control.model.*;

public void putBucketPolicy(String bucketArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testBucketPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId+ "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + bucketArn + "\"}]}";

    PutBucketPolicyRequest reqPutBucketPolicy = new PutBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withPolicy(policy);
```

```
    PutBucketPolicyResult respPutBucketPolicy =
 s3ControlClient.putBucketPolicy(reqPutBucketPolicy);
    System.out.printf("PutBucketPolicy Response: %s%n",
 respPutBucketPolicy.toString());

}
```

查看 Amazon S3 on Outposts 存储桶策略

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。有关更多信息，请参阅存储桶策略。

以下主题说明如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 或 适用于 Java 的 AWS SDK 查看 Amazon S3 on Outposts 存储桶策略。

使用 S3 控制台

创建或编辑存储桶策略

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择您要编辑其权限的 Outposts 存储桶。
4. 选择 Permissions 选项卡。
5. 在 Outposts bucket policy（Outposts 存储桶策略）部分，您可以查看现有的存储桶策略。有关更多信息，请参阅 使用 S3 on Outposts 设置 IAM。

使用 AWS CLI

以下 AWS CLI 示例获取 Outposts 存储桶的策略。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control get-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

使用适用于 Java 的 AWS 软件开发工具包

以下适用于 Java 的开发工具包示例为 Outpost 存储桶获取策略。

```
import com.amazonaws.services.s3control.model.*;

public void getBucketPolicy(String bucketArn) {

    GetBucketPolicyRequest reqGetBucketPolicy = new GetBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketPolicyResult respGetBucketPolicy =
 s3ControlClient.getBucketPolicy(reqGetBucketPolicy);
    System.out.printf("GetBucketPolicy Response: %s%n",
 respGetBucketPolicy.toString());

}
```

删除 Amazon S3 on Outposts 存储桶策略

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。有关更多信息，请参阅存储桶策略。

以下主题显示如何使用 AWS 管理控制台或 AWS Command Line Interface (AWS CLI) 查看 Amazon S3 on Outposts 存储桶策略。

使用 S3 控制台

删除存储桶策略

1. 通过以下网址打开 Simple Storage Service（Amazon S3）控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择您要编辑其权限的 Outposts 存储桶。
4. 选择 Permissions 选项卡。
5. 在 Outposts bucket policy（Outposts 存储桶策略）部分中，选择 Delete（删除）。
6. 确认删除操作。

使用 AWS CLI

以下示例使用 AWS CLI 删除 S3 on Outposts 存储桶策略 (`s3-outposts:DeleteBucket`)。要运行此命令，请将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control delete-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

存储桶策略示例

使用 S3 on Outposts 桶策略，您可以保护对 S3 on Outposts 桶中对象的访问，这样，只有具有适当权限的用户才能访问它们。您甚至可以阻止没有适当权限的经过身份验证的用户访问您的 S3 on Outposts 资源。

本节介绍关于 S3 on Outposts 桶策略的典型使用案例的示例。要测试这些策略，您需要将 *`user input placeholders`* 替换为您自己的信息（例如存储桶名称）。

要授予或拒绝对一组对象的权限，可以在 Amazon资源名称（ARN）和其他值中使用通配符（`*`）。例如，您可以控制对以通用前缀或以给定扩展名结尾的对象组的访问，例如 `.html`。

有关 AWS Identity and Access Management（IAM）策略语言的更多信息，请参阅使用 S3 on Outposts 设置 IAM。

> **Note**
>
> 使用 Amazon S3 控制台测试 s3outposts 权限时，您必须授予控制台所需的其它权限，如 `s3outposts:createendpoint`、`s3outposts:listendpoints` 等等。

用于创建存储桶策略的其他资源

- 有关在创建 S3 on Outposts 桶策略时可以使用的 IAM policy 操作、资源和条件键的列表，请参阅 Actions, resources, and condition keys for Amazon S3 on Outposts。
- 有关创建 S3 on Outposts 策略的指南，请参阅为 Amazon S3 on Outposts 存储桶添加或编辑存储桶策略。

主题

- 基于特定 IP 地址管理对 Amazon S3 on Outposts 桶的访问权限

基于特定 IP 地址管理对 Amazon S3 on Outposts 桶的访问权限

存储桶策略是基于资源的 AWS Identity and Access Management (IAM) 策略，您可以使用该策略向存储桶及其中对象授予访问权限。只有存储桶拥有者才能将策略与存储桶关联。附加到存储桶的权限适用于存储桶拥有者拥有的存储桶中所有对象。存储桶策略的大小限制为 20 KB。有关更多信息，请参阅存储桶策略。

限制特定 IP 地址的访问权限

以下示例拒绝所有用户对指定桶中的对象执行任何 S3 on Outposts 操作，除非请求来自指定的 IP 地址范围。

> **Note**
>
> 在限制对特定 IP 地址的访问权限时，务必还要指定哪些 VPC 端点、VPC 源 IP 地址或外部 IP 地址可以访问 S3 on Outposts 桶。否则，如果您的策略拒绝所有用户在没有适当权限的情况下对 S3 on Outposts 桶中的对象执行任何 s3outposts 操作，则您可能会失去对该桶的访问权限。

此策略的 `Condition` 语句确定允许的 IP 版本 4（IPv4）IP 地址范围为 *`192.0.2.0/24`*。

`Condition` 块使用 `NotIpAddress` 条件和 `aws:SourceIp` 条件键（这是 AWS 范围的条件键）。`aws:SourceIp` 条件键只能用于公有 IP 地址范围。有关条件键的更多信息，请参阅 Actions, resources, and condition keys for S3 on Outposts。`aws:SourceIp` IPv4 值使用标准 CIDR 表示法。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素参考。

> **Warning**
>
> 在使用此 S3 on Outposts 策略之前，将此示例中的 *`192.0.2.0/24`* IP 地址范围替换为适合您的使用案例的值。否则，您将失去访问桶的能力。

```
{
    "Version": "2012-10-17",
    "Id": "S3OutpostsPolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
```

```
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3-outposts:*",
            "Resource": [
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
accesspoint/EXAMPLE-ACCESS-POINT-NAME",
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/amzn-s3-demo-bucket"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

同时允许 IPv4 和 IPv6 地址

在您开始使用 IPv6 地址时，建议您使用您的 IPv6 地址范围以及现有的 IPv4 范围来更新组织的所有策略。这样做将有助于确保这些策略在您转换到 IPv6 时继续有效。

以下 S3 on Outposts 示例桶策略说明如何混用 IPv4 和 IPv6 地址范围来覆盖组织的所有有效 IP 地址。该示例策略将允许对示例 IP 地址 *192.0.2.1* 和 *2001:DB8:1234:5678::1* 的访问，而拒绝对地址 *203.0.113.1* 和 *2001:DB8:1234:5678:ABCD::1* 的访问。

`aws:SourceIp` 条件键只能用于公有 IP 地址范围。`aws:SourceIp` 的 IPv6 值必须采用标准的 CIDR 格式。对于 IPv6，我们支持使用 `::` 表示 0 范围（例如，`2001:DB8:1234:5678::/64`）。有关更多信息，请参阅《IAM 用户指南》中的 IP 地址条件运算符。

> **Warning**
>
> 在使用此 S3 on Outposts 策略之前，将此示例中的 IP 地址范围替换为适合您的使用案例的值。否则，您可能会失去访问存储桶的能力。

JSON

```
{
    "Id": "S3OutpostsPolicyId2",
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": [
                "s3-outposts:GetObject",
                "s3-outposts:PutObject",
                "s3-outposts:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket",
                "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket/*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678:ABCD::/80"
                    ]
                }
            }
        }
    ]
}
```

列出 Amazon S3 on Outposts 存储桶

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个

设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

有关使用 S3 on Outposts 中的存储桶的更多信息，请参阅使用 S3 on Outposts 存储桶。

以下示例介绍了如何使用 AWS 管理控制台、AWS CLI 和 适用于 Java 的 AWS SDK 返回 S3 on Outposts 存储桶的列表。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 在 Outposts buckets（Outposts 存储桶）下，查看 S3 on Outposts 存储桶的列表。

使用 AWS CLI

以下 AWS CLI 示例获取 Outpost 中的存储桶列表。要使用此命令，请将每个 *user input placeholder* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 list-regional-buckets。

```
aws s3control list-regional-buckets --account-id 123456789012 --outpost-
id op-01ac5d28a6a232904
```

使用适用于 Java 的 AWS SDK

以下适用于 Java 的 SDK 示例获取 Outpost 中的存储桶列表。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 ListRegionalBuckets。

```
import com.amazonaws.services.s3control.model.*;

public void listRegionalBuckets() {

    ListRegionalBucketsRequest reqListBuckets = new ListRegionalBucketsRequest()
            .withAccountId(AccountId)
            .withOutpostId(OutpostId);
```

```
    ListRegionalBucketsResult respListBuckets =
 s3ControlClient.listRegionalBuckets(reqListBuckets);
    System.out.printf("ListRegionalBuckets Response: %s%n",
 respListBuckets.toString());

}
```

使用 AWS CLI 和适用于 Java 的 SDK 获取 S3 on Outposts 存储桶

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

以下示例显示了如何使用 AWS CLI 和 适用于 Java 的 AWS SDK 获取 S3 on Outposts 存储桶。

> **Note**
>
> 当您通过 AWS CLI 或 AWS SDK 使用 Amazon S3 on Outposts 时，您可以提供 Outpost 的访问点 ARN 来代替存储桶名称。访问点 ARN 采用以下形式，其中 *region* 是 Outpost 归属的区域的 AWS 区域代码：
>
> `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point`
>
> 有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

使用 AWS CLI

以下 S3 on Outposts 示例使用 AWS CLI 获取存储桶。要使用此命令，请将每个 *user input placeholder* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 get-bucket。

```
aws s3control get-bucket --account-id 123456789012 --bucket "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket"
```

使用适用于 Java 的 AWS SDK

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 获取存储桶。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 GetBucket。

```
import com.amazonaws.services.s3control.model.*;

public void getBucket(String bucketArn) {

    GetBucketRequest reqGetBucket = new GetBucketRequest()
            .withBucket(bucketArn)
            .withAccountId(AccountId);

    GetBucketResult respGetBucket = s3ControlClient.getBucket(reqGetBucket);
    System.out.printf("GetBucket Response: %s%n", respGetBucket.toString());

}
```

删除 Amazon S3 on Outposts 存储桶

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

有关使用 S3 on Outposts 中的存储桶的更多信息，请参阅使用 S3 on Outposts 存储桶。

创建存储桶的 AWS 账户 拥有该存储桶，也是唯一可以删除它的账户。

> **Note**
>
> - Outposts 存储桶必须为空才能进行删除。
>
> Amazon S3 控制台不支持 S3 on Outposts 对象操作。要删除 S3 on Outposts 存储桶中的对象，您可以使用 AWS CLI、AWS 软件开发工具包或 REST API。

- 在删除 Outposts 存储桶之前，必须先删除该存储桶的任何 Outposts 访问点。有关更多信息，请参阅 删除接入点。
- 存储桶在删除后不能恢复。

以下示例显示如何使用 AWS 管理控制台和 AWS Command Line Interface (AWS CLI) 删除 S3 on Outposts 存储桶。

使用 S3 控制台

1. 登录到AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择要删除的存储桶，然后选择 Delete（删除）。
4. 确认删除操作。

使用 AWS CLI

以下示例使用 AWS CLI 删除 S3 on Outposts 存储桶 (`s3-outposts:DeleteBucket`)。要运行此命令，请将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control delete-bucket --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

使用 Amazon S3 on Outposts 访问点

要访问 Amazon S3 on Outposts 存储桶，您必须创建和配置访问点。

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

> **Note**
>
> 创建 Outposts 存储桶的 AWS 账户 拥有该存储桶，也是唯一可以为其分配访问点的账户。

以下部分介绍如何创建和管理 S3 on Outposts 存储桶的访问点。

主题

创建 S3 on Outposts 访问点

要访问 Amazon S3 on Outposts 存储桶，您必须创建和配置访问点。

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

以下示例显示如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 创建 S3 on Outposts 访问点。

> **Note**
>
> 创建 Outposts 存储桶的 AWS 账户 拥有该存储桶，也是唯一可以为其分配访问点的账户。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择要为其创建 Outposts 访问点的 Outposts 存储桶。
4. 选择 Outposts 访问点选项卡。
5. 在 Outposts access points（Outposts 访问点）部分中，选择 Create Outposts access point（创建 Outposts 访问点）。

6. 在 Outposts access point settings（Outposts 访问点设置）中，输入访问点的名称，然后选择访问点的 Virtual Private Cloud (VPC)。
7. 如果要为访问点添加策略，请在 Outposts access point policy（Outposts 访问点策略）部分中输入策略。

   有关更多信息，请参阅 使用 S3 on Outposts 设置 IAM。

使用 AWS CLI

Example

以下 AWS CLI 示例为 Outposts 存储桶创建访问点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

使用适用于 Java 的 AWS 软件开发工具包

Example

有关如何使用适用于 Java 的 AWS SDK 为 S3 Outposts 存储桶创建接入点的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsAccessPoint.java。

为您的 S3 on Outposts 桶访问点使用桶式别名

对于 S3 on Outposts，您必须使用接入点访问 Outposts 存储桶中的任何对象。每次您为存储桶创建接入点时，S3 on Outposts 都会自动生成一个接入点别名。对于任何数据面板操作，您都可以使用此访问点别名，而不是访问点 ARN。例如，您可以使用访问点别名来执行对象级操作，例如 PUT、GET、LIST 等。有关这些操作的列表，请参阅用于管理对象的 Amazon S3 API 操作。

以下示例显示了名为 *my-access-point* 的访问点的 ARN 和访问点别名。

- 访问点 ARN – `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/my-access-point`
- 访问点别名 – *my-access-po-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10*--op-s3

有关 ARN 的更多信息，请参阅《AWS 一般参考》中的 Amazon 资源名称（ARN）。

有关访问点别名的更多信息，请参阅以下主题：

主题

访问点别名

访问点别名是在与 S3 on Outposts 桶相同的命名空间中创建的。当您创建访问点时，S3 on Outposts 会自动生成一个无法更改的访问点别名。访问点别名符合有效 S3 on Outposts 桶名称的所有要求，包括以下部分：

access point name prefix`-metadata--op-s3`

> **Note**
>
> `--op-s3` 后缀保留用于访问点别名，我们建议您不要将其用于桶或访问点名称。有关 S3 on Outposts 桶命名规则的更多信息，请参阅使用 S3 on Outposts 存储桶。

查找访问点别名

以下示例显示了如何使用 AmazonS3 控制台和 AWS CLI 查找访问点别名。

Example：在 Amazon S3 控制台中查找并复制访问点别名

在控制台中创建访问点后，可以从 Access Points（访问点）列表的 Access Point alias（访问点别名）列中获取访问点别名。

复制访问点别名

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 要复制访问点别名，请执行以下操作之一：

   - 在 Access Points（访问点）列表中，选择访问点名称旁边的选项按钮，然后选择 Copy Access Point alias（复制访问点别名）。

- 请选择接入点名称。然后，在 Outposts access point overview（Outposts 访问点概述）下，复制访问点别名。

Example：使用 AWS CLI 创建访问点并在响应中查找访问点别名

以下有关 `create-access-point` 命令的 AWS CLI 示例创建访问点并返回自动生成的访问点别名。要运行此命令，请将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control create-access-point --bucket example-outposts-bucket --name example-
outposts-access-point --account-id 123456789012

{
    "AccessPointArn":
    "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
accesspoint/example-outposts-access-point",
    "Alias": "example-outp-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example：使用 AWS CLI 获取访问点别名

例如，以下有关 `get-access-point` 命令的 AWS CLI 示例返回关于指定的访问点的信息。此信息包括访问点别名。要运行此命令，请将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control get-access-point --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --name example-outposts-access-point --account-id 123456789012

{
    "Name": "example-outposts-access-point",
    "Bucket": "example-outposts-bucket",
    "NetworkOrigin": "Vpc",
    "VpcConfiguration": {
        "VpcId": "vpc-01234567890abcdef"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2022-09-18T17:49:15.584000+00:00",
```

```
    "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example：使用 AWS CLI 列出访问点以查找访问点别名

例如，以下有关 `list-access-points` 命令的 AWS CLI 示例列出关于指定的访问点的信息。此信息包括访问点别名。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket

{
    "AccessPointList": [
        {
            "Name": "example-outposts-access-point",
            "NetworkOrigin": "Vpc",
            "VpcConfiguration": {
                "VpcId": "vpc-01234567890abcdef"
            },
            "Bucket": "example-outposts-bucket",
            "AccessPointArn": "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point",
            "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
        }
    ]
}
```

在 S3 on Outposts 对象操作中使用访问点别名

采用访问点时，您可以使用访问点别名，而无需对代码进行大量更改。

此 AWS CLI 示例显示了针对 S3 on Outposts 桶的 `get-object` 操作。此示例使用访问点别名作为 `--bucket` 的值，而不是完整的访问点 ARN。

```
aws s3api get-object --bucket my-access-po-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3 --key  testkey  sample-object.rtf

{
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
```

```
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

限制

- 客户无法配置别名。
- 无法在访问点上删除、修改或禁用别名。
- 您不能将访问点别名用于 S3 on Outposts 控制面板操作。有关 S3 on Outposts 控制面板操作的列表，请参阅用于管理存储桶的 Amazon S3 控制 API。
- 别名不能用于 AWS Identity and Access Management（IAM）策略中。

查看有关访问点配置的信息

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

以下主题说明如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 返回 S3 on Outposts 访问点的配置信息。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 选择要查看其配置详细信息的 Outposts 访问点。
4. 在 Outposts access point overview（Outposts 访问点概述）下，查看访问点配置详细信息。

使用 AWS CLI

以下 AWS CLI 示例获取 Outpost 存储桶的访问点。将 *`user input placeholders`* 替换为您自己的信息。

```
aws s3control get-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

使用适用于 Java 的 AWS 软件开发工具包

以下适用于 Java 的软件开发工具包示例为 Outpost 存储桶获取访问点。

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPoint(String accessPointArn) {

    GetAccessPointRequest reqGetAP = new GetAccessPointRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointResult respGetAP = s3ControlClient.getAccessPoint(reqGetAP);
    System.out.printf("GetAccessPoint Response: %s%n", respGetAP.toString());

}
```

查看 Amazon S3 on Outposts 访问点的列表

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

以下主题说明如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 返回 S3 on Outposts 访问点的列表。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 在 Outposts access points（Outposts 访问点）下，查看您的 S3 on Outposts 访问点列表。

使用 AWS CLI

以下 AWS CLI 示例列出 Outposts 存储桶的接入点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

使用适用于 Java 的 AWS 软件开发工具包

以下 SDK for Java 示例列出 Outposts 存储桶的接入点。

```
import com.amazonaws.services.s3control.model.*;

public void listAccessPoints(String bucketArn) {

    ListAccessPointsRequest reqListAPs = new ListAccessPointsRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    ListAccessPointsResult respListAPs = s3ControlClient.listAccessPoints(reqListAPs);
    System.out.printf("ListAccessPoints Response: %s%n", respListAPs.toString());

}
```

删除接入点

访问点可简化对 Amazon S3 中的共享数据集的大规模数据访问管理。访问点是附加到存储桶的命名网络端点，您可以使用这些存储桶执行 Amazon S3 对象操作（如 `GetObject` 和 `PutObject`）。对于 S3 on Outposts，您必须使用访问点访问 Outposts 存储桶中的任何对象。访问点仅支持虚拟主机式寻址。

以下示例显示如何使用 AWS 管理控制台和 AWS Command Line Interface (AWS CLI) 删除访问点。

使用 S3 控制台

1. 通过以下网址打开 Simple Storage Service（Amazon S3）控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 在 Outposts access points（Outposts 访问点）部分中，选择要删除的 Outposts 访问点。

4. 选择 Delete。
5. 确认删除操作。

使用 AWS CLI

以下 AWS CLI 示例删除 Outposts 访问点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control delete-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

添加或编辑访问点策略

访问点具有不同的权限和网络控制，Amazon S3 on Outposts 将它们应用于通过该访问点发出的任何请求。每个接入点强制实施自定义接入点策略，该策略与附加到底层存储桶的存储桶策略结合使用。有关更多信息，请参阅 接入点。

以下主题说明了如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和适用于 Java 的 AWS SDK 添加或编辑您的 S3 on Outposts 访问点策略。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 存储桶）。
3. 选择要为其编辑访问点策略的 Outposts 存储桶。
4. 选择 Outposts 访问点选项卡。
5. 在 Outposts access points（Outposts 访问点）部分中，选择要编辑其策略的访问点，然后选择 Edit policy（编辑策略）。
6. 在 Outposts access point policy（Outposts 访问点策略）部分中添加或编辑策略。有关更多信息，请参阅 使用 S3 on Outposts 设置 IAM。

使用 AWS CLI

以下 AWS CLI 示例在 Outposts 访问点上放置策略。

1. 将以下访问点策略保存到 JSON 文件中。在本示例中，文件命名为 `appolicy1.json`。将 *user input placeholders* 替换为您自己的信息。

```
{
   "Version":"2012-10-17",
   "Id":"exampleAccessPointPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point
      }
   ]
}
```

2. 将 JSON 文件作为 put-access-point-policy CLI 命令的一部分提交。将 *user input placeholders* 替换为您自己的信息。

```
aws s3control put-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --policy file://appolicy1.json
```

使用适用于 Java 的 AWS 软件开发工具包

以下 SDK for Java 示例在 Outposts 接入点上放置策略。

```
import com.amazonaws.services.s3control.model.*;

public void putAccessPointPolicy(String accessPointArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testAccessPointPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId + "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + accessPointArn +
 "\"}]}";

    PutAccessPointPolicyRequest reqPutAccessPointPolicy = new
 PutAccessPointPolicyRequest()
            .withAccountId(AccountId)
```

```
            .withName(accessPointArn)
            .withPolicy(policy);

    PutAccessPointPolicyResult respPutAccessPointPolicy =
 s3ControlClient.putAccessPointPolicy(reqPutAccessPointPolicy);
    System.out.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());
    printWriter.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());

}
```

查看 S3 on Outposts 访问点策略

访问点具有不同的权限和网络控制，Amazon S3 on Outposts 将它们应用于通过该访问点发出的任何请求。每个接入点强制实施自定义接入点策略，该策略与附加到底层存储桶的存储桶策略结合使用。有关更多信息，请参阅 接入点。

有关使用 S3 on Outposts 中的访问点的更多信息，请参阅使用 S3 on Outposts 存储桶。

以下主题说明如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 查看您的 S3 on Outposts 访问点策略。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 选择要查看其策略的 Outposts 访问点。
4. 在 Permissions（权限）选项卡上，查看 S3 on Outposts 访问点策略。
5. 要编辑访问点策略，请参阅添加或编辑访问点策略。

使用 AWS CLI

以下 AWS CLI 示例获取 Outposts 接入点策略。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control get-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

使用适用于 Java 的 AWS 软件开发工具包

以下 SDK for Java 示例获取 Outposts 接入点策略。

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPointPolicy(String accessPointArn) {

    GetAccessPointPolicyRequest reqGetAccessPointPolicy = new
 GetAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointPolicyResult respGetAccessPointPolicy =
 s3ControlClient.getAccessPointPolicy(reqGetAccessPointPolicy);
    System.out.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());
    printWriter.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());

}
```

使用 Amazon S3 on Outposts 端点

要将请求路由到 Amazon S3 on Outposts 访问点，您必须创建并配置 S3 on Outposts 端点。为了创建端点，您需要使用服务链接建立到 Outposts 主区域的活跃连接。Outposts 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的端点。有关端点配额的更多信息，请参阅S3 on Outposts 网络要求。您必须创建一个端点，才能访问 Outposts 桶并执行对象操作。有关更多信息，请参阅 端点。

创建端点后，您可以使用“状态”字段来了解端点的状态。如果您的 Outposts 处于离线状态，它将返回 CREATE_FAILED。您可以检查服务链接连接，删除端点，并在连接恢复后重试创建操作。有关其他错误代码的列表，请参阅下文。有关更多信息，请参阅 端点。

API	Status	失败原因错误代码	消息 - 失败原因
`CreateEndpoint`	Create_Failed	OutpostNotReachable	由于与您的 Outposts 主区域的服务链接连接已断开，因此无法创建端点。请检查您的连接，删除端点，然后重试。

API	Status	失败原因错误代码	消息 - 失败原因
`CreateEndpoint`	Create_Failed	InternalError	由于内部错误，无法创建端点。请删除端点并重新创建。
`DeleteEndpoint`	Delete_Failed	OutpostNotReachable	由于与您的 Outposts 主区域的服务链接连接已断开，因此无法删除端点。检查连接，然后重试。
`DeleteEndpoint`	Delete_Failed	InternalError	由于内部错误，无法删除端点。请重试。

有关使用 S3 on Outposts 中的存储桶的更多信息，请参阅使用 S3 on Outposts 存储桶。

下面各部分介绍了如何创建和管理 S3 on Outposts 的端点。

主题

- 在 Outpost 上创建端点
- 查看 Amazon S3 on Outposts 端点列表
- 删除 Amazon S3 on Outposts 终端节点

在 Outpost 上创建端点

要将请求路由到 Amazon S3 on Outposts 访问点，您必须创建 S3 on Outposts 端点并进行配置。为了创建端点，您需要使用服务链接建立到 Outposts 主区域的活跃连接。Outposts 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的端点。有关端点配额的更多信息，请参阅S3 on Outposts 网络要求。您必须创建一个端点，才能访问 Outposts 桶并执行对象操作。有关更多信息，请参阅 端点。

权限

有关创建端点所需权限的更多信息，请参阅 S3 on Outposts 端点的权限。

当您创建端点时，S3 on Outposts 还会在您的 AWS 账户中创建一个服务相关角色。有关更多信息，请参阅 将服务相关角色用于 Amazon S3 on Outposts。

以下示例显示如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 创建 S3 on Outposts 端点。

使用 S3 控制台

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 选择 Outposts endpoints（Outposts 端点）选项卡。
4. 选择 Create Outposts endpoint（创建 Outposts 端点）。
5. 在 Outpost 下，选择 Outpost 以在其上创建此端点。
6. 在 VPC 下，选择一个还没有端点且也符合 Outposts 端点规则的 VPC。

   通过 Virtual Private Cloud (VPC)，您可以将 AWS 资源启动到您定义的虚拟网络中。这个虚拟网络与您在自己的数据中心中运行的传统网络极其相似，并会为您提供使用的可扩展基础设施的优势 AWS

   如果您没有 VPC，请选择 Create VPC（创建 VPC）。有关更多信息，请参阅《Amazon S3 用户指南》中的创建限制到虚拟私有云（VPC）的接入点。
7. 选择 Create Outposts endpoint（创建 Outposts 端点）。

使用 AWS CLI

Example

以下 AWS CLI 示例使用 VPC 资源访问类型，为 Outpost 创建端点。VPC 派生自子网。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

以下 AWS CLI 示例使用客户拥有的 IP 地址池（CoIP 池）访问类型为 Outpost 创建端点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

使用适用于 Java 的 AWS 软件开发工具包

Example

有关如何使用适用于 Java 的 AWS SDK 为 S3 Outpost 创建端点的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 CreateOutpostsEndPoint.java。

查看 Amazon S3 on Outposts 端点列表

要将请求路由到 Amazon S3 on Outposts 访问点，您必须创建 S3 on Outposts 端点并进行配置。为了创建端点，您需要使用服务链接建立到 Outposts 主区域的活跃连接。Outposts 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的端点。有关端点配额的更多信息，请参阅S3 on Outposts 网络要求。您必须创建一个端点，才能访问 Outposts 桶并执行对象操作。有关更多信息，请参阅 端点。

以下示例显示了如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 返回 S3 on Outposts 端点的列表。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 在 Outposts access points（Outposts 访问点）页上，选择 Outposts endpoints（Outposts 端点）选项卡。
4. 在 Outposts endpoints（Outposts 端点）下，您可以查看 S3 on Outposts 端点的列表。

使用 AWS CLI

以下 AWS CLI 示例列出了与您的账户关联的 AWS Outposts 资源的端点。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 list-endpoints。

```
aws s3outposts list-endpoints
```

使用适用于 Java 的 AWS SDK

以下适用于 Java 的 SDK 示例列出 Outpost 的端点。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 ListEndpoints。

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
```

```
import com.amazonaws.services.s3outposts.model.ListEndpointsRequest;
import com.amazonaws.services.s3outposts.model.ListEndpointsResult;

public void listEndpoints() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    ListEndpointsRequest listEndpointsRequest = new ListEndpointsRequest();
    ListEndpointsResult listEndpointsResult =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
    System.out.println("List endpoints result is " + listEndpointsResult);
}
```

删除 Amazon S3 on Outposts 终端节点

要将请求路由到 Amazon S3 on Outposts 访问点，您必须创建 S3 on Outposts 端点并进行配置。为了创建端点，您需要使用服务链接建立到 Outposts 主区域的活跃连接。Outposts 上的每个 Virtual Private Cloud (VPC) 都可以有一个关联的端点。有关端点配额的更多信息，请参阅S3 on Outposts 网络要求。您必须创建一个端点，才能访问 Outposts 桶并执行对象操作。有关更多信息，请参阅 端点。

以下示例显示如何使用 AWS 管理控制台、AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 删除 S3 on Outposts 终端节点。

使用 S3 控制台

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts access points（Outposts 访问点）。
3. 在 Outposts access points（Outposts 访问点）页上，选择 Outposts endpoints（Outposts 端点）选项卡。
4. 在 Outposts endpoints（Outposts 终端节点）下，选择要删除的终端节点，然后选择 Delete（删除）。

使用 AWS CLI

以下 AWS CLI 示例删除 Outpost 的终端节点。要运行此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3outposts delete-endpoint --endpoint-id example-endpoint-id --outpost-
id op-01ac5d28a6a232904
```

使用适用于 Java 的 AWS 软件开发工具包

以下适用于 Java 的软件开发工具包示例会删除 Outpost 的终端节点。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
import com.amazonaws.arn.Arn;
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.DeleteEndpointRequest;

public void deleteEndpoint(String endpointArnInput) {
    String outpostId = "op-01ac5d28a6a232904";
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    Arn endpointArn = Arn.fromString(endpointArnInput);
    String[] resourceParts = endpointArn.getResource().getResource().split("/");
    String endpointId = resourceParts[resourceParts.length - 1];
    DeleteEndpointRequest deleteEndpointRequest = new DeleteEndpointRequest()
                .withEndpointId(endpointId)
                .withOutpostId(outpostId);
    s3OutpostsClient.deleteEndpoint(deleteEndpointRequest);
    System.out.println("Endpoint with id " + endpointId + " is deleted.");
}
```

使用 S3 on Outposts 对象

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

对象 ARN 使用以下格式，其中包括 Outpost 归属的 AWS 区域、AWS 账户 ID、Outpost ID、存储桶名称和对象键：

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/bucket/amzn-s3-demo-bucket1/object/myobject
```

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

主题

- 将对象上传到 S3 on Outposts 存储桶
- 使用 适用于 Java 的 AWS SDK 复制 Amazon S3 on Outposts 存储桶中的对象
- 从 Amazon S3 on Outposts 存储桶获取对象

- 列出 Amazon S3 on Outposts 存储桶中的对象
- 删除 Amazon S3 on Outposts 存储桶中的对象
- 使用 HeadBucket 确定是否存在 S3 on Outposts 存储桶以及您是否具有访问权限。
- 使用适用于 Java 的 SDK 执行和管理分段上传
- 使用适用于 S3 on Outposts 的预签名 URL
- Amazon S3 on Outposts 与本地 Amazon EMR on Outposts
- 授权和身份验证缓存

将对象上传到 S3 on Outposts 存储桶

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

以下 AWS CLI 和适用于 Java 的 AWS SDK 示例演示如何通过使用访问点将对象上传到 S3 on Outposts 存储桶。

AWS CLI

Example

以下示例使用 AWS CLI 将一个名为 `sample-object.xml` 的对象放置到 S3 on Outposts 存储桶 (`s3-outposts:PutObject`)。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 put-object。

```
aws s3api put-object --bucket arn:aws:s3-
outposts:Region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml --body sample-object.xml
```

SDK for Java

Example

有关如何使用适用于 Java 的 AWS SDK 将对象上传到 S3 Outposts 存储桶的示例，请参阅《AWS SDK for Java 2.x Code Examples》中的 PutObjectOnOutpost.java。

使用 适用于 Java 的 AWS SDK 复制 Amazon S3 on Outposts 存储桶中的对象

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

以下示例显示如何使用 适用于 Java 的 AWS SDK 复制 S3 on Outposts 存储桶中的对象。

使用适用于 Java 的 AWS SDK

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 将一个对象复制到同一个存储桶中的新对象。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

public class CopyObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String sourceKey = "*** Source object key ***";
        String destinationKey = "*** Destination object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(accessPointArn,
 sourceKey, accessPointArn, destinationKey);
            s3Client.copyObject(copyObjectRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

从 Amazon S3 on Outposts 存储桶获取对象

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

下面的示例演示如何使用 AWS Command Line Interface (AWS CLI) 和适用于 Java 的 AWS SDK 下载（获取）对象。

使用 AWS CLI

以下示例使用 AWS CLI 从 S3 on Outposts 存储桶 (`s3-outposts:GetObject`) 获取一个名为 `sample-object.xml` 的对象。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 get-object。

```
aws s3api get-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point --key testkey sample-object.xml
```

使用适用于 Java 的 AWS SDK

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 获取对象。要使用此示例，请将每个 *`user input placeholder`* 替换为您自己的信息。有关更多信息，请参阅 Amazon Simple Storage Service API 参考中的 GetObject。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ResponseHeaderOverrides;
import com.amazonaws.services.s3.model.S3Object;

import java.io.BufferedReader;
```

```
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

public class GetObject {
    public static void main(String[] args) throws IOException {
        String accessPointArn = "*** access point ARN ***";
        String key = "*** Object key ***";

        S3Object fullObject = null, objectPortion = null, headerOverrideObject = null;
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Get an object and print its contents.
            System.out.println("Downloading an object");
            fullObject = s3Client.getObject(new GetObjectRequest(accessPointArn, key));
            System.out.println("Content-Type: " +
 fullObject.getObjectMetadata().getContentType());
            System.out.println("Content: ");
            displayTextInputStream(fullObject.getObjectContent());

            // Get a range of bytes from an object and print the bytes.
            GetObjectRequest rangeObjectRequest = new GetObjectRequest(accessPointArn,
 key)
                    .withRange(0, 9);
            objectPortion = s3Client.getObject(rangeObjectRequest);
            System.out.println("Printing bytes retrieved.");
            displayTextInputStream(objectPortion.getObjectContent());

            // Get an entire object, overriding the specified response headers, and
 print the object's content.
            ResponseHeaderOverrides headerOverrides = new ResponseHeaderOverrides()
                    .withCacheControl("No-cache")
                    .withContentDisposition("attachment; filename=example.txt");
            GetObjectRequest getObjectRequestHeaderOverride = new
 GetObjectRequest(accessPointArn, key)
                    .withResponseHeaders(headerOverrides);
            headerOverrideObject = s3Client.getObject(getObjectRequestHeaderOverride);
            displayTextInputStream(headerOverrideObject.getObjectContent());
```

```
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        } finally {
            // To ensure that the network connection doesn't remain open, close any
 open input streams.
            if (fullObject != null) {
                fullObject.close();
            }
            if (objectPortion != null) {
                objectPortion.close();
            }
            if (headerOverrideObject != null) {
                headerOverrideObject.close();
            }
        }
    }

    private static void displayTextInputStream(InputStream input) throws IOException {
        // Read the text input stream one line at a time and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

列出 Amazon S3 on Outposts 存储桶中的对象

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅[为您的 S3 on Outposts 桶访问点使用桶式别名]。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

> **Note**
>
> 对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

以下示例显示如何使用 AWS CLI 和适用于 Java 的 AWS SDK 列出 S3 on Outposts 存储桶中的对象。

使用 AWS CLI

以下示例使用 AWS CLI 列出 S3 on Outposts 存储桶 (`s3-outposts:ListObjectsV2`) 中的对象。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 list-objects-v2。

```
aws s3api list-objects-v2 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

> **Note**
>
> 通过 AWS SDK 将此操作与 Amazon S3 on Outposts 结合使用时，您可以通过以下表单使用 Outposts 访问点 ARN 代替存储桶的名称：`arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-Outposts-Access-Point`。有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

使用适用于 Java 的 AWS SDK

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 列出桶中的对象。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

> **Important**
>
> 此示例使用 ListObjectsV2，这是 ListObjects API 操作的最新版本。我们建议您使用此修订后的 API 操作进行应用程序开发。为了实现向后兼容，Amazon S3 继续支持此 API 操作的先前版本。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsV2Request;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;

public class ListObjectsV2 {

    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            System.out.println("Listing objects");

            // maxKeys is set to 2 to demonstrate the use of
            // ListObjectsV2Result.getNextContinuationToken()
            ListObjectsV2Request req = new
 ListObjectsV2Request().withBucketName(accessPointArn).withMaxKeys(2);
            ListObjectsV2Result result;

            do {
```

```
                result = s3Client.listObjectsV2(req);

                for (S3ObjectSummary objectSummary : result.getObjectSummaries()) {
                    System.out.printf(" - %s (size: %d)\n", objectSummary.getKey(),
 objectSummary.getSize());
                }
                // If there are more than maxKeys keys in the bucket, get a
 continuation token
                // and list the next objects.
                String token = result.getNextContinuationToken();
                System.out.println("Next Continuation Token: " + token);
                req.setContinuationToken(token);
            } while (result.isTruncated());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

删除 Amazon S3 on Outposts 存储桶中的对象

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在

AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

以下示例显示如何使用 AWS Command Line Interface (AWS CLI) 和适用于 Java 的 AWS SDK 从 S3 on Outposts 存储桶删除单个或多个对象。

使用 AWS CLI

以下示例显示如何从 S3 on Outposts 存储桶删除单个或多个对象。

delete-object

以下示例使用 AWS CLI 从 S3 on Outposts 存储桶 (`s3-outposts:DeleteObject`) 删除一个名为 `sample-object.xml` 的对象。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 delete-object。

```
aws s3api delete-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml
```

delete-objects

以下示例使用 AWS CLI 从 S3 on Outposts 存储桶 (`s3-outposts:DeleteObject`) 删除名为 `sample-object.xml` 和 `test1.text` 的两个对象。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 delete-objects。

```
aws s3api delete-objects --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --delete file://delete.json

delete.json
{
  "Objects": [
    {
      "Key": "test1.txt"
    },
    {
      "Key": "sample-object.xml"
```

```
    }
  ],
  "Quiet": false
}
```

使用适用于 Java 的 AWS SDK

以下示例显示如何从 S3 on Outposts 存储桶删除单个或多个对象。

DeleteObject

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 删除存储桶中的对象。要使用此示例，请指定 Outpost 的访问点 ARN 和要删除的对象的密钥名称。有关更多信息，请参阅 Amazon Simple Storage Service API 参考中的 DeleteObject。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

public class DeleteObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** key name ****";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(accessPointArn, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
```

```
            e.printStackTrace();
        }
    }
}
```

DeleteObjects

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 向存储桶上传对象，然后删除对象。要使用此示例，请指定 Outpost 的访问点 ARN。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 DeleteObjects。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectsRequest;
import com.amazonaws.services.s3.model.DeleteObjectsRequest.KeyVersion;
import com.amazonaws.services.s3.model.DeleteObjectsResult;

import java.util.ArrayList;

public class DeleteObjects {

    public static void main(String[] args) {
       String accessPointArn = "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point";


        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload three sample objects.
            ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
            for (int i = 0; i < 3; i++) {
                String keyName = "delete object example " + i;
                s3Client.putObject(accessPointArn, keyName, "Object number " + i + "
 to be deleted.");
```

```
                keys.add(new KeyVersion(keyName));
            }
            System.out.println(keys.size() + " objects successfully created.");

            // Delete the sample objects.
            DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(accessPointArn)
                    .withKeys(keys)
                    .withQuiet(false);

            // Verify that the objects were deleted successfully.
            DeleteObjectsResult delObjRes =
 s3Client.deleteObjects(multiObjectDeleteRequest);
            int successfulDeletes = delObjRes.getDeletedObjects().size();
            System.out.println(successfulDeletes + " objects successfully
 deleted.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

使用 HeadBucket 确定是否存在 S3 on Outposts 存储桶以及您是否具有访问权限。

对象是 Amazon S3 on Outposts 中存储的基础实体。每个对象都储存在一个存储桶中。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。当您为对象操作指定桶时，可以使用访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

以下示例显示了 S3 on Outposts 访问点的 ARN 格式，其中包括 Outpost 所属区域的 AWS 区域 代码、AWS 账户 ID、Outpost ID 和访问点名称：

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

有关 S3 on Outposts ARN 的更多信息，请参阅S3 on Outposts 的资源 ARN。

> **Note**
>
> 对于 Amazon S3 on Outposts，对象数据始终存储在 Outpost 上。当 AWS 安装 Outpost 机架时，您的数据将保留在 Outpost 的本地，以满足数据驻留要求。您的对象永远不会离开您的 Outpost，也不在 AWS 区域 中。由于 AWS 管理控制台 托管在区域内，您无法使用控制台上传或管理 Outpost 中的对象。但是，您可以使用 REST API、AWS Command Line Interface (AWS CLI) 和 AWS SDK 通过访问点上传和管理对象。

以下 AWS Command Line Interface (AWS CLI) 和 适用于 Java 的 AWS SDK 示例显示如何使用 HeadBucket API 操作确定 Amazon S3 on Outposts 存储桶是否存在以及您是否有权访问该存储桶。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 HeadBucket。

使用 AWS CLI

以下 S3 on Outposts AWS CLI 示例显示如何使用 `head-bucket` 命令确定存储桶是否存在以及您是否有权访问该存储桶。要使用此命令，请将每个 *`user input placeholder`* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 head-bucket。

```
aws s3api head-bucket --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

使用适用于 Java 的 AWS SDK

以下 S3 on Outposts 示例显示如何确定存储桶是否存在以及您是否有权访问该存储桶。要使用此示例，请指定 Outpost 的访问点 ARN。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 HeadBucket。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.HeadBucketRequest;

public class HeadBucket {
    public static void main(String[] args) {
```

```
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.headBucket(new HeadBucketRequest(accessPointArn));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

使用适用于 Java 的 SDK 执行和管理分段上传

使用 Amazon S3 on Outposts，您可以在 AWS Outposts 资源上创建 S3 存储桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序存储和检索对象。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅 什么是 Amazon S3 on Outposts？

以下示例显示如何结合使用 S3 on Outposts 和 适用于 Java 的 AWS SDK 来执行和管理分段上传。

主题

- 在 S3 on Outposts 存储桶中执行对象的分段上传
- 通过分段上传，在 S3 on Outposts 存储桶中复制大型对象
- 在 S3 on Outposts 存储桶中列出对象的分段
- 检索 S3 on Outposts 存储桶中正在进行的分段上传的列表

在 S3 on Outposts 存储桶中执行对象的分段上传

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 启动、上传和完成对象到存储桶的分段上传。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的使用分段上传操作上传对象。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
```

```
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
```

通过分段上传，在 S3 on Outposts 存储桶中复制大型对象

下面的 S3 on Outposts 示例使用适用于 Java 的 SDK 复制存储桶中的对象。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);
```

```
            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
```

```
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

在 S3 on Outposts 存储桶中列出对象的分段

以下 S3 on Outposts 示例使用适用于 Java 的 SDK 列出存储桶中对象的分段。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.List;

public class ListParts {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** Key name ***";
        String uploadId = "*** Upload ID ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
```

```
                    .build();

            ListPartsRequest listPartsRequest = new ListPartsRequest(accessPointArn,
 keyName, uploadId);
            PartListing partListing = s3Client.listParts(listPartsRequest);
            List<PartSummary> partSummaries = partListing.getParts();

            System.out.println(partSummaries.size() + " multipart upload parts");
            for (PartSummary p : partSummaries) {
                System.out.println("Upload part: Part number = \"" + p.getPartNumber()
 + "\", ETag = " + p.getETag());
            }

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

检索 S3 on Outposts 存储桶中正在进行的分段上传的列表

以下 S3 on Outposts 示例说明如何使用适用于 Java 的 SDK 从 Outposts 存储桶检索正在进行的分段上传的列表。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class ListMultipartUploads {
    public static void main(String[] args) {
                String accessPointArn = "*** access point ARN ***";
```

```
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Retrieve a list of all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(accessPointArn);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            List<MultipartUpload> uploads =
 multipartUploadListing.getMultipartUploads();

            // Display information about all in-progress multipart uploads.
            System.out.println(uploads.size() + " multipart upload(s) in progress.");
            for (MultipartUpload u : uploads) {
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\",
 id = " + u.getUploadId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

使用适用于 S3 on Outposts 的预签名 URL

要授予对 Outpost 本地存储对象的限时访问权限而不更新存储桶策略，您可以使用预签名 URL。借助预签名 URL，作为存储桶的所有者，您可以与您虚拟私有云（VPC）中的个人共享对象，或者向其授予上传或删除对象的权限。

使用 AWS SDK 或 AWS Command Line Interface（AWS CLI）创建预签名 URL 时，您会将该 URL 与某个特定的操作关联。您还可以通过选择自定义到期时间来授予对预签名 URL 的限时访问权限，自定义到期时间最短可为 1 秒，最长可为 7 天。共享预签名 URL 时，VPC 中的个人可以执行嵌入在 URL 中的操作，如同他们就是原始签名用户。URL 在到达其到期时间时将会过期，不再有效。

限制预签名 URL 功能

预签名 URL 的功能受创建它的用户的权限所限制。预签名 URL 实质上是一种不记名令牌，向持有相关 URL 的人授予了访问权限。因此，我们建议您适当地保护它们。

AWS 签名版本 4（SigV4）

使用 AWS 签名版本 4（SigV4）对预签名 URL 请求进行身份认证时要强制执行特定的行为，您可以在存储桶策略和访问点策略中使用条件键。例如，您可以创建一个使用 `s3-outposts:signatureAge` 条件的存储桶策略，以在相关签名的存在时间超过 10 分钟时，拒绝对 `example-outpost-bucket` 存储桶中对象的任何 Amazon S3 on Outposts 预签名 URL 请求。要使用此示例，请将 *`user input placeholders`* 替换为您自己的信息。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/
object/*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

有关在使用签名版本 4 对预签名 URL 请求进行身份认证时，可用于强制执行特定行为的条件键和其他示例策略的列表，请参阅 AWS 签名版本 4（SigV4）身份认证特定的策略键。

网络路径限制

如果要将预签名 URL 的使用和所有 S3 on Outposts 访问限定为特定的网络路径，您可以编写要求使用特定网络路径的策略。要对发起调用的 IAM 主体设置限制，您可以使用基于身份的 AWS Identity and Access Management（IAM）策略（例如用户、组或角色策略）。要对 S3 on Outposts 资源设置限制，您可以使用基于资源的策略（例如，存储桶和访问点策略）。

对 IAM 主体实施网络路径限制后，要求拥有这些凭证的用户从指定的网络发出请求。对存储桶或接入点实施限制后，要求对该资源的所有请求都必须来自指定的网络。这些限制也适用于预签名 URL 以外的场景。

您使用的 IAM 全局条件取决于端点的类型。如果您使用适用于 S3 on Outposts 的公有端点，请使用 `aws:SourceIp`。如果您使用适用于 S3 on Outposts 的 VPC 端点，请使用 `aws:SourceVpc` 或 `aws:SourceVpce`。

以下 IAM policy 语句要求主体仅从指定的网络范围访问。AWS使用此策略语句时，所有访问都必须来自该范围。这包括有人使用适用于 S3 on Outposts 的预签名 URL 的情况。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
    }
}
```

有关使用 `aws:SourceIP` AWS 全局条件键将对 S3 on Outposts 存储桶的访问限定为特定网络范围的示例存储桶策略，请参阅使用 S3 on Outposts 设置 IAM。

谁可以创建预签名 URL

具有有效安全凭证的任何人都可以创建预签名 URL。但要使 VPC 中的用户成功地访问对象，必须由拥有执行预签名 URL 所基于的操作权限的人创建该预签名 URL。

您可以使用下面的凭证创建预签名 URL：

- IAM 实例配置文件 – 有效期最长 6 小时。
- AWS Security Token Service – 使用永久凭证（例如，AWS 账户 根用户或 IAM 用户的凭证）签名时，有效期最长 36 小时。
- IAM 用户 – 使用 AWS 签名版本 4 时，有效期最长 7 天。

  要创建有效期最长 7 天的预签名 URL，请首先为所使用的开发工具包委托 IAM 用户凭证（访问密钥和秘密密钥）。然后使用 AWS 签名版本 4 生成预签名 URL。

> **Note**
>
> - 如果您已使用临时令牌创建了预签名 URL，则此 URL 将在令牌过期时过期，即使您使用更晚的到期时间创建了该 URL。
> - 由于预签名 URL 将向持有该 URL 的任何人授予访问 S3 on Outposts 存储桶的权限，我们建议您妥善保护它们。有关保护预签名 URL 的更多信息，请参阅 限制预签名 URL 功能。

S3 on Outposts 何时检查预签名 URL 的到期日期和时间？

在发出 HTTP 请求时，S3 on Outposts 会检查签名 URL 的到期日期和时间。例如，如果客户端刚好在到期时间之前开始下载某个大型文件，即使在下载过程中超过到期时间，下载也会继续进行。但如果连接断开，在客户端试图在超过到期时间后重新开始下载，则下载将会失败。

有关使用预签名 URL 共享或上传对象的更多信息，请参阅以下主题。

主题

- 使用预签名 URL 共享对象
- 生成预签名 URL 以将对象上传到 S3 on Outposts 存储桶

使用预签名 URL 共享对象

要授予对 Outpost 本地存储对象的限时访问权限而不更新存储桶策略，您可以使用预签名 URL。借助预签名 URL，作为存储桶的所有者，您可以与您虚拟私有云（VPC）中的个人共享对象，或者向其授予上传或删除对象的权限。

使用 AWS SDK 或 AWS Command Line Interface（AWS CLI）创建预签名 URL 时，您会将该 URL 与某个特定的操作关联。您还可以通过选择自定义到期时间来授予对预签名 URL 的限时访问权限，

自定义到期时间最短可为 1 秒，最长可为 7 天。共享预签名 URL 时，VPC 中的个人可以执行嵌入在 URL 中的操作，如同他们就是原始签名用户。URL 在到达其到期时间时将会过期，不再有效。

创建预签名 URL 时，必须提供您的安全凭证，然后指定以下内容：

- 该 Amazon S3 on Outposts 存储桶的一个访问点 Amazon 资源名称（ARN）
- 一个对象键
- 一个 HTTP方法（GET 用于下载对象）
- 一个到期日期和时间

预签名 URL 仅在指定的有效期内有效。也就是说，您必须在到期日期和时间到达之前启动该 URL 允许的操作。在到期日期和时间到达之前，您可以多次使用预签名 URL。如果您已使用临时令牌创建了预签名 URL，则此 URL 将在令牌过期时过期，即使您使用更晚的到期时间创建了该 URL。

虚拟私有云（VPC）中有权访问预签名 URL 的用户可以访问对象。例如，如果您在桶中具有一段视频，并且桶和对象均为私有，您可以通过生成预签名的 URL 来与其他用户共享视频。由于预签名 URL 将向持有该 URL 的任何人授予访问 S3 on Outposts 存储桶的权限，我们建议您妥善保护这些 URL。有关保护预签名 URL 的更多详细信息，请参阅 限制预签名 URL 功能。

具有有效安全凭证的任何人都可以创建预签名 URL。但必须由拥有执行预签名 URL 所基于的操作权限的人创建该预签名 URL。有关更多信息，请参阅 谁可以创建预签名 URL。

您可以使用 AWS SDK 和 AWS CLI 生成预签名 URL 以共享 S3 on Outposts 存储桶中的对象。有关更多信息，请参阅以下示例。

使用 AWS SDK

您可以使用 AWS SDK 生成可以提供给其他人的预签名 URL，以便他们可以检索对象。

> **Note**
>
> 使用 AWS SDK 生成预签名 URL 时，预签名 URL 的最长到期时间为创建之时起 7 天。

Java

Example

以下示例将生成一个预签名 URL，您可以将其提供给其他人，以便他们可以检索 S3 on Outposts 存储桶的对象。有关更多信息，请参阅 使用适用于 S3 on Outposts 的预签名 URL。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;

import java.io.IOException;
import java.net.URL;
import java.time.Instant;

public class GeneratePresignedURL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String accessPointArn = "*** access point ARN ***";
        String objectKey = "*** object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Set the presigned URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = Instant.now().toEpochMilli();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);

            // Generate the presigned URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest =
```

```
                    new GeneratePresignedUrlRequest(accessPointArn, objectKey)
                            .withMethod(HttpMethod.GET)
                            .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            System.out.println("Pre-Signed URL: " + url.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't
 process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example

以下示例将生成一个预签名 URL，您可以将其提供给其他人，以便他们可以检索 S3 on Outposts 存储桶的对象。有关更多信息，请参阅 使用适用于 S3 on Outposts 的预签名 URL。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;

namespace Amazon.DocSamples.S3
{
    class GenPresignedURLTest
    {
        private const string accessPointArn = "*** access point ARN ***";
        private const string objectKey = "*** object key ***";
        // Specify how long the presigned URL lasts, in hours.
        private const double timeoutDuration = 12;
        // Specify your bucket Region (an example Region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
```

```
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            string urlString = GeneratePreSignedURL(timeoutDuration);
        }
        static string GeneratePreSignedURL(double duration)
        {
            string urlString = "";
            try
            {
                GetPreSignedUrlRequest request1 = new GetPreSignedUrlRequest
                {
                    BucketName = accessPointArn,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration)
                };
                urlString = s3Client.GetPreSignedURL(request1);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            return urlString;
        }
    }
}
```

Python

使用适用于 Python (Boto3) 的 SDK 生成预签名 URL 以共享对象。例如，使用 Boto3 客户端和 `generate_presigned_url` 函数生成一个允许您 GET 对象的预签名 URL。

```
import boto3
    url = boto3.client('s3').generate_presigned_url(
    ClientMethod='get_object',
```

```
    Params={'Bucket': 'ACCESS_POINT_ARN', 'Key': 'OBJECT_KEY'},
    ExpiresIn=3600)
```

有关使用适用于 Python 的 SDK（Boto3）生成预签名 URL 的更多信息，请参阅《AWS SDK for Python (Boto) API 参考》中的 Python。

使用 AWS CLI

以下示例 AWS CLI 命令将为一个 S3 on Outposts 存储桶生成一个预签名 URL。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

> **Note**
>
> 使用 AWS CLI 生成预签名 URL 时，预签名 URL 的最长到期时间为创建之时起 7 天。

```
aws s3 presign s3://arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point/mydoc.txt --expires-in 604800
```

有关更多信息，请参阅《AWS CLI 命令参考》中的 presign。

生成预签名 URL 以将对象上传到 S3 on Outposts 存储桶

要授予对 Outpost 本地存储对象的限时访问权限而不更新存储桶策略，您可以使用预签名 URL。借助预签名 URL，作为存储桶的所有者，您可以与您虚拟私有云（VPC）中的个人共享对象，或者向其授予上传或删除对象的权限。

使用 AWS SDK 或 AWS Command Line Interface（AWS CLI）创建预签名 URL 时，您会将该 URL 与某个特定的操作关联。您还可以通过选择自定义到期时间来授予对预签名 URL 的限时访问权限，自定义到期时间最短可为 1 秒，最长可为 7 天。共享预签名 URL 时，VPC 中的个人可以执行嵌入在 URL 中的操作，如同他们就是原始签名用户。URL 在到达其到期时间时将会过期，不再有效。

创建预签名 URL 时，必须提供您的安全凭证，然后指定以下内容：

- 该 Amazon S3 on Outposts 存储桶的一个访问点 Amazon 资源名称（ARN）
- 一个对象键
- 一个 HTP 方法（PUT 用于上传对象）
- 一个到期日期和时间

预签名 URL 仅在指定的有效期内有效。也就是说，您必须在到期日期和时间到达之前启动该 URL 允许的操作。在到期日期和时间到达之前，您可以多次使用预签名 URL。如果您已使用临时令牌创建预签名 URL，则此 URL 将在令牌过期时过期，即使创建的 URL 的到期时间更晚也是如此。

如果预签名 URL 允许的操作由多个步骤构成（例如分段上传），则必须在到期时间前启动所有步骤。如果 S3 on Outposts 尝试使用某个已过期的 URL 启动某个步骤，您将会遇到错误。

虚拟私有云（VPC）中有权访问预签名 URL 的用户可上传对象。例如，VPC 中有权访问预签名 URL 的用户可以将对象上传到您的存储桶。由于预签名 URL 将向 VPC 中有权访问预签名 URL 的任何用户授予访问 S3 on Outposts 存储桶的权限，我们建议您妥善保护这些 URL。有关保护预签名 URL 的更多详细信息，请参阅 限制预签名 URL 功能。

具有有效安全凭证的任何人都可以创建预签名 URL。但必须由拥有执行预签名 URL 所基于的操作权限的人创建该预签名 URL。有关更多信息，请参阅 谁可以创建预签名 URL。

使用 AWS SDK 为 S3 on Outposts 对象操作生成预签名 URL

Java

适用于 Java 的 SDK 2.x

此示例演示了如何生成一个预签名 URL，以便在限定时间内用于将对象上传到某个 S3 on Outposts 存储桶。有关更多信息，请参阅 使用适用于 S3 on Outposts 的预签名 URL。

```
   public static void signBucket(S3Presigner presigner, String
outpostAccessPointArn, String keyName) {

       try {
           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(accessPointArn)
                   .key(keyName)
                   .contentType("text/plain")
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))
                   .putObjectRequest(objectRequest)
                   .build();

           PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);
```

```
            String myURL = presignedRequest.url().toString();
            System.out.println("Presigned URL to upload a file to: " +myURL);
            System.out.println("Which HTTP method must be used when uploading a
file: " +
                    presignedRequest.httpRequest().method());

            // Upload content to the S3 on Outposts bucket by using this URL.
            URL url = presignedRequest.url();

            // Create the connection and use it to upload the new object by using
the presigned URL.
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestProperty("Content-Type","text/plain");
            connection.setRequestMethod("PUT");
            OutputStreamWriter out = new
OutputStreamWriter(connection.getOutputStream());
            out.write("This text was uploaded as an object by using a presigned
URL.");
            out.close();

            connection.getResponseCode();
            System.out.println("HTTP response code is " +
connection.getResponseCode());

        } catch (S3Exception e) {
            e.getStackTrace();
        } catch (IOException e) {
            e.getStackTrace();
        }
    }
```

Python

适用于 Python 的 SDK（Boto3）

此示例演示了如何生成可在限定时间内执行某个 S3 on Outposts 操作的预签名 URL。有关更多信息，请参阅 使用适用于 S3 on Outposts 的预签名 URL。要使用该 URL 发出请求，请使用 `Requests` 程序包。

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)


def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
    """
    Generate a presigned S3 on Outposts URL that can be used to perform an
 action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds that the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method,
            Params=method_parameters,
            ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.",
 client_method)
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s')

    print('-'*88)
    print("Welcome to the Amazon S3 on Outposts presigned URL demo.")
    print('-'*88)
```

```
    parser = argparse.ArgumentParser()
    parser.add_argument('accessPointArn', help="The name of the S3 on Outposts
 access point ARN.")
    parser.add_argument(
        'key', help="For a GET operation, the key of the object in S3 on
 Outposts. For a "
                    "PUT operation, the name of a file to upload.")
    parser.add_argument(
        'action', choices=('get', 'put'), help="The action to perform.")
    args = parser.parse_args()

    s3_client = boto3.client('s3')
    client_action = 'get_object' if args.action == 'get' else 'put_object'
    url = generate_presigned_url(
        s3_client, client_action, {'Bucket': args.accessPointArn, 'Key':
 args.key}, 1000)

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == 'get':
        response = requests.get(url)
    elif args.action == 'put':
        print("Putting data to the URL.")
        try:
            with open(args.key, 'r') as object_file:
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(f"Couldn't find {args.key}. For a PUT operation, the key must
 be the "
                  f"name of a file that exists on your computer.")

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print('-'*88)


if __name__ == '__main__':
    usage_demo()
```

Amazon S3 on Outposts 与本地 Amazon EMR on Outposts

Amazon EMR 是一个托管式集群平台，可简化在 AWS 上运行大数据框架（如 Apache Hadoop 和 Apache Spark）以处理和分析海量数据的操作。通过使用这些框架和相关的开源项目，您可以处理用于分析目的的数据和业务情报工作负载。Amazon EMR 还可以协助您转换大量数据并将数据移入/移出其它 AWS 数据存储和数据库，并支持 Amazon S3 on Outposts。有关更多信息，请参阅《Amazon EMR 管理指南》中的 Amazon EMR on Outposts。

对于 Amazon S3 on Outposts，Amazon EMR 在版本 7.0.0 中开始支持 Apache Hadoop S3A 连接器。早期版本的 Amazon EMR 不支持本地 S3 on Outposts，也不支持 EMR 文件系统（EMRFS）。

受支持的应用程序

Amazon EMR 与 Amazon S3 on Outposts 结合使用可支持以下应用程序：

- Hadoop
- Spark
- Hue
- Hive
- Sqoop
- Pig
- Hudi
- Flink

有关更多信息，请参阅《Amazon EMR 版本指南》。

创建和配置 Amazon S3 on Outposts 桶

Amazon EMR 将 适用于 Java 的 AWS SDK 与 Amazon S3 on Outposts 结合使用来存储输入数据和输出数据。您的 Amazon EMR 日志文件存储在您选择的区域 Amazon S3 位置，而不是本地存储在 Outpost 上。有关更多信息，请参阅《Amazon EMR 管理指南》中的 Amazon EMR 日志。

为了符合 Amazon S3 和 DNS 的要求，S3 on Outposts 桶具有命名限制和局限性。有关更多信息，请参阅 创建 S3 on Outposts 存储桶。

在 Amazon EMR 版本 7.0.0 及更高版本中，您可以将 Amazon EMR 与 S3 on Outposts 和 S3A 文件系统结合使用。

先决条件

S3 on Outposts 权限 – 当您创建 Amazon EMR 实例配置文件时，您的角色必须包含 S3 on Outposts 的 AWS Identity and Access Management（IAM）命名空间。S3 on Outposts 具有自己的命名空间 `s3-outposts*`。有关使用此命名空间的示例策略，请参阅使用 S3 on Outposts 设置 IAM。

S3A 连接器 – 要将您的 EMR 集群配置为访问 Amazon S3 on Outposts 桶中的数据，您必须使用 Apache Hadoop S3A 连接器。要使用该连接器，请确保所有 S3 URI 都使用 `s3a` 方案。如果不是这样，您可以配置用于 EMR 集群的文件系统实现，以便 S3 URI 可以与 S3A 连接器结合使用。

要将文件系统实现配置为与 S3A 连接器结合使用，请为 EMR 集群使用 `fs.`*`file_scheme`*`.impl` 和 `fs.AbstractFileSystem.`*`file_scheme`*`.impl` 配置属性，其中 *`file_scheme`* 对应于您拥有的 S3 URI 类型。要使用以下示例，请将 *`user input placeholders`* 替换为您自己的信息。例如，要更改使用 `s3` 方案的 S3 URI 的文件系统实现，请指定以下集群配置属性：

```
[
  {
"Classification": "core-site",
    "Properties": {
    "fs.s3.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
    "fs.AbstractFileSystem.s3.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
  }
]
```

要使用 S3A，请将 `fs.`*`file_scheme`*`.impl` 配置属性设置为 `org.apache.hadoop.fs.s3a.S3AFileSystem`，并将 `fs.AbstractFileSystem.`*`file_scheme`*`.impl` 属性设置为 `org.apache.hadoop.fs.s3a.S3A`。

例如，如果您要访问路径 `s3a://`*`bucket`*`/...`，请将 `fs.s3a.impl` 属性设置为 `org.apache.hadoop.fs.s3a.S3AFileSystem`，然后将 `fs.AbstractFileSystem.s3a.impl` 属性设置为 `org.apache.hadoop.fs.s3a.S3A`。

开始将 Amazon EMR 与 Amazon S3 on Outposts 结合使用

以下主题介绍了如何开始将 Amazon EMR 与 Amazon S3 on Outposts 结合使用。

主题

- 创建权限策略
- 创建和配置集群
- 配置概述
- 注意事项

创建权限策略

在创建使用 Amazon S3 on Outposts 的 EMR 集群之前，您必须创建一个 IAM policy 以附加到该集群的 Amazon EC2 实例配置文件。该策略必须具有访问 S3 on Outposts 接入点 Amazon 资源名称（ARN）的权限。有关为 S3 on Outposts 创建 IAM policy 的更多信息，请参阅使用 S3 on Outposts 设置 IAM。

以下示例策略显示如何授予所需的权限。创建策略后，将策略附加到您用于创建 EMR 集群的实例配置文件角色，如 the section called “创建和配置集群” 部分所述。要使用此示例，请将 *user input placeholders* 替换为您自己的信息。

```
{
"Version":"2012-10-17",
  "Statement": [
        {
  "Effect": "Allow",
            "Resource": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name,
            "Action": [
                "s3-outposts:*"
            ]
        }
    ]

 }
```

创建和配置集群

要创建在 S3 on Outposts 中运行 Spark 的集群，请在控制台中完成以下步骤。

创建在 S3 on Outposts 中运行 Spark 的集群

1. 通过 https://console.aws.amazon.com/elasticmapreduce/ 打开 Amazon EMR 控制台。
2. 在左侧导航窗格中，选择集群。

3. 选择创建集群。

4. 对于 Amazon EMR 版本，请选择 emr-7.0.0 或更高版本。

5. 对于应用程序捆绑包，请选择 Spark 交互式。然后，选择要包含在集群中的任何其它受支持的应用程序。

6. 要启用 Amazon S3 on Outposts，请输入您的配置设置。

   示例配置设置

   要使用以下示例配置设置，请将 *user input placeholders* 替换为您自己的信息。

```
[
 {
   "Classification": "core-site",
   "Properties": {
     "fs.s3a.bucket.DOC-EXAMPLE-BUCKET.accesspoint.arn": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name"
     "fs.s3a.committer.name": "magic",
     "fs.s3a.select.enabled": "false"
    }
  },
  {
    "Classification": "hadoop-env",
    "Configurations": [
      {
        "Classification": "export",
        "Properties": {
          "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
      }
     ],
     "Properties": {}
   },
   {
     "Classification": "spark-env",
     "Configurations": [
       {
         "Classification": "export",
         "Properties": {
           "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
         }
       }
```

```
      ],
      "Properties": {}
     },
     {
      "Classification": "spark-defaults",
      "Properties": {
        "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-11-amazon-
corretto.x86_64",
        "spark.sql.sources.fastS3PartitionDiscovery.enabled": "false"
      }
     }
  ]
```

7. 在联网部分中，选择您的 AWS Outposts 机架上的虚拟私有云（VPC）和子网。有关 Amazon EMR on Outposts 的更多信息，请参阅《Amazon EMR 管理指南》中的 AWS Outposts 上的 EMR 集群。
8. 在 Amazon EMR 的 EC2 实例配置文件部分，选择附加了您之前创建的权限策略的 IAM 角色。
9. 配置剩余的集群设置，然后选择创建集群。

配置概述

下表描述了 S3A 配置，以及在设置将 S3 on Outposts 与 Amazon EMR 结合使用的集群时要为这些配置的参数指定的值。

参数	默认值	S3 on Outposts 的必需值	说明
`fs.s3a.aws.credentials.provider`	如果未指定，S3A 将在区域桶中查找具有 Outposts 桶名称的 S3。	S3 on Outposts 桶的接入点 ARN	Amazon S3 on Outposts 支持将纯 Virtual Private Cloud (VPC) 接入点作为访问 Outposts 存储桶的唯一方式。
`fs.s3a.committer.name`	`file`	`magic`	Magic 提交程序是 S3 on Outposts 唯一支持的提交程序。

参数	默认值	S3 on Outposts 的必需值	说明
fs.s3a.select.enabled	TRUE	FALSE	Outposts 上不支持 S3 Select。
JAVA_HOME	/usr/lib/jvm/java-8	/usr/lib/jvm/java-11-amazon-corretto.x86_64	S3A 上的 S3 on Outposts 需要 Java 版本 11。

下表描述了 Spark 配置，以及在设置将 S3 on Outposts 与 Amazon EMR 结合使用的集群时要为这些配置的参数指定的值。

参数	默认值	S3 on Outposts 的必需值	说明
spark.sql.sources.fastS3PartitionDiscovery.enabled	TRUE	FALSE	S3 on Outposts 不支持快速分区。
spark.executorEnv.JAVA_HOME	/usr/lib/jvm/java-8	/usr/lib/jvm/java-11-amazon-corretto.x86_64	S3A 上的 S3 on Outposts 需要 Java 版本 11。

注意事项

当您将 Amazon EMR 与 S3 on Outposts 桶集成时，请考虑以下几点：

- Amazon EMR 版本 7.0.0 及更高版本支持 Amazon S3 on Outposts。
- 将 S3 on Outposts 与 Amazon EMR 结合使用时需要 S3A 连接器。只有 S3A 具有与 S3 on Outposts 桶交互所需的功能。有关 S3A 连接器设置的信息，请参阅先决条件。

- Amazon S3 on Outposts 对于 Amazon EMR 仅支持采用 Amazon S3 托管式密钥的服务器端加密（SSE-S3）。有关更多信息，请参阅 the section called “数据加密”。
- Amazon S3 on Outposts 不支持使用 S3A FileOutputCommitter 进行写入。在 S3 on Outposts 桶上使用 S3A FileOutputCommitter 进行写入会导致以下错误：InvalidStorageClass：您指定的存储类无效。
- Amazon EMR Serverless 或 Amazon EMR on EKS 不支持 Amazon S3 on Outposts。
- Amazon EMR 日志存储在您选择的区域 Amazon S3 位置，而不是本地存储在 S3 on Outposts 桶中。

授权和身份验证缓存

S3 on Outposts 可在 Outposts 机架上安全地在本地缓存身份验证和授权数据。该缓存会针对每个请求，删除到父 AWS 区域的往返行程。这能够消除网络往返带来的可变性。借助 S3 on Outposts 中的身份验证和授权缓存功能，您可以获得与 Outposts 和AWS 区域 之间的连接延迟无关的一致延迟。

当您发出 S3 on Outposts API 请求时，身份验证和授权数据会被安全地缓存。然后，缓存的数据可用于对后续 S3 对象 API 请求进行身份验证。在使用签名版本 4A（SigV4A）对请求进行签名时，S3 on Outposts 仅缓存身份验证和授权数据。缓存本地存储在 S3 on Outposts 服务中的 Outposts 上。当您发出 S3 API 请求时，它会异步刷新。缓存已加密，Outposts 上不会存储任何纯文本加密密钥。

当 Outpost 连接到 AWS 区域时，缓存的有效期最长为 10 分钟。当您发出 S3 on Outposts API 请求时，它会异步刷新，以确保使用最新策略。如果 Outpost 与 AWS 区域断开连接，则缓存的有效期最长为 12 小时。

配置授权和身份验证缓存

S3 on Outposts 会针对使用 Sigv4a 算法进行签名的请求，自动缓存身份验证和授权数据。有关更多信息，请参阅《AWS Identity and Access Management 用户指南》中的对 AWS API 请求进行签名。在最新版本的 AWS SDK 中提供了 SigV4A 算法。您可以通过依赖 AWS 公共运行时（CRT）库来获取它。

您需要使用最新版本的 AWS SDK 并安装最新版本的 CRT。例如，您可以通过运行 `pip install awscrt` 使用 Boto3 来获取最新版本的 CRT。

S3 on Outposts 不会针对使用 SigV4 算法进行签名的请求，缓存身份验证和授权数据。

验证 SigV4A 签名

您可以使用 AWS CloudTrail 来验证是否已使用 SigV4A 对请求进行了签名。有关为 S3 on Outposts 设置 CloudTrail 的更多信息，请参阅使用 AWS CloudTrail 日志监控 S3 on Outposts。

配置 CloudTrail 后，您可以在 CloudTrail 日志的 `SignatureVersion` 字段中验证是否已对请求进行了签名。使用 SigV4A 进行了签名的请求的 `SignatureVersion` 将设置为 `AWS4-ECDSA-P256-SHA256`。使用 SigV4 进行了签名的请求的 `SignatureVersion` 将设置为 `AWS4-HMAC-SHA256`。

S3 on Outposts 中的安全性

AWS 十分重视云安全性。为了满足对安全性最敏感的组织的需求，我们打造了具有超高安全性的数据中心和网络架构。作为 AWS 的客户，您也可以从这些数据中心和网络架构受益。

安全性是 AWS 和您的共同责任。责任共担模式将其描述为云的安全性和云中的安全性：

- 云的安全性——AWS 负责保护在 AWS Cloud 中运行 AWS 服务 的基础设施。AWS 还提供可安全使用的服务。第三方审核员定期测试和验证我们的安全性的有效性，作为 AWS 合规性计划的一部分。要了解适用于 Amazon S3 on Outposts 的合规性计划，请参阅合规性计划范围内的 AWS 服务。
- 云中的安全性 – 您的责任由您使用的 AWS 服务 决定。您还需要对其它因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

此文档将帮助您了解如何在使用 S3 on Outposts 时应用责任共担模式。以下主题显示了如何配置 S3 on Outposts 以实现您的安全性和合规性目标。您还会了解如何使用其他 AWS 服务以帮助您监控和保护 S3 on Outposts 资源。

主题

使用 S3 on Outposts 设置 IAM

AWS Identity and Access Management（IAM）是一项，AWS 服务可以帮助管理员安全地控制对 AWS 资源的访问。IAM 管理员控制谁可以通过身份验证（登录）和获得授权（具有权限）来使用 Amazon S3 on Outposts 资源。IAM 是一项无需额外费用即可使用的。AWS 服务默认情况下，用户对 S3 on Outposts 资源和操作没有权限。要授予对 S3 on Outposts 资源和 API 操作的访问权限，您可以使用 IAM 创建用户、组或角色并附加权限。

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。

- 通过身份提供者在 IAM 中托管的用户：

  创建适用于身份联合验证的角色。按照《IAM 用户指南》中针对第三方身份提供者创建角色（联合身份验证）的说明进行操作。

- IAM 用户：
  - 创建您的用户可以担任的角色。按照《IAM 用户指南》中为 IAM 用户创建角色的说明进行操作。
  - （不推荐使用）将策略直接附加到用户或将用户添加到用户组。按照《IAM 用户指南》中向用户添加权限（控制台）中的说明进行操作。

除基于 IAM 身份的策略外，S3 on Outposts 还同时支持桶和接入点策略。存储桶策略和访问点策略是附加到 S3 on Outposts 资源的基于资源的策略。

- 存储桶策略用于附加到存储桶，它根据策略中的元素允许或拒绝对存储桶和存储桶中对象的请求。
- 而访问点策略用于附加到访问点，允许或拒绝对访问点的请求。

访问点策略可与附加到底层 S3 on Outposts 存储桶的存储桶策略结合使用。要使应用程序或用户能够通过某个 S3 on Outposts 访问点访问某个 S3 on Outposts 桶中的对象，访问点策略和桶策略都必须允许该请求。

您在接入点策略中包括的限制仅适用于通过该接入点发出的请求。例如，如果将某个访问点附加到某个桶，则无法使用访问点策略来允许或拒绝直接对该桶发出的请求。但是，应用到存储桶策略的限制可能会允许或拒绝直接对该存储桶或通过该访问点发出的请求。

在 IAM policy 或基于资源的策略中，您可以定义要允许或拒绝的 S3 on Outposts 操作。S3 on Outposts 操作对应于特定的 S3 on Outposts API 操作。S3 on Outposts 操作使用 `s3-outposts:` 命名空间前缀。对 AWS 区域中的 Amazon S3 on Outposts 控制 API 发出的请求，以及对 Outpost 上的对象 API 端点发出的请求，都将使用 IAM 进行身份认证并根据 `s3-outposts:` 命名空间前缀进行授权。要使用 S3 on Outposts，请配置您的 IAM 用户并针对 `s3-outposts:` IAM 命名空间对其进行授权。

有关更多信息，请参阅《服务授权参考》中的 Amazon S3 on Outposts 的操作、资源和条件键。

> **Note**
>
> - S3 on Outposts 不支持访问控制列表 (ACL)。
> - S3 on Outposts 默认将存储桶拥有者视为对象所有者，以帮助确保存储桶的拥有者不会被阻止访问或删除对象。
> - S3 on Outposts 始终启用“S3 阻止公有访问”，以帮助确保对象从不会具有公有访问权限。

有关为 S3 on Outposts 设置 IAM 的更多信息，请参阅以下主题。

主题

S3 on Outposts 策略的主体

创建基于资源的策略以授予对 S3 on Outposts 存储桶的访问权限时，您必须使用 `Principal` 元素来指定可请求对该资源执行某个操作或运算的人员或应用程序。对于 S3 on Outposts 策略，您可以使用下面的一个主体：

- 一个 AWS 账户
- IAM 用户
- IAM 角色
- 所有主体，通过在使用 `Condition` 元素将访问范围限定为特定 IP 范围的策略中指定通配符 (*) 来确定

> **Important**
>
> 您无法为 S3 on Outposts 桶编写在 * 元素中使用通配符（`Principal`）的策略，除非该策略还包含一个将访问范围限制为特定 IP 地址范围的 `Condition`。此限制有助于确保您的 S3 on Outposts 桶不会被公开访问。有关示例，请参阅S3 on Outposts 的示例策略。

有关 Principal 元素的更多信息，请参阅《IAM 用户指南》中的 AWS JSON 策略元素：Principal。

S3 on Outposts 的资源 ARN

S3 on Outposts 的 Amazon 资源名称 (ARN) 除了包含 Outpost 归属的 AWS 区域、AWS 账户 ID 和资源名称外，还包含 Outpost ID。要访问 Outposts 存储桶和对象并对它们执行操作，您必须使用下表中显示的一种 ARN 格式。

ARN 中的 *partition* 值是指一组 AWS 区域。每个 AWS 账户 的作用域为一个分区。以下是支持的分区：

- aws – AWS 区域
- aws-us-gov – AWS GovCloud (US) 区域

下表显示了 S3 on Outposts ARN 格式。

Amazon S3 on Outposts ARN	ARN 格式	示例
存储桶 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /bucket/*bucket_name*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /bucket/*amzn-s3-demo-bucket1*
访问点 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /accesspoint/ *accesspoint_name*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name*
对象 ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* :outpost/ *op-01ac5d*

Amazon S3 on Outposts ARN	ARN 格式	示例
	bucket/*bucket_name* / object/*object_key*	*28a6a232904* / bucket/*amzn-s3-demo-bucket1* /object/*myobject*
S3 on Outposts 访问点对象 ARN（在策略中使用）	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id* /accesspoint/ *accesspoint_name* / object/*object_key*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904* /accesspoint/ *access-point-name/object/myobject*
S3 on Outposts ARN	arn:*partition* :s3-outposts: *region*: *account_id* :outpost/ *outpost_id*	arn:*aws*:s3-outposts: *us-west-2* :*123456789012* : outpost/ *op-01ac5d28a6a232904*

S3 on Outposts 的示例策略

Example：具有 AWS 账户主体的 S3 on Outposts 桶策略

以下存储桶策略使用一个 AWS 账户主体授予对某个 S3 on Outposts 存储桶的访问权限。要使用这一桶策略，请将 *user input placeholders* 替换为您自己的信息。

Example：使用通配符主体 (*) 和条件键将访问范围限定为特定 IP 地址范围的 S3 on Outposts 桶策略

以下桶策略使用通配符主体 (*) 和 aws:SourceIp 条件将访问范围限定为特定的 IP 地址范围。要使用这一桶策略，请将 *user input placeholders* 替换为您自己的信息。

S3 on Outposts 端点的权限

S3 on Outposts 需要 IAM 中它自己的权限，来管理 S3 on Outposts 端点操作。

> **Note**
>
> - 对于使用客户拥有的 IP 地址池（CoIP 池）访问类型的端点，您还必须具有使用 CoIP 池中 IP 地址的权限，如下表所述。
> - 对于使用 AWS Resource Access Manager 访问 S3 on Outposts 的共享账户，这些共享账户中的用户无法在共享子网上创建其自己的端点。如果共享账户中的用户想管理自己的端点，则共享账户必须在 Outpost 上创建自己的子网。有关更多信息，请参阅 the section called “共享 S3 on Outposts”。

下表显示了 S3 on Outposts 端点相关的 IAM 权限。

Action	IAM 权限

`CreateEndpoint`	`s3-outposts:CreateEndpoint`
	`ec2:CreateNetworkInterface`
	`ec2:DescribeNetworkInterfaces`
	`ec2:DescribeVpcs`
	`ec2:DescribeSecurityGroups`
	`ec2:DescribeSubnets`
	`ec2:CreateTags`
	`iam:CreateServiceLinkedRole`
	对于使用本地客户拥有的 IP 地址池（CoIP 池）访问类型的端点，将需要以下额外权限：
	`s3-outposts:CreateEndpoint`
	`ec2:DescribeCoipPools`
	`ec2:GetCoipPoolUsage`
	`ec2:AllocateAddress`

Action	IAM 权限

	`ec2:AssociateAddress`
	
	`ec2:DescribeAddresses`
	
	`ec2:DescribeLocalGatewayRouteTableVpcAssociations`

`DeleteEndpoint`	`s3-outposts:DeleteEndpoint`
	
	`ec2:DeleteNetworkInterface`
	
	`ec2:DescribeNetworkInterfaces`
	
	对于使用本地客户拥有的 IP 地址池（CoIP 池）访问类型的端点，将需要以下额外权限：
	
	`s3-outposts:DeleteEndpoint`
	
	`ec2:DisassociateAddress`
	
	`ec2:DescribeAddresses`
	
	`ec2:ReleaseAddress`

`ListEndpoints`	`s3-outposts:ListEndpoints`

> **Note**
>
> 您可以使用 IAM policy 中的资源标签来管理权限。

S3 on Outposts 的服务相关角色

S3 on Outposts 使用 IAM 服务相关角色代表您创建一些网络资源。有关更多信息，请参阅 将服务相关角色用于 Amazon S3 on Outposts。

S3 on Outposts 中的数据加密

原定设置情况下，存储在 Amazon S3 on Outposts 中的所有数据都使用带 Amazon S3 托管式加密密钥 (SSE-S3) 的服务器端加密进行加密。有关更多信息，请参阅《Amazon S3 用户指南》中的使用具有 Amazon S3 托管式密钥的服务器端加密（SSE-S3）。

您可以选择使用带客户提供的加密密钥的服务器端加密 (SSE-C)。要使用 SSE-C，请在对象 API 请求中指定加密密钥。服务器端加密仅加密对象数据而非加密对象元数据。有关更多信息，请参阅《Amazon S3 用户指南》中的使用具有客户提供的密钥的服务器端加密。

> **Note**
>
> S3 on Outposts 不支持带 AWS Key Management Service (AWS KMS) 密钥的服务器端加密 (SSE-KMS)。

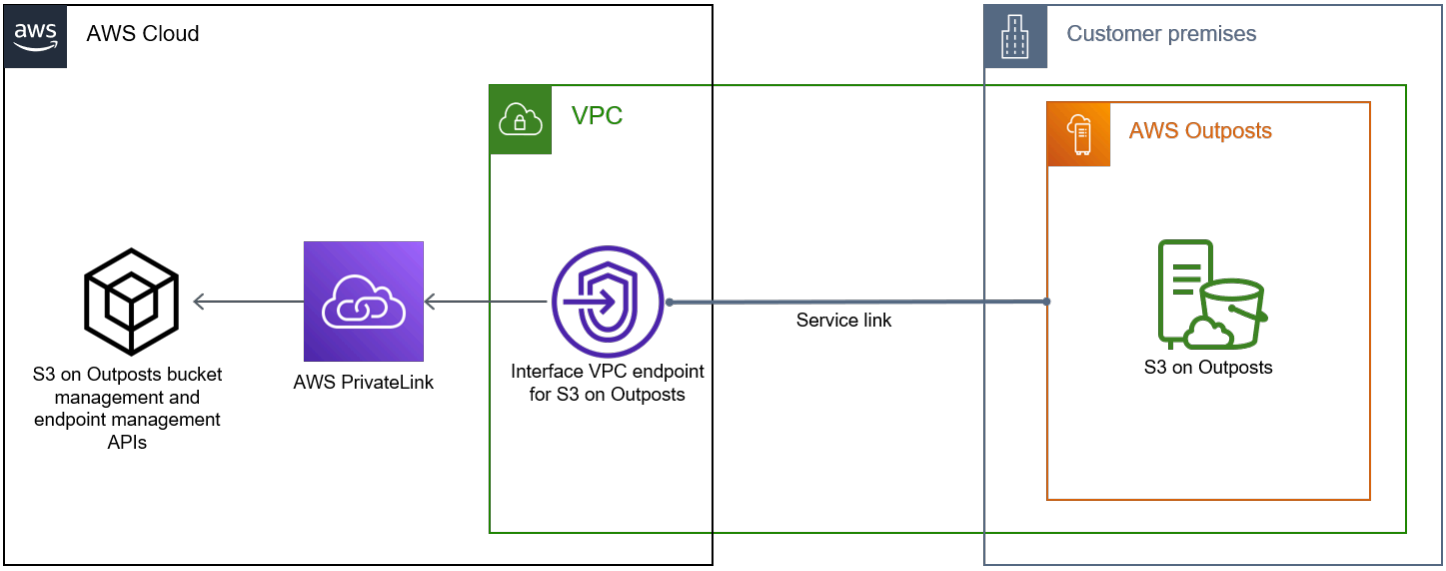
S3 on Outposts 的 AWS PrivateLink

S3 on Outposts 支持 AWS PrivateLink，后者通过虚拟专用网络中的私有端点，提供对 S3 on Outposts 存储的直接管理访问。这样，您就可以使用虚拟私有云（VPC）中的私有 IP 地址，简化内部网络架构并对 Outposts 对象存储执行管理操作。使用 AWS PrivateLink，无需使用公有 IP 地址或代理服务器。

通过将 AWS PrivateLink 用于 Amazon S3 on Outposts，您可以在虚拟私有云（VPC）中预调配接口 VPC 端点，以访问 S3 on Outposts 存储桶管理和端点管理 API。通过虚拟专用网络（VPN）或 AWS Direct Connect，可以直接从部署在 VPC 中或本地的应用程序访问接口 VPC 端点。您可以通过 AWS PrivateLink 访问桶和端点管理 API。AWS PrivateLink 不支持数据传输 API 操作，例如 GET、PUT 和类似的 API。这些操作已经通过 S3 on Outposts 端点和接入点配置私密传输。有关更多信息，请参阅 S3 on Outposts 的网络。

接口端点由一个或多个弹性网络接口 (ENI) 代表，这些接口是从 VPC 中的子网分配的私有 IP 地址。向 S3 on Outposts 的接口端点发出的请求将自动路由到 AWS 网络上的 S3 on Outposts 桶和端点管理 API。您还可以通过 AWS Direct Connect 或 AWS Virtual Private Network (Site-to-Site VPN) 从本地部署应用程序访问 VPC 中的接口端点。有关如何将 VPC 与本地网络连接的更多信息，请参阅 Direct Connect 用户指南和 AWS Site-to-Site VPN 用户指南。

接口端点通过 AWS 网络和通过 AWS PrivateLink 路由对 S3 on Outposts 桶和端点管理 API 的请求，如下图所示。



有关接口端点的一般信息，请参阅 AWS PrivateLink 指南中的接口 VPC 端点 (AWS PrivateLink)。

主题

- 限制和局限性
- 访问 S3 on Outposts 接口端点
- 更新本地 DNS 配置
- 为 S3 on Outposts 创建 VPC 端点
- 为 S3 on Outposts 创建桶策略和 VPC 端点策略

限制和局限性

当您通过 AWS PrivateLink 访问 S3 on Outposts 桶和端点管理 API 时，将适用 VPC 限制。有关更多信息，请参阅 AWS PrivateLink 指南中的接口端点属性和限制以及 AWS PrivateLink 配额。

此外，AWS PrivateLink 不支持以下内容：

- 美国联邦信息处理标准 (FIPS) 端点
- S3 on Outposts 数据传输 API，例如，GET、PUT 和类似的对象 API 操作。
- 私有 DNS

访问 S3 on Outposts 接口端点

要使用 AWS PrivateLink 访问 S3 on Outposts 桶和端点管理 API，您必须更新应用程序以使用特定于端点的 DNS 名称。创建接口端点时，AWS PrivateLink 会生成两种特定于端点的 S3 on Outposts 名称：区域 和可用区。

- 区域 DNS 名称 – 包括唯一的 VPC 端点 ID、服务标识符、AWS 区域和 `vpce.amazonaws.com`，例如 *`vpce-1a2b3c4d-5e6f`*`.s3-outposts.us-east-1.vpce.amazonaws.com`。
- 可用区 DNS 名称 – 包括唯一的 VPC 端点 ID、可用区、服务标识符、AWS 区域和 `vpce.amazonaws.com`，例如 *`vpce-1a2b3c4d-5e6f`*`-us-east-1a.s3-outposts.us-east-1.vpce.amazonaws.com`。如果您的架构隔离了可用区，则可以使用此选项。例如，您可以将可用区 DNS 名称用于故障控制或降低区域数据传输成本。

> **Important**
>
> S3 on Outposts 接口端点是从公有 DNS 域解析出来的。S3 on Outposts 不支持私有 DNS。将 `--endpoint-url` 参数用于所有桶和端点管理 API。

AWS CLI 示例

使用 `--region` 和 `--endpoint-url` 参数，通过 S3 on Outposts 接口端点访问桶管理和端点管理 API。

Example：使用端点 URL 列出具有 S3 控制 API 的桶

在以下示例中，将区域 *`us-east-1`*、VPC 端点 URL *`vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com`* 和账户 ID *`111122223333`* 替换为相应的信息。

```
aws s3control list-regional-buckets --region us-east-1 --endpoint-url
 https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com --account-
id 111122223333
```

AWS SDK 示例

将 SDK 更新到最新版本，然后将客户端配置为使用端点 URL 访问 S3 on Outposts 接口端点的 S3 控制 API。

SDK for Python (Boto3)

Example：使用端点 URL 访问 S3 控制 API

在以下示例中，将区域 *us-east-1* 和 VPC 端点 URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* 替换为相应的信息。

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com'
)
```

有关更多信息，请参阅《Boto3 开发人员指南》中的适用于 Amazon S3 的 AWS PrivateLink。

SDK for Java 2.x

Example：使用端点 URL 访问 S3 控制 API

在以下示例中，将 VPC 端点 URL *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* 和区域 *Region.US_EAST_1* 替换为相应的信息。

```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com"))
                                  .build()
```

有关更多信息，请参阅《适用于 Java 的 AWS SDK API Reference》中的 S3ControlClient。

更新本地 DNS 配置

使用特定于端点的 DNS 名称访问适用于 S3 on Outposts 桶管理和端点管理 API 的接口端点时，您不必更新本地 DNS 解析程序。您可以使用来自公有 S3 on Outposts DNS 域的接口端点的私有 IP 地址解析特定于端点的 DNS 名称。

为 S3 on Outposts 创建 VPC 端点

要为 S3 on Outposts 创建 VPC 接口端点，请参阅《AWS PrivateLink 指南》中的创建 VPC 端点。

为 S3 on Outposts 创建桶策略和 VPC 端点策略

您可以向 VPC 端点附加用于控制对 S3 on Outposts 的访问的端点策略。您还可以在 S3 on Outposts 桶策略中使用 `aws:sourceVpce` 条件来限制从特定 VPC 端点访问特定的桶。使用 VPC 端点策略，您可以控制对 S3 on Outposts 桶管理 API 和端点管理 API 的访问。使用桶策略，您可以控制对 S3 on Outposts 桶管理 API 的访问。但是，您无法使用 `aws:sourceVpce` 管理对 S3 on Outposts 的对象操作的访问。

S3 on Outposts 的访问策略指定以下信息：

- 允许或拒绝其操作的 AWS Identity and Access Management (IAM) 主体。
- 允许或拒绝的 S3 控制操作。
- 允许或拒绝其操作的 S3 on Outposts 资源。

以下示例显示了限制对桶或端点的访问的策略。有关 VPC 连接的更多信息，请参阅 AWS 白皮书 Amazon Virtual Private Cloud 连接性选项中的 Network-to-VPC connectivity options（从网络到 VPC 的连接性选项）。

> **Important**
>
> - 当您如本节所述对 VPC 端点应用示例策略时，您可能会无意中阻止对桶的访问权限。桶权限会限制桶访问源自 VPC 端点的连接，而这可能会阻止到桶的所有连接。有关如何修复此问题的信息，请参阅我的桶策略有错误的 VPC 或 VPC 端点 ID。支持 知识中心内的如何修复策略才能访问桶？。
> - 在使用以下示例桶策略之前，请将 VPC 端点 ID 替换为适合您的使用案例的值。否则，您将无法访问您的桶。
> - 如果您的策略仅允许从特定 VPC 端点访问 S3 on Outposts 桶，它将禁用通过控制台访问该桶，因为控制台请求不是源自指定的 VPC 端点。

主题

示例：限制从 VPC 端点对特定桶的访问

您可以创建一个端点策略以仅允许访问特定的 S3 on Outposts 桶。以下策略将 GetBucketPolicy 操作的访问权限仅限于 *example-outpost-bucket*。要使用此策略，请将示例值替换为您自己的值。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "Policy1415115909151",
    "Statement": [
        {
            "Sid": "Access-to-specific-bucket-only",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": "s3-outposts:GetBucketPolicy",
            "Effect": "Allow",
            "Resource": "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket"
        }
    ]
}
```

示例：在 S3 on Outposts 桶策略中拒绝从特定 VPC 端点访问

以下 S3 on Outposts 桶策略拒绝通过 *vpce-1a2b3c4d* VPC 端点访问 *example-outpost-bucket* 桶上的 GetBucketPolicy。

`aws:sourceVpce` 条件指定端点，不需要 VPC 端点资源的 Amazon 资源名称 (ARN)，只需要端点 ID。要使用此策略，请将示例值替换为您自己的值。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "Policy1415115909152",
    "Statement": [
        {
            "Sid": "Deny-access-to-specific-VPCE",
```

```
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": "s3-outposts:GetBucketPolicy",
            "Effect": "Deny",
            "Resource": "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpce": "vpce-1a2b3c4d"
                }
            }
        }
    ]
}
```

AWS 签名版本 4（SigV4）身份认证特定的策略键

下表显示了与 AWS 签名版本 4（SigV4）身份认证有关的条件键，您可以将这些条件键用于 Amazon S3 on Outposts。您可以在存储桶策略中添加这些条件，以便在使用签名版本 4 对请求进行身份认证时强制执行特定的行为。有关示例策略，请参阅 使用与签名版本 4 有关的条件键的存储桶策略示例。有关使用签名版本 4 对请求进行身份验证的更多信息，请参阅《Amazon Simple Storage Service API 参考》中的对请求进行身份验证（AWS 签名版本 4）

适用的键	说明
s3-outposts:authType	S3 on Outposts 支持多种身份认证方法。要限定传入的请求使用特定的身份认证方法，您可以使用此可选条件键。例如，您可以使用此条件键以仅允许在请求身份认证中使用 HTTP Authorization 标头。 有效值： REST-HEADER REST-QUERY-STRING
s3-outposts:signatureAge	签名在已通过身份认证的请求中的有效期（以毫秒为单位）。 此条件仅适用于预签名 URL。

适用的键	说明
	在签名版本 4 中，签名键的有效期最长为 7 天。因此，签名的有效期最初也为 7 天。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的签名请求简介。您可以使用此条件进一步限制签名有效期。 示例值：600000
s3-outposts:x-amz-content-sha256	您可以使用此条件键禁止存储桶中未签名的内容。 使用签名版本 4 时，对于使用 Authorization 标头的请求，您需要在签名计算中添加 x-amz-content-sha256 标头，然后将它的值设置为哈希负载。 您可以在存储桶策略中使用此条件键来拒绝上传任何未签名的有效负载。例如： • 拒绝使用 Authorization 标头对请求进行身份认证，但未对有效负载进行签名的上传。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的在单个区块中传输有效载荷。 • 拒绝使用预签名 URL 的上传。预签名 URL 始终有一个 UNSIGNED_PAYLOAD 。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的对请求进行身份验证和身份验证方法。 有效值：UNSIGNED-PAYLOAD

使用与签名版本 4 有关的条件键的存储桶策略示例

要使用以下示例，请将 *user input placeholders* 替换为您自己的信息。

Example: s3-outposts:signatureAge

以下存储桶策略将在签名的存在时间超过 10 分钟时拒绝对 example-outpost-bucket 中对象的 S3 on Outposts 预签名 URL 请求。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/
object/*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Example: s3-outposts:authType

以下存储桶策略仅允许使用 Authorization 标头进行请求身份认证的请求。任何预签名的 URL 请求都将被拒绝，因为预签名 URL 使用查询参数来提供请求和身份认证信息。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的身份验证方法。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
         {
               "Sid": "Allow only requests that use the Authorization header for
 request authentication. Deny presigned URL requests.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/
object/*",
               "Condition": {
                     "StringNotEquals": {
```

```
                        "s3-outposts:authType": "REST-HEADER"
                    }
                }
        }
    ]
}
```

Example: s3-outposts:x-amz-content-sha256

以下存储桶策略拒绝任何带有未签名有效负载的上传，例如使用预签名 URL 的上传。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的对请求进行身份认证和身份认证方法。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
         {
               "Sid": "Deny uploads with unsigned payloads.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/
object/*",
               "Condition": {
                     "StringEquals": {
                           "s3-outposts:x-amz-content-sha256": "UNSIGNED-PAYLOAD"
                     }
               }
         }
   ]
}
```

适用于 Amazon S3 on Outposts 的 AWS 托管式策略

AWS 托管式策略是由 AWS 创建和管理的独立策略。AWS 托管式策略旨在为许多常见使用案例提供权限，以便您可以开始为用户、组和角色分配权限。

请记住，AWS 托管式策略可能不会为您的特定使用案例授予最低权限，因为它们可供所有 AWS 客户使用。我们建议通过定义特定于使用案例的客户管理型策略来进一步减少权限。

您无法更改 AWS 托管式策略中定义的权限。如果 AWS 更新在 AWS 托管式策略中定义的权限，则更新会影响该策略所附加到的所有主体身份（用户、组和角色）。当新的 AWS 服务 启动或新的 API 操作可用于现有服务时，AWS 最有可能更新 AWS 托管式策略。

有关更多信息，请参阅AWS《IAM 用户指南》中的托管策略。

AWS 托管式策略：AWSS3OnOutpostsServiceRolePolicy

作为服务相关角色 `AWSServiceRoleForS3OnOutposts` 的一部分，帮助您管理网络资源。

要查看此策略的权限，请参阅 AWSS3OnOutpostsServiceRolePolicy。

AWS 托管式策略的 S3 on Outposts 更新

查看自 S3 on Outposts 开始跟踪更改以来，有关此服务的 AWS 托管式策略的更新的详细信息。

更改	描述	日期
S3 on Outposts 添加了 `AWSS3OnOutpostsServiceRolePolicy`	S3 on Outposts 添加了 `AWSS3OnOutpostsServiceRolePolicy` 作为服务相关角色 `AWSServiceRoleForS3OnOutposts` 的一部分，它可帮助您管理网络资源。	2023 年 10 月 3 日
S3 on Outposts 开始跟踪更改	S3 on Outposts 开始跟踪其 AWS 托管式策略的更改。	2023 年 10 月 3 日

将服务相关角色用于 Amazon S3 on Outposts

Amazon S3 on Outposts 使用 AWS Identity and Access Management（IAM）服务相关角色。服务相关角色是一种与 S3 on Outposts 直接关联的独特类型的 IAM 角色。服务相关角色由 S3 on Outposts 预定义，具有服务代表您调用其它 AWS 服务所需的所有权限。

服务相关角色可让您更轻松地设置 S3 on Outposts，因为您不必手动添加必要的权限。S3 on Outposts 定义其服务相关角色的权限，除非另外定义，否则只有 S3 on Outposts 可以代入该角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

只有在首先删除相关资源后，您才能删除服务关联角色。这将保护您的 S3 on Outposts 资源，因为您不会无意中删除对资源的访问权限。

有关支持服务相关角色的其他服务的信息，请参阅与 IAM 配合使用的 AWS 服务，并查找 Service-linked roles（服务相关角色）列中显示为 Yes（是）的服务。选择是和链接，查看该服务的服务关联角色文档。

S3 on Outposts 的服务相关角色权限

S3 on Outposts 使用名为 AWSServiceRoleForS3OnOutposts 的服务相关角色来帮助您管理网络资源。

`AWSServiceRoleForS3OnOutposts` 服务相关角色信任以下服务代入该角色：

- `s3-outposts.amazonaws.com`

名为 `AWSS3OnOutpostsServiceRolePolicy` 的角色权限策略允许 S3 on Outposts 对指定资源完成以下操作：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcs",
                "ec2:DescribeCoipPools",
                "ec2:GetCoipPoolUsage",
                "ec2:DescribeAddresses",
                "ec2:DescribeLocalGatewayRouteTableVpcAssociations"
            ],
            "Resource": "*",
            "Sid": "DescribeVpcResources"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ],
            "Sid": "CreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForCreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:ipv4pool-ec2/*"
            ],
            "Sid": "AllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:elastic-ip/*"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForAllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:DisassociateAddress",
                "ec2:ReleaseAddress",
                "ec2:AssociateAddress"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "ReleaseVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "CreateNetworkInterface",
                        "AllocateAddress"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "S3 On Outposts"
                    ]
                }
            },
```

```
            "Sid": "CreateTags"
        }
    ]
}
```

必须配置权限，以允许 IAM 实体（如角色）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》中的服务相关角色权限。

为 S3 on Outposts 创建服务相关角色

您无需手动创建服务关联角色。当您在 AWS 管理控制台、AWS CLI 或 AWS API 中创建 S3 on Outposts 端点时，S3 on Outposts 会为您创建服务相关角色。

如果您删除该服务关联角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。当您创建 S3 on Outposts 端点时，S3 on Outposts 会再次为您创建服务相关角色。

您也可以使用 IAM 控制台为 S3 on Outposts 使用案例创建服务相关角色。在 AWS CLI 或 AWS API 中，使用 `s3-outposts.amazonaws.com` 服务名称创建服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的创建服务相关角色。如果您删除了此服务相关角色，可以使用同样的过程再次创建角色。

编辑 S3 on Outposts 的服务相关角色

S3 on Outposts 不允许您编辑 `AWSServiceRoleForS3OnOutposts` 服务相关角色。这包括角色的名称，因为可能有不同的实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务关联角色。

删除 S3 on Outposts 的服务相关角色

如果不再需要使用某个需要服务相关角色的特征或服务，我们建议您删除该角色。这样就没有未被主动监控或维护的未使用实体。但是，必须先清除服务相关角色的资源，然后才能手动删除它。

> **Note**
>
> 如果在您试图删除资源时 S3 on Outposts 服务正在使用该角色，则删除操作可能会失败。如果发生这种情况，请等待几分钟后重试。

删除 AWSServiceRoleForS3OnOutposts 角色使用的 S3 on Outposts 资源

1. 在您的 AWS 账户中跨所有 AWS 区域删除 S3 on Outposts 端点。
2. 使用 IAM 删除服务相关角色。

   使用 IAM 控制台，即 AWS CLI 或 AWS API 来删除 `AWSServiceRoleForS3OnOutposts` 服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的删除服务相关角色。

S3 on Outposts 服务相关角色的受支持区域

S3 on Outposts 支持在提供服务相关角色的所有 AWS 区域中使用该服务。有关更多信息，请参阅 S3 on Outposts 区域和端点。

管理 S3 on Outposts 存储

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅 什么是 Amazon S3 on Outposts？。

有关管理共享 Amazon S3 on Outposts 存储容量的更多信息，请参阅以下主题。

主题

- 为 S3 on Outposts 桶管理 S3 版本控制
- 为 Amazon S3 on Outposts 桶创建和管理生命周期配置
- 复制 S3 on Outposts 的对象
- 通过使用 AWS RAM 共享 S3 on Outposts
- 使用 S3 on Outposts 的其他 AWS 服务

为 S3 on Outposts 桶管理 S3 版本控制

启用后，S3 版本控制功能将对象的多个不同副本保存到同一个桶中。对于 Outpost 桶中存储的每个对象，您可以使用 S3 版本控制功能来保留、检索和还原其每个版本。S3 版本控制功能可帮助您从用户意外操作和应用程序故障中恢复。

Amazon S3 on Outposts 桶具有三种版本控制状态：

- Unversioned（不受版本控制）- 如果您从未在桶上启用或暂停过 S3 版本控制，则桶处于不受版本控制状态，并且不返回 S3 版本控制状态。有关 S3 版本控制的更多信息，请参阅为 S3 on Outposts 桶管理 S3 版本控制。
- Enabled（已启用）- 为桶中的对象启用 S3 版本控制。添加到桶的所有对象都将收到唯一的版本 ID。启用版本控制时存在于存储桶中的对象的版本 ID 为`null`。如果使用其他操作修改这些（或任何其他）对象，例如 PutObject，则新对象将获得唯一的版本 ID。

- Suspended（暂停）- 对桶中的对象暂停 S3 版本控制。暂停版本控制后添加到桶的所有对象都将收到版本 ID null。有关更多信息，请参阅《Amazon S3 用户指南》中的将对象添加到已暂停版本控制的存储桶。

在对 S3 on Outposts 桶启用 S3 版本控制后，它将无法返回到不受版本控制状态。但是，您可以暂停版本控制。有关 S3 版本控制的更多信息，请参阅为 S3 on Outposts 桶管理 S3 版本控制。

对于桶中的每个对象，您都有一个当前版本以及零个或零个以上的非当前版本。为了降低存储成本，您可以将桶 S3 生命周期规则配置为使非当前版本在指定时间段后过期。有关更多信息，请参阅 为 Amazon S3 on Outposts 桶创建和管理生命周期配置。

以下示例显示如何使用 AWS 管理控制台 和 AWS Command Line Interface（AWS CLI）对现有 S3 on Outposts 桶启用或暂停版本控制。要创建启用了 S3 版本控制的桶，请参阅创建 S3 on Outposts 存储桶。

> **Note**
>
> 创建存储桶的 AWS 账户 拥有该存储桶，也是唯一可以向其提交操作的账户。存储桶具有配置属性，如 Outpost、标签、默认加密和访问点设置。访问点设置包括用于访问存储桶中对象的 Virtual Private Cloud (VPC) 和访问点策略以及其他元数据。有关更多信息，请参阅 S3 on Outposts 规范。

使用 S3 控制台

编辑桶的 S3 版本控制设置

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要为其启用 S3 版本控制的 Outposts 桶。
4. 选择属性选项卡。
5. 在存储桶版本控制下，请选择编辑。
6. 通过选择以下选项之一，编辑桶的 S3 版本控制设置：
   - 要暂停 S3 版本控制并停止创建新的对象版本，请选择 Suspend（暂停）。
   - 要启用 S3 版本控制并保存每个对象的多个不同副本，请选择 Enable（启用）。

7. 选择保存更改。

使用 AWS CLI

要使用 AWS CLI 对桶启用或暂停 S3 版本控制，请使用 `put-bucket-versioning` 命令，如以下示例所示。要使用这些示例，请将每个 *`user input placeholder`* 替换为您自己的信息。

有关更多信息，请参阅《AWS CLI 参考》中的 put-bucket-versioning。

Example：启用 S3 版本控制

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --versioning-configuration Status=Enabled
```

Example：暂停 S3 版本控制

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket --versioning-configuration Status=Suspended
```

为 Amazon S3 on Outposts 桶创建和管理生命周期配置

您可以使用 S3 生命周期优化 Amazon S3 on Outposts 的存储容量。您可以创建生命周期规则，使对象在老化时过期或被较新版本取代。您可以创建、启用、禁用或删除生命周期规则。

有关 S3 生命周期的更多信息，请参阅为 Amazon S3 on Outposts 桶创建和管理生命周期配置。

> **Note**
>
> 创建桶的 AWS 账户拥有该桶，并且是唯一可以创建、启用、禁用或删除生命周期规则的账户。

要创建和管理 S3 on Outposts 桶的生命周期配置，请参阅以下主题。

主题

- 使用 AWS 管理控制台创建和管理生命周期规则
- 使用 AWS CLI 和适用于 Java 的 SDK 创建和管理生命周期配置

使用 AWS 管理控制台创建和管理生命周期规则

您可以使用 S3 生命周期优化 Amazon S3 on Outposts 的存储容量。您可以创建生命周期规则，使对象在老化时过期或被较新版本取代。您可以创建、启用、禁用或删除生命周期规则。

有关 S3 生命周期的更多信息，请参阅为 Amazon S3 on Outposts 桶创建和管理生命周期配置。

> **Note**
>
> 创建桶的 AWS 账户拥有该桶，并且是唯一可以创建、启用、禁用或删除生命周期规则的账户。

要使用 AWS 管理控制台 为 S3 on Outposts 创建和管理生命周期规则，请参阅以下主题。

主题

创建生命周期规则

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要为其创建生命周期规则的 Outposts 桶。
4. 选择 Management（管理）选项卡，然后选择 Create Lifecycle rule（创建生命周期规则）。
5. 为 Lifecycle rule name（生命周期规则名称）输入一个值。
6. 在 Rule scope（角色范围）下，选择以下选项之一：
   - 为将范围限制到特定筛选条件，请选择 Limit the scope of this rule using one or more filters（使用一个或多个筛选条件限制此规则的范围）。然后，添加前缀筛选条件、标签或对象大小。
   - 要将此规则应用于桶中的所有对象，请选择 Apply to all objects in the bucket（应用到桶中的所有对象）。
7. 在 Lifecycle rule actions（生命周期规则操作）下，选择以下选项之一：

- Expire current versions of objects（将对象的当前版本设为过期）- 对于启用版本控制的桶，S3 on Outposts 会添加删除标记，并将对象保留为非当前版本。对于不使用 S3 版本控制的桶，S3 on Outposts 会永久删除对象。
- Permanently delete noncurrent versions of objects（永久删除对象的非当前版本）– S3 on Outposts 永久删除对象的非当前版本。
- Delete expired object delete markers or incomplete multipart uploads（删除过期的删除标记或未完成的分段上传）– S3 on Outposts 永久删除过期的删除标记或未完成的分段上传。

  如果您使用对象标签限制生命周期规则的范围，则无法选择 Delete expired object delete markers（删除过期的对象删除标记）。如果您选择 Expire current object versions（使当前对象版本过期），也无法选择 Delete expired object delete markers（删除过期的对象删除标记）。

> **Note**
>
> 基于大小的筛选条件不能用于删除标记和未完成的分段上传。

8. 如果您选择了 Expire current versions of objects（使当前版本的对象过期）或 Permanently delete noncurrent versions of objects（永久删除对象的非当前版本），请根据特定日期或对象的存在期限配置规则触发器。
9. 如果您选择了 Delete expired object delete markers（删除过期的对象删除标记），要确认删除过期的对象删除标记，请选择 Delete expired object delete markers（删除过期的对象删除标记）。
10. 在 Timeline Summary（时间线摘要）下，查看您的生命周期规则，然后选择 Create rule（创建规则）。

启用生命周期规则

启用或禁用桶生命周期规则

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要为其启用或禁用生命周期规则的 Outposts 桶。
4. 选择 Management（管理）选项卡，然后在 Lifecycle rule（生命周期规则）下选择要启用或禁用的规则。
5. 对于 Action（操作），选择 Enable or disable rule（启用或禁用规则）。

编辑生命周期规则

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要为其编辑生命周期规则的 Outposts 桶。
4. 选择 Management（管理）选项卡，然后选择要编辑的生命周期规则。
5. （可选）更新 Lifecycle rule name（生命周期规则名称）的值。
6. 在 Rule scope（规则范围）下，根据需要编辑范围：

   - 为将范围限制到特定筛选条件，请选择 Limit the scope of this rule using one or more filters（使用一个或多个筛选条件限制此规则的范围）。然后，添加前缀筛选条件、标签或对象大小。
   - 要将此规则应用于桶中的所有对象，请选择 Apply to all objects in the bucket（应用到桶中的所有对象）。
7. 在 Lifecycle rule actions（生命周期规则操作）下，选择以下选项之一：

   - Expire current versions of objects（将对象的当前版本设为过期）- 对于启用版本控制的桶，S3 on Outposts 会添加删除标记，并将对象保留为非当前版本。对于不使用 S3 版本控制的桶，S3 on Outposts 会永久删除对象。
   - Permanently delete noncurrent versions of objects（永久删除对象的非当前版本）– S3 on Outposts 永久删除对象的非当前版本。
   - Delete expired object delete markers or incomplete multipart uploads（删除过期的删除标记或未完成的分段上传）– S3 on Outposts 永久删除过期的删除标记或未完成的分段上传。

     如果您使用对象标签限制生命周期规则的范围，则无法选择 Delete expired object delete markers（删除过期的对象删除标记）。如果您选择 Expire current object versions（使当前对象版本过期），也无法选择 Delete expired object delete markers（删除过期的对象删除标记）。

   > **Note**
   >
   > 基于大小的筛选条件不能用于删除标记和未完成的分段上传。

8. 如果您选择了 Expire current versions of objects（使当前版本的对象过期）或 Permanently delete noncurrent versions of objects（永久删除对象的非当前版本），请根据特定日期或对象存在期限配置规则触发器。

9. 如果您选择了 Delete expired object delete markers（删除过期的对象删除标记），要确认删除过期的对象删除标记，请选择 Delete expired object delete markers（删除过期的对象删除标记）。
10. 选择保存。

删除生命周期规则

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要为其删除生命周期规则的 Outposts 桶。
4. 选择 Management（管理）选项卡，然后 Lifecycle rule（生命周期规则）下，选择要删除的规则。
5. 选择 Delete。

使用 AWS CLI 和适用于 Java 的 SDK 创建和管理生命周期配置

您可以使用 S3 生命周期优化 Amazon S3 on Outposts 的存储容量。您可以创建生命周期规则，使对象在老化时过期或被较新版本取代。您可以创建、启用、禁用或删除生命周期规则。

有关 S3 生命周期的更多信息，请参阅为 Amazon S3 on Outposts 桶创建和管理生命周期配置。

> **Note**
>
> 创建桶的 AWS 账户拥有该桶，并且是唯一可以创建、启用、禁用或删除生命周期规则的账户。

要使用 AWS Command Line Interface（AWS CLI）和 适用于 Java 的 AWS SDK 创建和管理 S3 on Outposts 桶的生命周期配置，请参阅以下示例。

主题

- 放置 (PUT) 生命周期配置
- 获取（GET）S3 on Outposts 桶上的生命周期配置

放置 (PUT) 生命周期配置

AWS CLI

以下 AWS CLI 示例在 Outposts 桶上放置生命周期配置策略。此策略指定具有标记前缀 (*myprefix*) 的所有对象，并且标签在 10 天后过期。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

1. 将生命周期配置策略保存到 JSON 文件中。在本示例中，文件命名为 `lifecycle1.json`。

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ],
                    "ObjectSizeGreaterThan": 1000,
                    "ObjectSizeLessThan": 5000
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 10
            }
        }
    ]
}
```

2. 将 JSON 文件作为`put-bucket-lifecycle-configuration` CLI 命令的一部分提交。要使用此命令，请将每个 *user input placeholder* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 put-bucket-lifecycle-configuration。

```
aws s3control put-bucket-lifecycle-configuration --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket --lifecycle-configuration file://lifecycle1.json
```

SDK for Java

以下适用于 Java 的 SDK 示例在 Outposts 桶上放置生命周期配置。此生命周期配置指定具有标记前缀 (*myprefix*) 的所有对象，并且标签在 10 天后过期。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 PutBucketLifecycleConfiguration。

```
import com.amazonaws.services.s3control.model.*;

public void putBucketLifecycleConfiguration(String bucketArn) {

    S3Tag tag1 = new S3Tag().withKey("mytagkey1").withValue("mytagkey1");
    S3Tag tag2 = new S3Tag().withKey("mytagkey2").withValue("mytagkey2");

    LifecycleRuleFilter lifecycleRuleFilter = new LifecycleRuleFilter()
            .withAnd(new LifecycleRuleAndOperator()
                    .withPrefix("myprefix")
                    .withTags(tag1, tag2))
                    .withObjectSizeGreaterThan(1000)
                    .withObjectSizeLessThan(5000);

    LifecycleExpiration lifecycleExpiration = new LifecycleExpiration()
            .withExpiredObjectDeleteMarker(false)
            .withDays(10);

    LifecycleRule lifecycleRule = new LifecycleRule()
            .withStatus("Enabled")
            .withFilter(lifecycleRuleFilter)
            .withExpiration(lifecycleExpiration)
            .withID("id-1");

    LifecycleConfiguration lifecycleConfiguration = new LifecycleConfiguration()
            .withRules(lifecycleRule);

    PutBucketLifecycleConfigurationRequest reqPutBucketLifecycle = new
 PutBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
```

```
            .withBucket(bucketArn)
            .withLifecycleConfiguration(lifecycleConfiguration);

    PutBucketLifecycleConfigurationResult respPutBucketLifecycle =
 s3ControlClient.putBucketLifecycleConfiguration(reqPutBucketLifecycle);
    System.out.printf("PutBucketLifecycleConfiguration Response: %s%n",
 respPutBucketLifecycle.toString());

}
```

获取（GET）S3 on Outposts 桶上的生命周期配置

AWS CLI

以下 AWS CLI 示例获取 Outposts 桶上的生命周期配置。要使用此命令，请将每个 *user input placeholder* 替换为您自己的信息。有关此命令的更多信息，请参阅《AWS CLI 参考》中的 get-bucket-lifecycle-configuration。

```
aws s3control get-bucket-lifecycle-configuration --account-id 123456789012 --bucket
 arn:aws:s3-outposts:<your-region>:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket
```

SDK for Java

以下适用于 Java 的 SDK 示例获取 Outposts 桶的生命周期配置。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 GetBucketLifecycleConfiguration。

```
import com.amazonaws.services.s3control.model.*;

public void getBucketLifecycleConfiguration(String bucketArn) {

    GetBucketLifecycleConfigurationRequest reqGetBucketLifecycle = new
 GetBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketLifecycleConfigurationResult respGetBucketLifecycle =
 s3ControlClient.getBucketLifecycleConfiguration(reqGetBucketLifecycle);
    System.out.printf("GetBucketLifecycleConfiguration Response: %s%n",
 respGetBucketLifecycle.toString());
```

```
}
```

复制 S3 on Outposts 的对象

借助 AWS Outposts 上的 S3 复制，您可以将 Amazon S3 on Outposts 配置为在不同的 Outposts 之间或在同一 Outpost 上的桶之间自动复制 S3 对象。您可以使用 Outposts 上的 S3 复制，在相同或不同的 Outposts 或跨不同的账户维护数据的多个副本，以帮助满足数据驻留需求。Outposts 上的 S3 复制有助于满足您的合规存储需求并实现跨账户的数据共享。如果需要确保副本与源数据完全相同，可以使用 Outposts 上的 S3 复制来制作保留所有元数据的对象的副本，如原始对象创建时间、标签和版本 ID。

Outposts 上的 S3 复制还提供详细的指标和通知，以监控桶之间对象复制的状态。您可以使用 Amazon CloudWatch，通过跟踪待复制的字节数、待复制的操作数以及源桶和目标桶之间的复制延迟来监控复制进度。要快速诊断和纠正配置问题，还可以设置 Amazon EventBridge 以接收有关复制对象失败的通知。要了解更多信息，请参阅管理您的复制。

主题

- 复制配置
- Outposts 上的 S3 复制的要求
- 复制什么内容？
- 不复制什么内容？
- Outposts 上的 S3 复制不支持什么？
- 设置复制
- 管理您的复制

复制配置

S3 on Outposts 以 XML 格式存储复制配置。在复制配置 XML 文件中，您将指定一个 AWS Identity and Access Management (IAM) 角色以及一个或多个规则。

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
```

```
        ...
    </Rule>
     ...
</ReplicationConfiguration>
```

S3 on Outposts 无法在未经您许可的情况下复制对象。您使用在复制配置中指定的 IAM 角色向 S3 on Outposts 授予权限。S3 on Outposts 代入此 IAM 角色以代表您复制对象。在开始复制之前，您必须向 IAM 角色授予所需的权限。有关 S3 on Outposts 的这些权限的更多信息，请参阅创建 IAM 角色。

在以下场景的复制配置中添加一个规则：

- 您希望复制所有对象。
- 您希望复制对象子集。通过在规则中添加一个筛选条件，可标识对象子级。在该筛选条件中，指定对象键前缀、标签或二者的组合以标识要向其应用规则的对象子集。

如果要复制其他对象子集，请在复制配置中添加多个规则。在每个规则中，指定一个选择不同对象子集的筛选条件。例如，您可能选择了键前缀为 `tax/` 或 `document/` 的复制对象。要做到这一点，您需要添加两个规则，一个规则指定 `tax/` 键前缀筛选条件，另一个指定 `document/` 键前缀。

有关 S3 on Outposts 复制配置和复制规则的更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 ReplicationConfiguration。

Outposts 上的 S3 复制的要求

复制有下列要求：

- 目标 Outpost CIDR 范围必须在源 Outpost 子网表中关联。有关更多信息，请参阅 创建复制规则的先决条件。
- 源桶和目标桶必须均已启用 S3 版本控制。有关版本控制的更多信息，请参阅为 S3 on Outposts 桶管理 S3 版本控制。
- Amazon S3 on Outposts 必须有权代表您将对象从源桶复制到目标桶。这意味着您必须创建一个服务角色，以便将 `GET` 和 `PUT` 权限委派给 S3 on Outposts。
  1. 在创建服务角色之前，您必须拥有源桶的 `GET` 权限和目标桶的 `PUT` 权限。
  2. 要创建服务角色以将权限委派给 S3 on Outposts，您必须先配置权限以允许 IAM 实体（用户或角色）执行 `iam:CreateRole` 和 `iam:PassRole` 操作。然后，您允许 IAM 实体创建服务角色。要使 S3 on Outposts 代表您代入此服务角色，并将 `GET` 和 `PUT` 权限委派给 S3 on Outposts，您必须为该角色分配必要的信任和权限策略。有关 S3 on Outposts 的这些权限的更多信息，请参阅创建 IAM 角色。有关创建服务角色的更多信息，请参阅创建服务角色。

复制什么内容？

原定设置情况下，S3 on Outposts 会复制以下内容：

- 添加复制配置之后创建的对象。
- 从源对象到副本的对象元数据。有关如何将元数据从副本复制到源对象的信息，请参阅在 Outposts 上启用了 Amazon S3 副本修改同步时的复制状态。
- 对象标签（如果有）。

删除操作对复制操作有何影响

如果您从源存储桶中删除对象，则默认情况下会执行以下操作：

- 如果您发出 DELETE 请求而未指定对象版本 ID，S3 on Outposts 会添加删除标记。S3 on Outposts 将按如下所示处理该删除标记：
  - 原定设置情况下，S3 on Outposts 不复制删除标记。
  - 但是，您可以将删除标记复制添加到非基于标记的规则。有关如何在复制配置中启用删除标记复制的更多信息，请参阅使用 S3 控制台。
- 如果您在 DELETE 请求中指定一个要删除的对象版本 ID，S3 on Outposts 会在源桶中永久删除该对象版本。但是，它不会将删除操作复制到目标桶中。换句话说，它不会从目标存储桶中删除同一对象版本。此行为可防止恶意删除数据。

不复制什么内容？

原定设置情况下，S3 on Outposts 不复制以下内容：

- 源存储桶中作为另一个复制规则所建副本的对象。例如，假设您配置的复制中，存储桶 A 是源，存储桶 B 是目标。现在假设您添加另一个复制配置，其中存储桶 B 是源，而存储桶 C 是目标。在这种情况下，存储桶 B 中作为存储桶 A 中对象的副本的对象不会复制到存储桶 C。
- 源存储桶中已复制到其他目标的对象。例如，如果您在现有复制配置中更改目标桶，则 S3 on Outposts 不会再次复制对象。
- 使用具有客户提供的加密密钥的服务器端加密（SSE-C）创建的对象。
- 对存储桶级别子资源进行的更新。

  例如，如果您更改生命周期配置或向源存储桶添加通知配置，则这些更改不会应用于目标存储桶。此功能使您可以在源桶和目标桶中具有不同的配置。

- 由生命周期配置执行的操作。

  例如，如果您只在源桶上启用生命周期配置并配置过期操作，S3 on Outposts 会为源桶中的过期对象创建删除标记，但不会将这些标记复制到目标桶。如果您希望对源存储桶和目标存储桶应用相同的生命周期配置，请对这两个存储桶启用相同的生命周期配置。有关生命周期配置的更多信息，请参阅为 Amazon S3 on Outposts 桶创建和管理生命周期配置。

Outposts 上的 S3 复制不支持什么？

S3 on Outposts 目前不支持以下 S3 复制功能。

- S3 Replication Time Control（S3 RTC）。不支持 S3 RTC，因为 Outposts 上的 S3 复制中的对象流量通过您的本地网络（本地网关）传输。有关本地网关的更多信息，请参阅《AWS Outposts 用户指南》中的使用本地网关。
- 适用于批量操作的 S3 复制。

设置复制

> **Note**
>
> 将不会自动复制桶中在您设置复制规则之前就存在的对象。换句话说，Amazon S3 on Outposts 不以回溯方式复制对象。要复制在复制配置之前创建的对象，您可以使用 `CopyObject` API 操作将它们复制到同一个桶中。复制对象后，它们在桶中显示为“新”对象，复制配置将应用于它们。有关复制对象的详细信息，请参阅使用 适用于 Java 的 AWS SDK 复制 Amazon S3 on Outposts 存储桶中的对象和《Amazon Simple Storage Service API 参考》中的 CopyObject。

要启用 Outposts 上的 S3 复制，请将复制规则添加到您的源 Outposts 桶。复制规则指示 S3 on Outposts 按照指定的方式复制对象。在复制规则中，您必须提供以下内容：

- 源 Outposts 桶接入点 – 您希望 S3 on Outposts 从中复制对象的桶的接入点 Amazon 资源名称（ARN）或接入点别名。有关使用接入点别名的更多信息，请参阅为 S3 on Outposts 桶接入点使用桶式别名。
- 要复制的对象 – 您可以复制源 Outposts 桶中的所有对象或对象子集。通过在配置中提供一个键名前缀和/或一个或多个对象标签，可标识子集。

例如，如果您配置复制规则以仅复制键名称前缀为 `Tax/` 的对象，则 S3 on Outposts 仅复制键为 `Tax/doc1` 或 `Tax/doc2` 之类的对象。但它不复制具有键 `Legal/doc3` 的对象。如果您同时指定前缀和一个或多个标签，则 S3 on Outposts 仅复制具有特定键前缀和标签的对象。

- 目标 Outposts 桶 – 您希望 S3 on Outposts 将对象复制到的桶的 ARN 或接入点别名。

您可以使用 REST API、AWS SDK、AWS Command Line Interface（AWS CLI）或 Amazon S3 控制台配置复制规则。

S3 on Outposts 还提供了 API 操作来支持设置复制规则。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的以下主题：

- PutBucketReplication
- GetBucketReplication
- DeleteBucketReplication

主题

创建复制规则的先决条件

主题

连接您的源和目标 Outpost 子网

要使复制流量通过本地网关从源 Outpost 传输到目标 Outpost，您必须添加一条新路由来设置网络。必须将接入点的无类别域间路由（CIDR）网络范围连接在一起。对于每对接入点，您只需设置一次此连接。

设置连接的某些步骤有所不同，具体取决于与接入点关联的 Outposts 端点的访问类型。端点的访问类型为私有（适用于 AWS Outposts 的直接虚拟私有云 [VPC] 路由）或客户拥有的 IP（本地网络中客户拥有的 IP 地址池 [CoIP 池]）。

步骤 1：查找源 Outposts 端点的 CIDR 范围

查找与源接入点关联的源端点的 CIDR 范围

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 在 Outposts 桶列表中，选择要用于复制的源桶。
4. 选择 Outposts 接入点选项卡，然后为您的复制规则选择源桶的 Outposts 接入点。
5. 选择 Outposts 端点。
6. 复制子网 ID 以在步骤 5 中使用。
7. 用于查找源 Outposts 端点的 CIDR 范围的方法取决于您的端点的访问类型。

   在 Outposts 端点概览部分中，查看访问类型。

   - 如果访问类型为私有，请复制无类别域间路由（CIDR）值以在步骤 6 中使用。
   - 如果访问类型为客户拥有的 IP，请执行以下操作：
     1. 复制客户拥有的 IPv4 池值，稍后将用作地址池的 ID。
     2. 打开 AWS Outposts 控制台 (https://console.aws.amazon.com/outposts/)。
     3. 在导航窗格中，选择本地网关路由表。
     4. 选择您的源 Outpost 的本地网关路由表 ID 值。
     5. 在详细信息窗格中，选择 CoIP 池选项卡。将您之前复制的 CoIP 池 ID 的值粘贴到搜索框中。
     6. 对于匹配的 CoIP 池，复制您的源 Outposts 端点的相应 CIDR 值，以便在步骤 6 中使用。

步骤 2：查找目标 Outposts 端点的子网 ID 和 CIDR 范围

要查找与目标接入点关联的目标端点的子网 ID 和 CIDR 范围，请按照步骤 1 中的相同子步骤进行操作，并在应用这些子步骤时将源 Outposts 端点更改为目标 Outposts 端点。复制目标 Outposts 端点的子网 ID 值，以便在步骤 6 中使用。复制目标 Outposts 端点的 CIDR 值，以便在步骤 5 中使用。

步骤 3：查找源 Outpost 的本地网关 ID

查找源 Outpost 的本地网关 ID

1. 打开 AWS Outposts 控制台 (https://console.aws.amazon.com/outposts/)。

2. 在左侧导航窗格中，选择本地网关。
3. 在本地网关页面上，找到要用于复制的源 Outpost 的 Outpost ID。
4. 复制源 Outpost 的本地网关 ID 值，以便在步骤 5 中使用。

有关本地网关的更多信息，请参阅《AWS Outposts 用户指南》中的本地网关。

步骤 4：查找目标 Outpost 的本地网关 ID

要查找目标 Outpost 的本地网关 ID，请按照步骤 3 中的相同子步骤操作，不同之处是查找目标 Outpost 的 Outpost ID。复制目标 Outpost 的本地网关 ID 值，以便在步骤 6 中使用。

步骤 5：设置从源 Outpost 子网到目标 Outpost 子网的连接

从源 Outpost 子网连接到目标 Outpost 子网

1. 登录到 AWS 管理控制台并打开 VPC 控制台，网址为：https://console.aws.amazon.com/vpc/。
2. 在左侧导航窗格中，选择 Subnets。
3. 在搜索框中，输入您在步骤 1 中找到的源 Outposts 端点的子网 ID。选择具有匹配的子网 ID 的子网。
4. 对于匹配的子网项目，选择该子网的路由表值。
5. 在包含选定路由表的页面上，选择操作，然后选择编辑路由。
6. 在编辑路由页面上，选择添加路由。
7. 在目标下，输入您在步骤 2 中找到的目标 Outposts 端点的 CIDR 范围。
8. 在目标下，选择 Outpost 本地网关，然后输入您在步骤 3 中找到的源 Outpost 的本地网关 ID。
9. 选择保存更改。
10. 确保路由的状态为活动。

步骤 6：设置从目标 Outpost 子网到源 Outpost 子网的连接

1. 登录到 AWS 管理控制台并打开 VPC 控制台，网址为：https://console.aws.amazon.com/vpc/。
2. 在左侧导航窗格中，选择 Subnets。
3. 在搜索框中，输入您在步骤 2 中找到的目标 Outposts 端点的子网 ID。选择具有匹配的子网 ID 的子网。
4. 对于匹配的子网项目，选择该子网的路由表值。
5. 在包含选定路由表的页面上，选择操作，然后选择编辑路由。

6. 在编辑路由页面上，选择添加路由。
7. 在目标下，输入您在步骤 1 中找到的源 Outposts 端点的 CIDR 范围。
8. 在目标下，选择 Outpost 本地网关，然后输入您在步骤 4 中找到的目标 Outpost 的本地网关 ID。
9. 选择保存更改。
10. 确保路由的状态为活动。

连接源接入点和目标接入点的 CIDR 网络范围后，必须创建 AWS Identity and Access Management（IAM）角色。

创建 IAM 角色

原定设置情况下，所有 S3 on Outposts 资源（桶、对象和相关子资源）都是私有的，只有资源拥有者可以访问相应的资源。S3 on Outposts 需要权限才能从源 Outposts 桶读取和复制对象。您通过创建一个 IAM 服务角色，然后在复制配置中指定该角色，授予这些权限。

此部分介绍信任策略及所需的最低权限策略。示例演练提供创建 IAM 角色的分步说明。有关更多信息，请参阅 在 Outposts 上创建复制规则。有关 IAM 角色的更多信息，请参阅《IAM 用户指南》中的 IAM 角色。

- 以下示例显示了一个信任策略，您在其中将 S3 on Outposts 标识为可代入此角色的服务主体。

  JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

- 以下示例显示了一个访问策略，您在其中向角色授予代表您执行复制任务的权限。当 S3 on Outposts 代入此角色时，它将具有您在此策略中指定的权限。要使用这一策略，请将 *user input placeholders* 替换为您自己的信息。确保将它们替换为源和目标 Outpost 的 Outpost ID 以及源和目标 Outposts 桶的桶名称和接入点名称。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3-outposts:GetObjectVersionForReplication",
                "s3-outposts:GetObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-east-1:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
                "arn:aws:s3-outposts:us-east-1:123456789012:outpost/SOURCE-OUTPOST-ID/accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3-outposts:ReplicateObject",
                "s3-outposts:ReplicateDelete"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-east-1:123456789012:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
                "arn:aws:s3-outposts:us-east-1:123456789012:outpost/DESTINATION-OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
        }
    ]
}
```

访问策略授予以下操作的权限：

- `s3-outposts:GetObjectVersionForReplication` – 授予对所有对象执行此操作的权限，以允许 S3 on Outposts 获取与每个对象关联的特定对象版本。

- `s3-outposts:GetObjectVersionTagging` – 针对 *SOURCE-OUTPOSTS-BUCKET* 桶（源桶）中的对象执行此操作的权限允许 S3 on Outposts 读取对象标签以进行复制。有关更多信息，请参阅 为 S3 on Outposts 存储桶添加标签。如果 S3 on Outposts 没有此权限，它将复制对象而不是对象标签。
- `s3-outposts:ReplicateObject` 和 `s3-outposts:ReplicateDelete` – 针对 *DESTINATION-OUTPOSTS-BUCKET* 桶（目标桶）中的所有对象执行这些操作的权限允许 S3 on Outposts 将对象或删除标记复制到目标 Outposts 桶。有关删除标记的信息，请参阅 删除操作对复制操作有何影响。

  > **Note**
  >
  > - 对 *DESTINATION-OUTPOSTS-BUCKET* 桶（目标桶）执行 `s3-outposts:ReplicateObject` 操作的权限还允许复制对象标签。因此，您无需显式授予执行 `s3-outposts:ReplicateTags` 操作的权限。
  > - 对于跨账户复制，目标 Outposts 桶的拥有者必须更新其桶策略，以授予对 *DESTINATION-OUTPOSTS-BUCKET* 执行 `s3-outposts:ReplicateObject` 操作的权限。`s3-outposts:ReplicateObject` 操作允许 S3 on Outposts 将对象和对象标签复制到目标 Otposts 桶。

有关 S3 on Outposts 操作的列表，请参阅 S3 on Outposts 定义的操作。

> **Important**
>
> 拥有 IAM 角色的 AWS 账户 必须拥有其向 IAM 角色授予的操作的权限。
> 例如，假定源 Outposts 桶包含由另一个 AWS 账户拥有的对象。对象的拥有者必须通过桶策略和接入点策略，向拥有 IAM 角色的 AWS 账户显式授予必要的权限。否则，S3 on Outposts 无法访问这些对象，因而这些对象的复制将失败。
> 此处介绍的权限与最低复制配置相关。如果您选择添加可选复制配置，则必须向 S3 on Outposts 授予其他权限。

在源和目标 Outposts 桶由不同的 AWS 账户拥有时授予权限

当源和目标 Outposts 桶由不同的账户拥有时，目标 Outposts 桶的拥有者必须针对目标桶更新桶策略和接入点策略。这些策略必须向源 Outposts 桶的拥有者和 IAM 服务角色授予执行复制操作的权限，如以下策略示例所示，否则复制将失败。在以下策略示例中，*DESTINATION-OUTPOSTS-BUCKET* 是目标桶。要使用这些策略示例，请将 *user input placeholders* 替换为您自己的信息。

如果您手动创建 IAM 服务角色，请将角色路径设置为 `role/service-role/`，如以下策略示例所示。有关更多信息，请参阅 IAM 用户指南中的 IAM ARN。

JSON

```
{
    "Version":"2012-10-17",
    "Id": "PolicyForDestinationBucket",
    "Statement": [
        {
            "Sid": "Permissions on objects",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/service-role/source-account-IAM-role"
            },
            "Action": [
                "s3-outposts:ReplicateDelete",
                "s3-outposts:ReplicateObject"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-east-1:444455556666:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*"
            ]
        }
    ]
}
```

JSON

```
{
    "Version":"2012-10-17",
    "Id": "PolicyForDestinationAccessPoint",
    "Statement": [
        {
            "Sid": "Permissions on objects",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/service-role/source-account-IAM-role"
```

```
            },
            "Action": [
                "s3-outposts:ReplicateDelete",
                "s3-outposts:ReplicateObject"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-east-1:111122223333:outpost/DESTINATION-
OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
        }
    ]
}
```

> **Note**
>
> 如果源 Outposts 桶中的对象带有标签，请注意以下事项：
> 如果源 Outposts 桶拥有者向 S3 on Outposts 授予执行 `s3-outposts:GetObjectVersionTagging` 和 `s3-outposts:ReplicateTags` 操作的权限来复制对象标签（通过 IAM 角色），则 Amazon S3 将复制标签以及对象。有关 IAM 角色的信息，请参阅 创建 IAM 角色。

在 Outposts 上创建复制规则

Outposts 上的 S3 复制是跨相同或不同 AWS Outposts 中的桶自动、异步复制对象。复制操作会将源 Outposts 桶中新创建的对象和对象更新复制到目标 Outposts 桶。有关更多信息，请参阅 复制 S3 on Outposts 的对象。

> **Note**
>
> 将不会复制在设置复制规则之前存在于源 Outposts 桶中的对象。换句话说，S3 on Outposts 不以回溯方式复制对象。要复制在复制配置之前创建的对象，您可以使用 `CopyObject` API 操作将它们复制到同一个桶中。复制对象后，它们在桶中显示为“新”对象，复制配置将应用于它们。有关复制对象的详细信息，请参阅使用 适用于 Java 的 AWS SDK 复制 Amazon S3 on Outposts 存储桶中的对象和《Amazon Simple Storage Service API 参考》中的 CopyObject。

配置复制时，需要向源 Outposts 桶添加复制规则。复制规则定义要复制的源 Outposts 桶对象和将存储复制的对象的目标 Outposts 桶。您可以创建一条规则，以复制存储桶中的所有对象或具有特定键名

前缀和/或一个或多个对象标签的对象子集。目标 Outposts 桶与源 Outposts 桶可以位于同一 Outpost 中，也可以位于不同的 Outpost 中。

对于 S3 on Outposts 复制规则，您必须提供源 Outposts 桶的接入点 Amazon 资源名称（ARN）和目标 Outposts 桶的接入点 ARN，而不是源和目标 Outposts 桶名称。

如果您指定要删除的对象版本 ID，S3 on Outposts 会在源 Outposts 桶中删除该对象版本。但它不会将删除操作复制到目标 Outposts 桶中。换句话说，它不会从目标 Outposts 桶中删除同一对象版本。此行为可防止恶意删除数据。

当您将复制规则添加到 Outposts 桶后，原定设置情况下将启用复制规则，使该规则在您保存它后立即开始发挥作用。

在此示例中，您为位于不同的 Outposts 上且由同一 AWS 账户拥有的源和目标 Outposts 桶设置复制。提供了使用 Amazon S3 控制台、AWS Command Line Interface (AWS CLI) 以及 适用于 Java 的 AWS SDK 和 适用于 .NET 的 AWS SDK 的示例。有关跨账户 Outposts 上的 S3 复制权限的信息，请参阅在源和目标 Outposts 桶由不同的 AWS 账户拥有时授予权限。

有关设置 S3 on Outposts 复制规则的先决条件，请参阅创建复制规则的先决条件。

使用 S3 控制台

当目标 Amazon S3 on Outposts 桶位于与源 Outposts 桶不同的 Outpost 中时，请按照以下步骤配置复制规则。

如果目标 Outposts 桶与源 Outposts 桶位于不同的账户中，您必须向目标 Outposts 桶添加桶策略，以便向源 Outposts 桶账户的拥有者授予向目标 Outposts 桶复制对象的权限。

创建复制规则

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在 Outposts 桶列表中，选择要用作源桶的桶名称。
3. 选择管理选项卡，向下滚动到复制规则部分，然后选择创建复制规则。
4. 在复制规则名称下，输入规则的名称以帮助稍后识别规则。该名称是必填项，并且它在存储桶内必须是唯一的。
5. 在状态下，已启用在原定设置情况下处于选中状态。已启用的规则将在您保存它后立即开始工作。如果您想以后再启用该规则，请选择已禁用。

6. 在优先级下，规则的优先级值决定了在存在重叠的规则时要应用哪条规则。当对象包含在多个复制规则的范围内时，S3 on Outposts 使用这些优先级值来避免冲突。原定设置情况下，新规则以最高优先级添加到复制配置中。数字越大，优先级越高。

   要更改规则的优先级，请在保存规则后，从复制规则列表中选择规则名称，选择操作，然后选择编辑优先级。

7. 在源桶下，您可以通过以下选项设置复制源：

   - 要复制整个桶，请选择应用到桶中的所有 对象。
   - 要将前缀或标签筛选应用到复制源，请选择使用一个或多个筛选条件限制此规则的范围。您可以组合前缀和标签。
     - 要复制所有具有相同前缀的对象，请在前缀下方的框中输入前缀。使用前缀筛选条件将复制限制为名称以相同字符串（例如，`pictures`）开头的所有对象。

       如果您输入的前缀是文件夹名称，则必须使用 /（正斜杠）作为最后一个字符（例如，`pictures/`）。
     - 要复制具有一个或多个相同对象标签的所有对象，请选择添加标签，然后在框中输入键值对。要添加另一个标签，请重复该过程。有关对象标签的更多信息，请参阅 为 S3 on Outposts 存储桶添加标签。

8. 要访问您的 S3 on Outposts 源桶以进行复制，请在源接入点名称下，选择一个连接到源桶的接入点。

9. 在目标下，选择您希望 S3 on Outposts 向其中复制对象的目标 Outposts 桶的接入点 ARN。目标 Outposts 桶可以与源 Outposts 桶位于相同或不同的 AWS 账户中。

   如果目标桶与源 Outposts 桶位于不同的账户中，您必须向目标 Outposts 桶添加桶策略，以便为源 Outposts 桶账户的拥有者授予将对象复制到目标 Outposts 桶中的权限。有关更多信息，请参阅 在源和目标 Outposts 桶由不同的 AWS 账户拥有时授予权限。

   > **Note**
   >
   > 如果未对目标 Outposts 桶启用版本控制，您将收到包含启用版本控制按钮的警告。选择此按钮可对桶启用版本控制。

10. 设置 AWS Identity and Access Management（IAM）服务角色，S3 on Outposts 可以代入该角色以代表您复制对象。

    要设置 IAM 角色，请在 IAM 角色下执行以下操作之一：

- 要让 S3 on Outposts 为复制配置创建新的 IAM 角色，请选择从现有 IAM 角色中选择，然后选择创建新角色。当您保存该规则后，将为 IAM 角色生成一个与您选择的源和目标 Outposts 桶匹配的新策略。建议您选择创建新角色。
- 还可以选择使用现有的 IAM 角色。在这种情况下，您必须选择一个可以授予 S3 on Outposts 必要权限以进行复制的角色。如果此角色没有授予 S3 on Outposts 足够的权限来遵循您的复制规则，则复制将失败。

  要选择现有角色，请选择从现有 IAM 角色中选择，然后从下拉菜单中选择角色。您也可以选择输入 IAM 角色 ARN，然后输入 IAM 角色的 Amazon 资源名称（ARN）。

> **Important**
>
> 当您将复制规则添加到 S3 on Outposts 桶时，您必须具有 `iam:CreateRole` 和 `iam:PassRole` 权限才能创建和传递向 S3 on Outposts 授予复制权限的 IAM 角色。有关更多信息，请参阅《IAM 用户指南》中的向用户授予将角色传递给 AWS 服务的权限。

11. 原定设置情况下，Outposts 桶中的所有对象均加密。有关 S3 on Outposts 加密的更多信息，请参阅S3 on Outposts 中的数据加密。只能复制使用具有 Amazon S3 托管式密钥的服务器端加密（SSE-S3）进行加密的对象。不支持复制使用具有 AWS Key Management Service（AWS KMS）密钥的服务器端加密（SSE-KMS）或使用客户提供的加密密钥的服务器端加密（SSE-C）进行加密的对象。
12. 根据需要，在设置复制规则配置时启用以下附加选项：

    - 如果要在复制配置中启用 S3 on Outposts 复制指标，请选择复制指标。有关更多信息，请参阅使用复制指标监控进度。
    - 如果要在复制配置中启用删除标记复制，请选择 Delete marker replication (删除标记复制)。有关更多信息，请参阅 删除操作对复制操作有何影响。
    - 如果要将对副本所做的元数据更改复制回源对象，请选择副本修改同步。有关更多信息，请参阅在 Outposts 上启用了 Amazon S3 副本修改同步时的复制状态。
13. 要完成操作，请选择创建规则。

当您保存规则之后，您可以编辑、启用、禁用或删除您的规则。要执行这些操作，请转到源 Outposts 桶的管理选项卡，向下滚动到复制规则部分，选择您的规则，然后选择编辑规则。

使用 AWS CLI

要使用 AWS CLI 在源和目标 Outposts 桶由同一 AWS 账户拥有时设置复制，请执行以下操作：

- 创建源和目标 Outposts 桶。
- 对这两个桶启用版本控制。
- 创建向 S3 on Outposts 授予复制对象的权限的 IAM 角色。
- 将复制配置添加到源 Outposts 桶中。

要验证您的设置，请对其进行测试。

当源和目标 Outposts 桶由同一个 AWS 账户拥有时设置复制

1. 为 AWS CLI 设置凭证配置文件。在此示例中，我们使用配置文件名称 `acctA`。有关设置凭证配置文件的信息，请参阅《AWS Command Line Interface 用户指南》中的命名配置文件。

   > **Important**
   >
   > 用于执行此操作的配置文件必须具有必要的权限。例如，在复制配置中，可以指定 S3 on Outposts 可代入的 IAM 服务角色。仅当您所使用的配置文件具有 `iam:CreateRole` 和 `iam:PassRole` 权限时，您才能执行此操作。有关更多信息，请参阅《IAM 用户指南》中的向用户授予将角色传递给 AWS 服务的权限。如果您使用管理员凭证创建命名配置文件，命名配置文件将具有执行所有任务的必要权限。

2. 创建一个源存储桶，并对其启用版本控制。以下 `create-bucket` 命令在美国东部（弗吉尼亚州北部）（`us-east-1`）区域中创建一个 *`SOURCE-OUTPOSTS-BUCKET`* 桶。要使用此命令，请将 *`user input placeholders`* 替换为您自己的信息。

   ```
   aws s3control create-bucket --bucket SOURCE-OUTPOSTS-BUCKET --outpost-id SOURCE-OUTPOST-ID --profile acctA --region us-east-1
   ```

   以下 `put-bucket-versioning` 命令对于 *`SOURCE-OUTPOSTS-BUCKET`* 桶启用版本控制。要使用此命令，请将 *`user input placeholders`* 替换为您自己的信息。

   ```
   aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
   ```

3. 创建一个目标存储桶，并对其启用版本控制。以下 `create-bucket` 命令在美国西部（俄勒冈州）（`us-west-2`）区域中创建一个 *DESTINATION-OUTPOSTS-BUCKET* 桶。要使用此命令，请将 *user input placeholders* 替换为您自己的信息。

> **Note**
>
> 要在源和目标 Outposts 桶位于同一 AWS 账户中时设置复制配置，请使用同一命名配置文件。此示例使用 `acctA`。要在两个桶由不同 AWS 账户拥有时对复制配置进行测试，您需要为每个桶指定不同的配置文件。

```
aws s3control create-bucket --bucket DESTINATION-OUTPOSTS-BUCKET --create-bucket-configuration LocationConstraint=us-west-2 --outpost-id DESTINATION-OUTPOST-ID --profile acctA --region us-west-2
```

以下 `put-bucket-versioning` 命令对于 *DESTINATION-OUTPOSTS-BUCKET* 桶启用版本控制。要使用此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
```

4. 创建 IAM 服务角色。稍后在复制配置中，将此服务角色添加到 *SOURCE-OUTPOSTS-BUCKET* 桶中。S3 on Outposts 代入此角色以代表您复制对象。分两个步骤创建 IAM 角色：

   a. 创建 IAM 角色。

      i. 复制以下信任策略，并将其保存到本地计算机上当前目录中一个名为 `s3-on-outposts-role-trust-policy.json` 的文件。此策略向 S3 on Outposts 服务主体授予代入该服务角色的权限。

      JSON

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

```
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

ii. 运行以下命令以创建角色。将 *user input placeholders* 替换为您自己的信息。

```
aws iam create-role --role-name replicationRole --assume-role-policy-
document file://s3-on-outposts-role-trust-policy.json --profile acctA
```

b. 将权限策略附加到服务角色。

i. 复制以下权限策略，并将其保存到本地计算机上当前目录中一个名为 s3-on-outposts-role-permissions-policy.json 的文件。此策略授予对各种 S3 on Outposts 桶和对象操作的权限。要使用这一策略，请将 *user input placeholders* 替换为您自己的信息。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3-outposts:GetObjectVersionForReplication",
                "s3-outposts:GetObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-
BUCKET/object/*",
                "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/SOURCE-OUTPOST-ID/accesspoint/SOURCE-
OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3-outposts:ReplicateObject",
                "s3-outposts:ReplicateDelete"
            ],
            "Resource": [
                "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/DESTINATION-OUTPOST-ID/
bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
                "arn:aws:s3-outposts:us-
east-1:123456789012:outpost/DESTINATION-OUTPOST-ID/
accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
            ]
        }
    ]
}
```

ii. 运行以下命令以创建策略并将其附加到角色。将 *user input placeholders* 替换为您自己的信息。

```
aws iam put-role-policy --role-name replicationRole --policy-
document file://s3-on-outposts-role-permissions-policy.json --policy-
name replicationRolePolicy --profile acctA
```

5. 将复制配置添加到 *SOURCE-OUTPOSTS-BUCKET* 桶中。

a. 尽管 S3 on Outposts API 要求复制配置采用 XML 格式，但 AWS CLI 要求您以 JSON 格式指定复制配置。将以下 JSON 保存到计算机上本地目录中一个名为 `replication.json` 文件。要使用此命令，请将 *user input placeholders* 替换为您自己的信息。

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket":
        "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-
ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT"
      }
    }
  ]
```

```
}
```

b. 运行以下 put-bucket-replication 命令以将复制配置添加到源 Outposts 桶中。要使用此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control put-bucket-replication --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-
ID/bucket/SOURCE-OUTPOSTS-BUCKET --replication-configuration file://
replication.json --profile acctA
```

c. 要检索复制配置，请使用 get-bucket-replication 命令。要使用此命令，请将 *user input placeholders* 替换为您自己的信息。

```
aws s3control get-bucket-replication --account-id 123456789012 --bucket
 arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET --profile acctA
```

6. 在 Amazon S3 控制台中测试该设置：

a. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。

b. 在 *SOURCE-OUTPOSTS-BUCKET* 存储桶中，创建一个名为 Tax 的文件夹。

c. 将示例对象添加到 *SOURCE-OUTPOSTS-BUCKET* 存储桶中的 Tax 文件夹。

d. 在 *DESTINATION-OUTPOSTS-BUCKET* 存储桶中，确认以下几点：

- S3 on Outposts 复制了对象。

> **Note**
>
> S3 on Outposts 复制对象所需的时间量取决于对象大小。有关如何查看复制状态的信息，请参阅获取复制状态信息。

- 在对象的属性选项卡中，复制状态设置为副本（将其标识为副本对象）。

管理您的复制

本节介绍 S3 on Outposts 上可用的其他复制配置选项、如何确定复制状态以及如何排查复制问题。有关核心复制配置的信息，请参阅设置复制。

主题

使用复制指标监控进度

Outposts 上的 S3 复制为复制配置中的复制规则提供了详细的指标。使用复制指标，您可以通过跟踪待复制的字节数、复制延迟的复制和待处理的操作数，以 5 分钟为间隔监控复制的进度。为了帮助排查任何配置问题，您还可以设置 Amazon EventBridge 来接收有关复制失败的通知。

启用复制指标后，Outposts 上的 S3 复制会将以下指标发布到 Amazon CloudWatch：

- 待复制的字节数 – 给定复制规则的待复制对象的总字节数。
- 复制延迟 – 对于给定的复制规则，复制目标桶落后于源桶的最大秒数。
- 待复制的操作 – 给定复制规则的等待复制的操作的数量。操作包括对象、删除标记和标签。

> **Note**
>
> Outposts 上的 S3 复制指标的费率与 CloudWatch 自定义指标费率相同。有关更多信息，请参阅 CloudWatch 定价。

获取复制状态信息

复制状态可以帮助您确定 Amazon S3 on Outposts 正在复制的对象的当前状态。源对象的复制状态将返回 `PENDING`、`COMPLETED`、或 `FAILED`。副本的复制状态将返回 `REPLICA`。

复制状态概述

在复制场景中，您有一个源桶（对它配置复制）和一个目标桶（S3 on Outposts 将对象复制到其中）。当您请求这些桶中的对象（使用 `GetObject`）或对象元数据（使用 `HeadObject`）时，S3 on Outposts 将在响应中返回 `x-amz-replication-status` 标头，如下所示：

- 如果请求源桶中的对象，S3 on Outposts 将返回 `x-amz-replication-status` 标头（如果请求中的对象符合复制条件）。

例如，假设您在复制配置中指定了对象前缀 `TaxDocs`，以指示 S3 on Outposts 仅复制键名称前缀为 `TaxDocs` 的对象。您上传的具有此键名前缀的任何对象（例如 `TaxDocs/document1.pdf`）都将复制。对于具有此键名称前缀的对象请求，S3 on Outposts 会返回其对象复制状态为以下值之一的 `x-amz-replication-status` 标头：`PENDING`、`COMPLETED` 或 `FAILED`。

> **Note**
>
> 如果在上传对象后对象复制失败，您无法重试复制。您必须重新上传对象。由于缺少复制角色权限或缺少桶权限等问题，对象将转换为 `FAILED` 状态。对于临时故障（例如，如果桶或 Outpost 不可用），复制状态不会转换为 `FAILED`，而是保持 `PENDING`。在资源恢复联机后，S3 on Outposts 将恢复复制这些对象。

- 当请求目标桶中的对象时，如果请求中的对象是 S3 on Outposts 创建的副本，S3 on Outposts 会返回值为 `REPLICA` 的 `x-amz-replication-status` 标头。

> **Note**
>
> 在从启用了复制的源存储桶中删除对象之前，请检查该对象的复制状态，以确保已复制了该对象。

在 Outposts 上启用了 Amazon S3 副本修改同步时的复制状态

当您的复制规则启用了 S3 on Outposts 副本修改同步时，副本可能报告 `REPLICA` 以外的状态。如果元数据更改正在复制过程中，副本的 `x-amz-replication-status` 标头将返回 `PENDING`。如果副本修改同步无法复制元数据，副本的标头将返回 `FAILED`。如果正确复制元数据，则副本的标头将返回值 `REPLICA`。

对复制进行问题排查

如果在您配置复制之后，对象副本未出现在目标 Amazon S3 on Outposts 桶中，请使用以下问题排查提示确定并修复问题。

- S3 on Outposts 复制对象所需的时间取决于多个因素，包括源和目标 Outposts 之间的距离，以及对象的大小。

  您可以检查源对象的复制状态。如果对象的复制状态为 `PENDING`，表示 S3 on Outpost 尚未完成复制。如果对象的复制状态为 `FAILED`，则应检查对源桶设置的复制配置。

- 在源桶上的复制配置中，验证以下几点：
  - 目标桶的接入点 Amazon 资源名称（ARN）是正确的。
  - 键名前缀是否正确。例如，如果将配置设置为复制具有前缀 `Tax` 的对象，则仅复制具有 `Tax/document1` 或 `Tax/document2` 等键名的对象。不会复制具有键名 `document3` 的对象。
  - 状态是否为 `Enabled`。
- 验证没有在任何一个桶上暂停版本控制。源桶和目标桶必须均已启用版本控制。
- 如果目标桶由另一个 AWS 账户拥有，请验证桶拥有者是否对目标桶设置了允许源桶拥有者复制对象的桶策略。有关示例，请参阅在源和目标 Outposts 桶由不同的 AWS 账户拥有时授予权限。
- 如果对象副本未出现在目标桶中，以下问题可能会阻止复制：
  - 如果源桶中的某对象是另一个复制配置所创建的副本，S3 on Outposts 不会复制该对象。例如，如果您设置从桶 A 到桶 B 再到桶 C 的复制配置，则 S3 on Outposts 不会将桶 B 中的对象副本复制到桶 C。

    如果您想将桶 A 中的对象复制到桶 B 和桶 C，请在不同的复制规则中为源桶复制配置设置多个桶目标。例如，在源桶 A 上创建两条复制规则，其中一条规则旨在复制到目标桶 B，另一条规则旨在复制到目标桶 C。
  - 源桶拥有者可以向其他 AWS 账户授予上载对象的权限。默认情况下，源桶拥有者对于其他账户创建的对象没有权限。复制配置仅复制源桶拥有者对其具有访问权限的对象。为避免复制问题，源桶拥有者可以向其他 AWS 账户授予有条件地创建对象的权限，从而要求针对这些对象的显式访问权限。
- 假设在复制配置中，您添加了一个规则以复制具有特定标签的对象子集。在这种情况下，您必须在创建对象时分配特定的标签键和值，以便 S3 on Outposts 复制对象。如果您先创建对象，然后向该现有对象添加标签，S3 on Outposts 将不会复制对象。
- 如果桶策略拒绝针对以下任一操作访问复制角色，则复制将失败：

  源桶：

```
"s3-outposts:GetObjectVersionForReplication",
"s3-outposts:GetObjectVersionTagging"
```

  目标桶：

```
"s3-outposts:ReplicateObject",
"s3-outposts:ReplicateDelete",
"s3-outposts:ReplicateTags"
```

- 当对象无法复制到其目标 Outposts 时，Amazon EventBridge 可以通知您。有关更多信息，请参阅将 EventBridge 用于 Outposts 上的 S3 复制。

将 EventBridge 用于 Outposts 上的 S3 复制

Amazon S3 on Outposts 已与 Amazon EventBridge 集成，并使用 `s3-outposts` 命名空间。EventBridge 是一种无服务器事件总线服务，您可以使用它将应用程序与来自各种来源的数据相连接。有关更多信息，请参阅 Amazon EventBridge 用户指南中的什么是 Amazon EventBridge？。

为了帮助解决任何复制配置问题，您可以设置 Amazon EventBridge 来接收有关复制失败事件的通知。在对象未复制到其目标 Outposts 的情况下，EventBridge 可以通知您。有关正在复制的对象的当前状态的更多信息，请参阅复制状态概述。

每当 Outposts 桶中发生某些事件时，S3 on Outposts 都可以将事件发送到 EventBridge。与其他目标不同，您无需选择要提供的事件类型。还可以使用 EventBridge 规则将事件路由到其他目标。启用 EventBridge 后，S3 on Outposts 会将以下所有事件发送到 EventBridge。

Event type	说明	命名空间
`Operation FailedReplication`	复制规则内的对象复制已失败。有关 Outposts 上的 S3 复制失败原因的更多信息，请参阅使用 EventBridge 查看 Outposts 上的 S3 复制失败原因。	`s3-outposts`

使用 EventBridge 查看 Outposts 上的 S3 复制失败原因

下表列出了 Outposts 上的 S3 复制失败原因。您可以将 EventBridge 规则配置为通过 Amazon Simple Queue Service（Amazon SQS）、Amazon Simple Notification Service（Amazon SNS）、AWS Lambda 或 Amazon CloudWatch Logs 发布和查看失败原因。有关将这些资源用于 EventBridge 所需权限的更多信息，请参阅将基于资源的策略用于 EventBridge。

复制失败原因	说明
`AssumeRoleNotPermitted`	S3 on Outposts 无法代入在复制配置中指定的 AWS Identity and Access Management（IAM）角色。

复制失败原因	说明
DstBucketNotFound	S3 on Outposts 找不到在复制配置中指定的目标桶。
DstBucketUnversioned	未在 Outposts 目标桶上启用版本控制。要使用 Outposts 上的 S3 复制来复制对象，必须在目标桶上启用版本控制。
DstDelObjNotPermitted	S3 on Outposts 无法将删除操作复制到目标桶。针对目标桶的 s3-outposts:ReplicateDelete 权限可能缺失。
DstMultipartCompleteNotPermitted	S3 on Outposts 无法在目标桶中完成对象的分段上传。针对目标桶的 s3-outposts:ReplicateObject 权限可能缺失。
DstMultipartInitNotPermitted	S3 on Outposts 无法启动将对象分段上传到目标桶。针对目标桶的 s3-outposts:ReplicateObject 权限可能缺失。
DstMultipartPartUploadNotPermitted	S3 on Outposts 无法在目标存储桶中上传分段上传对象。针对目标桶的 s3-outposts:ReplicateObject 权限可能缺失。
DstOutOfCapacity	因为目标 Outpost 的 S3 存储容量不足，所以 S3 on Outposts 无法复制到此 Outpost。
DstPutObjNotPermitted	S3 on Outposts 无法将对象复制到目标桶。针对目标桶的 s3-outposts:ReplicateObject 权限可能缺失。
DstPutTaggingNotPermitted	S3 on Outposts 无法将对象标签复制到目标桶。针对目标桶的 s3-outposts:ReplicateObject 权限可能缺失。
DstVersionNotFound	S3 on Outposts 无法在目标桶中找到所需的对象版本，从而无法复制该对象版本的元数据。

复制失败原因	说明
SrcBucketReplicationConfigMissing	S3 on Outposts 找不到与源 Outposts 桶关联的接入点的复制配置。
SrcGetObjNotPermitted	S3 on Outposts 无法访问源桶中的对象来进行复制。针对源桶的 s3-outposts:GetObjectVersionForReplication 权限可能缺失。
SrcGetTaggingNotPermitted	S3 on Outposts 无法访问源桶中的对象标签信息。针对源桶的 s3-outposts:GetObjectVersionTagging 权限可能缺失。
SrcHeadObjectNotPermitted	S3 on Outposts 无法从源桶检索对象元数据。针对源桶的 s3-outposts:GetObjectVersionForReplication 权限可能缺失。
SrcObjectNotEligible	对象不符合复制条件。对象或其对象标签与复制配置不匹配。

有关排查复制问题的更多信息，请参阅以下主题：

- 创建 IAM 角色
- 对复制进行问题排查

使用 CloudWatch 监控 EventBridge

为进行监控，Amazon EventBridge 与 Amazon CloudWatch 相集成。EventBridge 会自动每分钟向 CloudWatch 发送指标。这些指标包括规则匹配的事件数以及规则调用目标的次数。当规则在 EventBridge 中运行时，将调用与该规则关联的所有目标。您可以按照以下方式通过 CloudWatch 监控 EventBridge 行为。

- 您可以从 CloudWatch 控制面板监控 EventBridge 规则的可用 EventBridge 指标。然后，您可以使用 CloudWatch 功能（例如 CloudWatch 告警）对某些指标设置告警。如果这些指标达到您在告警中指定的自定义阈值，您将收到通知并可以相应地采取措施。

- 可以将 Amazon CloudWatch Logs 设置为 EventBridge 规则的目标。然后，EventBridge 创建日志流，而 CloudWatch Logs 将事件中的文本存储为日志条目。有关更多信息，请参阅 EventBridge 和 CloudWatch Logs。

有关调试 EventBridge 事件交付和存档事件的更多信息，请参阅以下主题：

- 事件重试策略和使用死信队列
- 存档 EventBridge 事件

通过使用 AWS RAM 共享 S3 on Outposts

Amazon S3 on Outposts 支持通过使用 AWS Resource Access Manager (AWS RAM) 跨一个企业内的多个账户共享 S3 容量。借助 S3 on Outposts 共享，您可以允许其他人在您的 Outpost 上创建和管理存储桶、端点和访问点。

本主题演示如何使用 AWS RAM 与您的 AWS 企业中的另一个 AWS 账户 共享 S3 on Outposts 和相关资源。

先决条件

- Outpost 拥有者账户在 AWS Organizations 中配置了一个企业。有关更多信息，请参阅 AWS Organizations 用户指南中的创建企业。
- 该企业包括您想与之共享 S3 on Outposts 容量的 AWS 账户。有关更多信息，请参阅 AWS Organizations 用户指南中的向 AWS 账户 发送邀请。
- 请选择要共享的以下选项之一。必须选择第二个资源（Subnets（子网）或 Outposts），以便端点也可供访问。端点是网络要求，以便访问 S3 on Outposts 中存储的数据。

选项 1	选项 2
S3 on Outposts	S3 on Outposts
允许用户在您的 Outposts 和访问点上创建存储桶，并将对象添加到这些存储桶。	允许用户在您的 Outposts 和访问点上创建存储桶，并将对象添加到这些存储桶。
子网	Outposts

选项 1	选项 2
允许用户使用您的 Virtual Private Cloud (VPC) 和与您的子网关联的端点。	允许用户查看 S3 容量图表和 AWS Outposts 控制台主页。还允许用户在共享的 Outposts 上创建子网并创建端点。

过程

1. 使用拥有 Outpost 的 AWS 账户登录到 AWS 管理控制台，然后打开 AWS RAM 控制台，地址为 https://console.aws.amazon.com/ram/home。
2. 确保您已在 AWS RAM 中启用与 AWS Organizations 共享。有关更多信息，请参阅 AWS RAM 用户指南中的在 AWS Organizations 中启用资源共享。
3. 使用先决条件中的选项 1 或选项 2 创建资源共享。如果您有多个 S3 on Outposts 资源，请选择要共享的资源的 Amazon 资源名称 (ARN)。要启用端点，请共享您的子网或 Outpost。

   有关如何创建资源共享的更新信息，请参阅 AWS RAM 用户指南中的创建资源共享。
4. 您与之共享资源的 AWS 账户 现在应该能够使用 S3 on Outposts 了。根据您在先决条件中选择的选项，向账户用户提供以下信息：

选项 1	选项 2
Outpost ID VPC ID 子网 ID 安全组 ID	Outpost ID

> **Note**
>
> 用户可以使用 AWS RAM 控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 确认已与他们共享资源。用户可以使用 get-resource-shares CLI 命令查看他们的现有资源共享。

用法示例

在您与另一个账户共享 S3 on Outposts 资源后，该账户可以管理 Outpost 上的存储桶和对象。如果您共享了 Subnets（子网）资源，那么该账户可以使用您创建的端点。以下示例演示了用户如何使用 AWS CLI 在您共享这些资源后与您的 Outpost 互动。

Example：创建存储桶

以下示例在 Outpost *op-01ac5d28a6a232904* 上创建名为 *amzn-s3-demo-bucket1* 的存储桶。在使用此命令之前，请针对您的使用案例将每个 *user input placeholder* 替换为适合的值。

```
aws s3control create-bucket --bucket amzn-s3-demo-bucket1 --outpost-
id op-01ac5d28a6a232904
```

有关此命令的更多信息，请参阅《AWS CLI 命令参考》中的 create-bucket。

Example：创建访问点

以下示例使用下表中的示例参数在 Outpost 上创建访问点。在使用此命令之前，请针对您的使用案例将这些 *user input placeholder* 值和 AWS 区域 代码替换为适合的值。

参数	值
账户 ID	*111122223333*
访问点名称	*example-outpost-access-point*
Outpost ID	*op-01ac5d28a6a232904*
Outpost 存储桶名称	*amzn-s3-demo-bucket1*
VPC ID	*vpc-1a2b3c4d5e6f7g8h9*

> **Note**
>
> 账户 ID 参数必须是存储桶拥有者的 AWS 账户 ID，即共享用户。

```
aws s3control create-access-point --account-id 111122223333 --name example-outpost-
access-point \
```

```
--bucket arn:aws:s3-outposts:us-east-1:111122223333:outpost/op-01ac5d28a6a232904/
bucket/amzn-s3-demo-bucket1 \
--vpc-configuration VpcId=vpc-1a2b3c4d5e6f7g8h9
```

有关此命令的更多信息，请参阅《AWS CLI 参考》中的 create-access-point 。

Example：上传对象

以下示例将通过 AWS 账户 *111122223333* 拥有的 Outpost *op-01ac5d28a6a232904* 上的访问点 *example-outpost-access-point*，将文件 *my_image.jpg* 从用户的本地文件系统上传到名为 *images/my_image.jpg* 的对象。在使用此命令之前，请针对您的使用案例将这些 *user input placeholder* 值和 AWS 区域 代码替换为适合的值。

```
aws s3api put-object --bucket arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point \
--body my_image.jpg --key images/my_image.jpg
```

有关此命令的更多信息，请参阅《AWS CLI 参考》中的 put-object。

> **Note**
>
> 如果此操作导致未找到资源错误或无响应，则您的 VPC 可能没有共享端点。
> 要检查是否存在共享端点，请使用 list-shared-endpoints AWS CLI 命令。如果没有共享端点，请与 Outpost 拥有者合作创建一个。有关更多信息，请参阅《Amazon Simple Storage Service API 参考》中的 ListSharedEndpoints。

Example：创建端点

以下示例在共享的 Outpost 上创建端点。在使用此命令之前，请针对您的使用案例将 Outpost ID、子网 ID 和安全组 ID 的 *user input placeholder* 值替换为适合的值。

> **Note**
>
> 只有在资源共享中包含 Outposts 资源时，用户才可以执行此操作。

```
aws s3outposts create-endpoint --outposts-id op-01ac5d28a6a232904 --subnet-id XXXXXX --
security-group-id XXXXXXX
```

有关此命令的更多信息，请参阅《AWS CLI 参考》中的 create-endpoint。

使用 S3 on Outposts 的其他 AWS 服务

在您的 AWS Outposts 本地运行的其他 AWS 服务也可以使用 Amazon S3 on Outposts 容量。在 Amazon CloudWatch 中，`S3Outposts` 命名空间显示 S3 on Outposts 中存储桶的详细指标，但这些指标不包括其他 AWS 服务的使用情况。要管理其他 AWS 服务使用的 S3 on Outposts 容量，请参阅下表中的信息。

AWS 服务	说明	了解更多
Amazon S3	所有直接使用 S3 on Outposts 的情况都具有匹配的账户和存储桶 CloudWatch 指标。	请参阅指标
Amazon Elastic Block Store (Amazon EBS)	对于 Amazon EBS on Outposts，您可以选择 AWS Outpost 作为快照目的地，并将其存储在 S3 on Outpost 本地。	了解详情
Amazon Relationa l Database Service (Amazon RDS)	您可以使用 Amazon RDS 本地备份将 RDS 备份存储在 Outpost 本地。	了解详情

监控 S3 on Outposts

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅 什么是 Amazon S3 on Outposts？。

有关监控 Amazon S3 on Outposts 存储容量的更多信息，请参阅以下主题。

主题

- 使用 Amazon CloudWatch 指标管理 S3 on Outposts 容量
- 使用 Amazon CloudWatch Events 接收 S3 on Outposts 事件通知
- 使用 AWS CloudTrail 日志监控 S3 on Outposts

使用 Amazon CloudWatch 指标管理 S3 on Outposts 容量

为帮助管理 Outpost 上的固定 S3 容量，我们建议您创建 CloudWatch 提示，在存储利用率超过特定阈值时向您发出提示。有关 S3 on Outposts 的 CloudWatch 指标的更多信息，请参阅CloudWatch 指标。如果没有足够的空间在您的 Outpost 上存储对象，API 将返回容量不足异常 (ICE)。要释放空间，您可以创建触发显式数据删除的 CloudWatch 告警，或者使用生命周期过期策略使对象过期。为在删除数据之前保存数据，您可以使用 AWS DataSync 将数据从 Amazon S3 on Outposts 桶复制到 AWS 区域中的 S3 桶。有关使用 DataSync 的更多信息，请参阅《AWS DataSync 用户指南》中的 AWS DataSync 入门。

CloudWatch 指标

`S3Outposts` 命名空间包括 Amazon S3 on Outposts 桶的以下指标。您可以监控给定桶的总预置 S3 on Outposts 字节数、适用于对象的总可用字节数以及所有对象的总大小。所有直接使用 S3 的情况都存在与桶或账户相关的指标。间接使用 S3（例如在 Outpost 上存储 Amazon Elastic Block Store 本地快照或Amazon Relational Database Service 备份）会消耗 S3 容量，但不包括在桶或账户相关指标中。有关 Amazon EBS 本地快照的更多信息，请参阅 Outposts 上的 Amazon EBS 本地快照。要查看您的 Amazon EBS 成本报告，请访问 https://console.aws.amazon.com/costmanagement/。

> **Note**
>
> S3 on Outposts 仅支持以下指标，而不支持其他 Amazon S3 指标。
> 由于 S3 on Outposts 具有固定容量限制，因此我们建议您创建 CloudWatch 告警，以便在存储利用率超过特定阈值时向您发出通知。

指标	说明	时段	单位	类型
`Outpost` `talBytes`	Outpost 的预置总容量（以字节为单位）。	5 分钟	字节	S3 on Outposts
`Outpostl` `eeBytes`	Outpost 上可用于存储客户数据的可用字节数。	5 分钟	字节	S3 on Outposts
`BucketU` `dBytes`	给定桶的所有对象的总大小。	5 分钟	字节	S3 on Outposts。仅限直接使用 S3。
`Accountl` `edBytes`	指定的 Outposts 账户中所有对象的总大小。	5 分钟	字节	S3 on Outposts。仅限直接使用 S3。

`BytesPe`
`ingRepl`
`ation`
	给定复制规则的待复制对象的总字节数。有关如何启用复制指标的更多信息，请参阅在 Outposts 之间创建复制规则。	5 分钟	字节	可选。适用于 Outposts 上的 S3 复制。

`Operati`
`sPendin`
`eplicat`
`n`
	给定复制规则的待复制操作的总数。有关如何启用复制指标的更多信息，请参阅在 Outposts 之间创建复制规则。	5 分钟	计数	可选。适用于 Outposts 上的 S3 复制。

`Replica` `onLaten`	对于给定的复制规则，复制目标桶落后于源桶的当前延迟秒数。有关如何启用复制指标的更多信息，请参阅在 Outposts 之间创建复制规则。	5 分钟	秒	可选。适用于 Outposts 上的 S3 复制。

使用 Amazon CloudWatch Events 接收 S3 on Outposts 事件通知

您可以使用 CloudWatch Events 为任何 Amazon S3 on Outposts API 事件创建规则。创建规则时，您可以选择通过所有支持的 CloudWatch 目标获得通知，包括 Amazon Simple Queue Service（Amazon SQS）、Amazon Simple Notification Service（Amazon SNS）和 AWS Lambda。有关更多信息，请参阅《Amazon CloudWatch Events 用户指南》中的可以作为 CloudWatch Events 目标的 AWS 服务列表。要选择目标服务以与 S3 on Outposts 结合使用，请参阅《Amazon CloudWatch Events 用户指南》中的使用 AWS CloudTrail 创建对 AWS API 调用触发的 CloudWatch Events 规则。

> **Note**
>
> 对于 S3 on Outposts 对象操作，只有将跟踪（可选带有事件选择器）配置为接收 CloudTrail 发送的 AWS API 调用事件时，这些事件才会匹配您的规则。有关更多信息，请参阅e AWS CloudTrail《用户指南》 中的 使用 CloudTrail 日志文件。

Example

以下是 `DeleteObject` 操作的示例规则。要使用此示例规则，请将 *amzn-s3-demo-bucket1* 替换为 S3 on Outposts 桶的名称。

```
{
  "source": [
    "aws.s3-outposts"
  ],
  "detail-type": [
    "AWS API call through CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3-outposts.amazonaws.com"
    ],
    "eventName": [
      "DeleteObject"
    ],
    "requestParameters": {
      "bucketName": [
        "amzn-s3-demo-bucket1"
      ]
    }
  }
```

```
}
```

使用 AWS CloudTrail 日志监控 S3 on Outposts

Amazon S3 on Outposts 与 AWS CloudTrail 集成，后者是在 S3 on Outposts 中提供用户、角色或 AWS 服务所采取操作的记录的服务。您可以使用 AWS CloudTrail 获取关于 S3 on Outposts 桶级别请求以及对象级别请求的信息，这些请求旨在审计和记录您的 S3 on Outposts 事件活动。

要为所有 Outposts 桶或特定 Outposts 桶的列表启用 CloudTrail 数据事件，您必须在 CloudTrail 中手动创建跟踪。有关 CloudTrail 日志文件条目的更多信息，请参阅 S3 on Outposts 日志文件条目。

有关 S3 on Outposts 的 CloudTrail 数据事件的完整列表，请参阅《Amazon S3 用户指南》中的 CloudTrail 中的 Amazon S3 数据事件。

> **Note**
>
> - 最佳实践是为 AWS CloudTrail 数据事件 Outposts 桶创建生命周期策略。配置生命周期策略，以便在您需要审计日志文件的时间段之后定期删除这些日志文件。这样做可以减少 Amazon Athena 为每个查询分析的数据量。有关更多信息，请参阅 为 Amazon S3 on Outposts 桶创建和管理生命周期配置。
> - 有关如何查询 CloudTrail 日志的示例，请参阅 AWS 大数据博客文章使用 AWS CloudTrail 和 Amazon Athena 分析安全性、合规性和操作活动。

为 S3 on Outposts 桶中的对象启用 CloudTrail 日志记录

您可以使用 Amazon S3 控制台配置 AWS CloudTrail 跟踪来记录 Amazon S3 on Outposts 桶中对象的数据事件。CloudTrail 支持记录 S3 on Outposts 对象级别 API 操作，例如 `GetObject`、`DeleteObject` 和 `PutObject`。这些事件称为数据事件。

原定设置情况下，CloudTrail 跟踪不记录数据事件。但是，可以将跟踪配置为记录您指定的 S3 on Outposts 存储桶的数据事件，或记录 AWS 账户中所有 S3 on Outposts 存储桶的数据事件。

CloudTrail 不会在 CloudTrail 事件历史记录中填充数据事件。此外，并非所有 S3 on Outposts 桶级别 API 操作都会填充在 CloudTrail 事件历史记录中。有关如何查询 CloudTrail 日志的更多信息，请参阅 AWS 知识中心上的使用 Amazon CloudWatch Logs 筛选条件模式和 Amazon Athena 查询 CloudTrail 日志。

要配置跟踪以记录某个 S3 on Outposts 桶的数据事件，您可以使用 AWS CloudTrail 控制台或 Amazon S3 控制台。如果您要配置跟踪以记录您的 AWS 账户中所有 S3 on Outposts 桶的数据事件，使用 CloudTrail 控制台会更轻松。有关使用 CloudTrail 控制台配置跟踪以记录 S3 on Outposts 数据事件的信息，请参阅《AWS CloudTrail 用户指南》中的数据事件。

> **Important**
>
> 记录数据事件将收取额外费用。有关更多信息，请参阅 AWS CloudTrail 定价。

以下过程演示如何使用 Amazon S3 控制台配置 CloudTrail 跟踪来记录 S3 on Outposts 桶的数据事件。

> **Note**
>
> 创建桶的 AWS 账户拥有该桶，并且是唯一可以配置要发送到 AWS CloudTrail 的 S3 on Outposts 数据事件的账户。

为 S3 on Outposts 桶中的对象启用 CloudTrail 数据事件日志记录

1. 登录到 AWS 管理控制台，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在左侧导航窗格中，选择 Outposts buckets（Outposts 桶）。
3. 选择要使用 CloudTrail 记录其数据事件的 Outposts 桶的名称。
4. 选择属性。
5. 在 AWS CloudTrail 数据事件部分中，选择在 CloudTrail 中配置。

   AWS CloudTrail 控制台将打开。

   您可以创建新的 CloudTrail 跟踪或重用现有跟踪，并配置要在跟踪中记录的 S3 on Outposts 数据事件。
6. 在 CloudTrail 控制台的控制面板页面上，选择创建跟踪。
7. 在步骤 1 选择跟踪属性页面上，提供跟踪的名称，选择用于存储跟踪日志的 S3 桶，指定所需的任何其他设置，然后选择下一步。
8. 在步骤 2 选择日志事件页面的事件类型下，选择数据事件。

对于数据事件类型，选择 S3 Outposts。选择下一步。

> **Note**
>
> - 当为 S3 on Outposts 创建跟踪和配置数据事件日志记录时，必须正确指定数据事件类型。
>   - 如果您使用 CloudTrail 控制台，请对于数据事件类型选择 S3 Outposts。有关如何在 CloudTrail 控制台中创建跟踪的信息，请参阅 AWS CloudTrail 用户指南中的使用控制台创建和更新跟踪。有关如何在 CloudTrail 控制台中配置 S3 on Outposts 数据事件日志记录的信息，请参阅《AWS CloudTrail 用户指南》中的记录 Amazon S3 对象的数据事件。
>   - 如果您使用 AWS Command Line Interface（AWS CLI）或 AWS SDK，请将 `resources.type` 字段设置为 `AWS::S3Outposts::Object`。有关如何使用 AWS CLI 记录 S3 on Outposts 数据事件的更多信息，请参阅《AWS CloudTrail 用户指南》中的记录 S3 on Outposts 事件。
> - 如果您使用 CloudTrail 控制台或 Amazon S3 控制台配置某个跟踪以记录 S3 on Outposts 桶的数据事件，Amazon S3 控制台将显示已为该桶启用对象级别日志记录。

9. 在步骤 3 查看和创建页面上，查看您配置的跟踪属性和日志事件。然后，选择创建跟踪。

为 S3 on Outposts 桶中的对象禁用 CloudTrail 数据事件日志记录

1. 登录到 AWS 管理控制台，然后通过以下网址打开 CloudTrail 控制台：https://console.aws.amazon.com/cloudtrail。
2. 在左侧导航窗格中，选择跟踪。
3. 选择您创建用于记录 S3 on Outposts 桶的事件的跟踪名称。
4. 在跟踪的详细信息页面上，选择右上角的停止记录。
5. 在随后显示的对话框中，选择停止记录。

Amazon S3 on Outposts AWS CloudTrail 日志文件条目

Amazon S3 on Outposts 管理事件可通过 AWS CloudTrail 提供。此外，您还可以选择性地为 AWS CloudTrail 中的数据事件启用日志记录。

跟踪是一种配置，可用于将事件作为日志文件传送到您指定的区域的 S3 存储桶中。Outposts 存储桶的 CloudTrail 日志包括一个新字段 `edgeDeviceDetails`，该字段用于识别指定存储桶所在的 Outpost。

其他日志字段包括请求的操作、操作的日期和时间以及请求参数。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序显示。

下面的示例显示了一个 CloudTrail 日志条目，该条目说明了 `s3-outposts` 上的 PutObject 操作。


```
{
      "eventVersion": "1.08",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:user/yourUserName",
        "accountId": "222222222222",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "yourUserName"
      },
      "eventTime": "2020-11-30T15:44:33Z",
      "eventSource": "s3-outposts.amazonaws.com",
      "eventName": "PutObject",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "26.29.66.20",
      "userAgent": "aws-cli/1.18.39 Python/3.4.10 Darwin/18.7.0 botocore/1.15.39",
      "requestParameters": {
        "expires": "Wed, 21 Oct 2020 07:28:00 GMT",
        "Content-Language": "english",
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "ObjectCannedACL": "BucketOwnerFullControl",
        "x-amz-server-side-encryption": "Aes256",
        "Content-Encoding": "gzip",
        "Content-Length": "10",
        "Cache-Control": "no-cache",
        "Content-Type": "text/html; charset=UTF-8",
        "Content-Disposition": "attachment",
        "Content-MD5": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY",
        "x-amz-storage-class": "Outposts",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "bucketName": "amzn-s3-demo-bucket1",
        "Key": "path/upload.sh"
      },
```

```
      "responseElements": {
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "x-amz-server-side-encryption": "Aes256",
        "x-amz-version-id": "001",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "ETag": "d41d8cd98f00b204e9800998ecf8427f"
      },
      "additionalEventData": {
        "CipherSuite": "ECDHE-RSA-AES128-SHA",
        "bytesTransferredIn": 10,
        "x-amz-id-2": "29xXQBV20
+xOHKItvzY1suLv1i6A52E0zOX159fpfsItYd58JhXwKxXAXI4IQkp6",
        "SignatureVersion": "SigV4",
        "bytesTransferredOut": 20,
        "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "8E96D972160306FA",
      "eventID": "ee3b4e0c-ab12-459b-9998-0a5a6f2e4015",
      "readOnly": false,
      "resources": [
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Object",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/path/upload.sh"
        },
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Bucket",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/"
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "444455556666",
      "sharedEventID": "02759a4c-c040-4758-b84b-7cbaaf17747a",
      "edgeDeviceDetails": {
        "type": "outposts",
        "deviceId": "op-01ac5d28a6a232904"
      },
      "eventCategory": "Data"
```

```
    }
```

使用 Amazon S3 on Outposts 进行开发

通过使用 Amazon S3 on Outposts，您可以在 AWS Outposts 上创建 S3 桶，并在本地为需要本地数据访问、本地数据处理和数据驻留的应用程序轻松存储和检索对象。S3 on Outposts 提供了一个新的存储类 S3 Outposts (OUTPOSTS)；该存储类使用 Amazon S3 API，并且用于在 AWS Outposts 上的多个设备和服务器之间持久冗余地存储数据。您通过 Virtual Private Cloud（VPC）使用接入点和端点连接与 Outposts 桶进行通信。您可以像在 Amazon S3 桶中一样在 Outpost 桶上使用相同的 API 和功能，包括访问策略、加密和标记。您可以通过 AWS 管理控制台、AWS Command Line Interface (AWS CLI)、AWS SDK 或 REST API 使用 S3 on Outposts。有关更多信息，请参阅什么是 Amazon S3 on Outposts？

以下主题提供有关使用 S3 on Outposts 进行开发的信息。

主题

支持 S3 on Outposts 的区域

以下 AWS 区域支持 S3 on Outposts。

- 美国东部（弗吉尼亚州北部）(us-east-1)
- 美国东部（俄亥俄州）(us-east-2)
- 美国西部（加利福尼亚北部）(us-west-1)
- 美国西部（俄勒冈州）(us-west-2)
- 非洲（开普敦）(af-south-1)
- 亚太地区（雅加达）（ap-sourtheast-3）
- 亚太地区（孟买）(ap-south-1)
- 亚太地区（大阪）(ap-northeast-3)
- 亚太地区（首尔）(ap-northeast-2)
- 亚太地区（新加坡）(ap-southeast-1)
- 亚太地区（悉尼）(ap-southeast-2)

- 亚太地区（东京）(ap-northeast-1)
- 加拿大（中部）(ca-central-1)
- 欧洲地区（法兰克福）(eu-central-1)
- 欧洲地区（爱尔兰）(eu-west-1)
- 欧洲（伦敦）(eu-west-2)
- 欧洲（米兰）(eu-south-1)
- 欧洲（巴黎）（eu-west-3）
- 欧洲地区（斯德哥尔摩）(eu-north-1)
- 以色列（特拉维夫）（il-central-1）
- 中东（巴林）(me-south-1)
- 南美洲（圣保罗）（sa-east-1）
- AWS GovCloud（美国东部）(us-gov-east-1)
- AWS GovCloud（美国西部）(us-gov-west-1)

Amazon S3 on Outposts API 操作

本主题列出了您可以与 Amazon S3 on Outposts 一起使用的 Amazon S3、Amazon S3 Control 和 Amazon S3 on Outposts API 操作。

主题

用于管理对象的 Amazon S3 API 操作

S3 on Outposts 旨在使用与 Amazon S3 相同的对象 API 操作。您必须使用访问点才能访问 Outpost 存储桶中的任何对象。将对象 API 操作与 S3 on Outposts 结合使用时，您需要提供 Outposts 访问点 Amazon 资源名称（ARN）或访问点别名。有关访问点别名的更多信息，请参阅为您的 S3 on Outposts 桶访问点使用桶式别名。

Amazon S3 on Outposts 支持以下Amazon S3 API 操作：

- AbortMultipartUpload

- CompleteMultipartUpload
- CopyObject
- CreateMultipartUpload
- DeleteObject
- DeleteObjects
- DeleteObjectTagging
- GetObject
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListObjectVersions
- ListParts
- PutObject
- PutObjectTagging
- UploadPart
- UploadPartCopy

用于管理存储桶的 Amazon S3 控制 API

S3 on Outposts 支持以下用于处理存储桶的 Amazon S3 控制 API 操作。

- CreateAccessPoint
- CreateBucket
- DeleteAccessPoint
- DeleteAccessPointPolicy
- DeleteBucket
- DeleteBucketLifecycleConfiguration
- DeleteBucketPolicy
- DeleteBucketReplication

- DeleteBucketTagging
- GetAccessPoint
- GetAccessPointPolicy
- GetBucket
- GetBucketLifecycleConfiguration
- GetBucketPolicy
- GetBucketReplication
- GetBucketTagging
- GetBucketVersioning
- ListAccessPoints
- ListRegionalBuckets
- PutAccessPointPolicy
- PutBucketLifecycleConfiguration
- PutBucketPolicy
- PutBucketReplication
- PutBucketTagging
- PutBucketVersioning

用于管理端点的 S3 on Outposts API 操作

S3 on Outposts 支持以下用于管理终端节点的 Amazon S3 on Outposts API 操作。

- CreateEndpoint
- DeleteEndpoint
- ListEndpoints
- ListOutpostsWithS3
- ListSharedEndpoints

使用适用于 Java 的 SDK 为 S3 on Outposts 配置 S3 控制客户端

以下示例使用 适用于 Java 的 AWS SDK 为 Amazon S3 on Outposts 配置 Amazon S3 控制客户端。要使用此示例，请将每个 *user input placeholder* 替换为您自己的信息。

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;

public AWSS3Control createS3ControlClient() {

    String accessKey = AWSAccessKey;
    String secretKey = SecretAccessKey;
    BasicAWSCredentials awsCreds = new BasicAWSCredentials(accessKey, secretKey);

    return AWSS3ControlClient.builder().enableUseArnRegion()
            .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
            .build();

}
```

通过 IPv6 向 S3 on Outposts 发出请求

Amazon S3 on Outposts 和 S3 on Outposts 双堆栈端点支持使用 IPv6 或 IPv4 协议向 S3 on Outposts 桶发出的请求。借助 S3 on Outposts 对 IPv6 的支持，可以通过 IPv6 网络使用 S3 on Outposts API 访问和操作桶以及控制面板资源。

> **Note**
>
> 不支持 IPv6 网络上的 S3 on Outposts 对象操作（例如 `PutObject` 或 `GetObject`）。

通过 IPv6 网络访问 S3 on Outposts 不额外收费。有关 S3 on Outposts 的更多信息，请参阅 S3 on Outposts 定价。

主题

- IPv6 入门
- 使用双堆栈端点通过 IPv6 网络发出请求
- 在 IAM 策略中使用 IPv6 地址
- 测试 IP 地址兼容性
- 将 IPv6 与 AWS PrivateLink 结合使用

- 使用 S3 on Outposts 双堆栈端点

IPv6 入门

要通过 IPv6 向 S3 on Outposts 桶发出请求，您必须使用双堆栈端点。下一节介绍如何使用双堆栈终端节点通过 IPv6 发出请求。

下面是在尝试通过 IPv6 访问 S3 on Outposts 桶之前的重要注意事项：

- 访问存储桶的客户端和网络必须支持使用 IPv6。
- 虚拟托管类型和路径类型请求必须都受支持，以便进行 IPv6 访问。有关更多信息，请参阅 使用 S3 on Outposts 双堆栈端点。
- 如果您在 AWS Identity and Access Management（IAM）用户策略或 S3 on Outposts 桶策略中使用源 IP 地址筛选，则必须更新策略以包括 IPv6 地址范围。

> **Note**
>
> 此要求仅适用于跨 IPv6 网络的 S3 on Outposts 桶操作和控制面板资源。不支持跨 IPv6 网络的 Amazon S3 on Outposts 对象操作。

- 如果使用 IPv6，服务器访问日志文件以 IPv6 格式输出 IP 地址。您必须对用于分析 S3 on Outposts 日志文件的现有工具、脚本和软件进行更新，以便它们能够分析 IPv6 格式的远程 IP 地址。然后，更新的工具、脚本和软件将正确解析 IPv6 格式的远程 IP 地址。

使用双堆栈端点通过 IPv6 网络发出请求

要通过 IPv6 使用 S3 on Outposts API 调用发出请求，可以通过 AWS CLI 或 AWS SDK 使用双堆栈端点。无论您是通过 IPv6 协议还是 IPv4 协议访问 S3 on Outposts，Amazon S3 控制 API 操作和 S3 on Outposts API 操作的工作方式都是相同的。但请注意，IPv6 网络不支持 S3 on Outposts 对象操作（例如 `PutObject` 或 `GetObject`）。

如果使用 AWS Command Line Interface（AWS CLI）和 AWS SDK，可使用参数或标志以更改为双堆栈端点。您还可以在配置文件中直接将双堆栈端点指定为覆盖 S3 on Outposts 端点。

您可以通过以下任一方式，使用双堆栈端点通过 IPv6 访问 S3 on Outposts 桶：

- AWS CLI，请参阅从 AWS CLI 使用双堆栈端点。

- 有关 AWS 开发工具包，请参阅 从 AWS SDK 使用 S3 on Outposts 双堆栈端点。

在 IAM 策略中使用 IPv6 地址

在尝试使用 IPv6 协议访问 S3 on Outposts 桶之前，请确保用于 IP 地址筛选的 IAM 用户策略或 S3 on Outposts 桶策略已更新为包括 IPv6 地址范围。如果未更新 IP 地址筛选策略以便处理 IPv6 地址，则在尝试使用 IPv6 协议时，您可能会无法访问 S3 on Outposts 桶。

筛选 IP 地址的 IAM policy 使用 IP 地址条件运算符。下面的 S3 on Outposts 桶策略通过使用 IP 地址条件运算符，确定允许的 IPv4 地址的 IP 范围为 54.240.143.*。此范围之外的所有 IP 地址对 S3 on Outposts 桶（DOC-EXAMPLE-BUCKET）的访问都会被拒绝。由于所有 IPv6 地址都在允许范围之外，此策略会阻止 IPv6 地址访问 DOC-EXAMPLE-BUCKET。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/OUTPOSTS-ID/bucket/DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "54.240.143.0/24"
                }
            }
        }
    ]
}
```

您可以将 S3 on Outposts 桶策略的 Condition 元素修改为同时允许 IPv4（54.240.143.0/24）和 IPv6（2001:DB8:1234:5678::/64）地址范围，如以下示例所示。您可以使用示例中所示的相同类型的 Condition 块来更新 IAM 用户和存储桶策略。

```
        "Condition": {
```

```
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
        }
```

在使用 IPv6 之前，您必须对使用 IP 地址筛选来允许 IPv6 地址范围的所有相关 IAM 用户和存储桶策略进行更新。建议您使用组织的 IPv6 地址范围以及现有的 IPv4 地址范围来更新您的 IAM 策略。有关同时允许通过 IPv6 和 IPv4 进行访问的存储桶策略示例，请参阅限制特定 IP 地址的访问权限。

您可以在 https://console.aws.amazon.com/iam/ 上使用 IAM 控制台查看 IAM 用户策略。有关 IAM 的更多信息，请参阅 IAM 用户指南。有关编辑 S3 on Outposts 桶策略的信息，请参阅为 Amazon S3 on Outposts 存储桶添加或编辑存储桶策略。

测试 IP 地址兼容性

如果您使用的是 Linux 或 Unix 实例或 macOS X 平台，则可以测试您通过 IPv6 对双堆栈端点的访问。例如，要测试通过 IPv6 与 Amazon S3 on Outposts 端点的连接，请使用 `dig` 命令：

```
dig s3-outposts.us-west-2.api.aws AAAA +short
```

如果正确设置了通过 IPv6 网络的双堆栈端点，则 `dig` 命令将返回连接的 IPv6 地址。例如：

```
dig s3-outposts.us-west-2.api.aws AAAA +short

2600:1f14:2588:4800:b3a9:1460:159f:ebce

2600:1f14:2588:4802:6df6:c1fd:ef8a:fc76

2600:1f14:2588:4801:d802:8ccf:4e04:817
```

将 IPv6 与 AWS PrivateLink 结合使用

S3 on Outposts 支持对 AWS PrivateLink 服务和端点使用 IPv6 协议。由于 AWS PrivateLink 支持 IPv6 协议，您可以通过 IPv6 网络，从本地或通过其它私有连接来连接到 VPC 内的服务端点。AWS PrivateLink for S3 on Outposts 对 IPv6 的支持还允许您将 AWS PrivateLink 与双堆栈端点集成。有关如何为 AWS PrivateLink 启用 IPv6 的步骤，请参阅 Expedite your IPv6 adoption with AWS PrivateLink services and endpoints。

> **Note**
>
> 要将支持的 IP 地址类型从 IPv4 更新为 IPv6，请参阅《AWS PrivateLink User Guide》中的 Modify the supported IP address type。

将 IPv6 与 AWS PrivateLink 结合使用

如果您将 AWS PrivateLink 与 IPv6 结合使用，则必须创建 IPv6 或双堆栈 VPC 接口端点。有关如何使用 AWS 管理控制台创建 VPC 端点的一般步骤，请参阅《AWS PrivateLink User Guide》中的 Access an AWS service using an interface VPC endpoint。

AWS 管理控制台

使用以下过程创建连接到 S3 on Outposts 的接口 VPC 端点。

1. 登录到 AWS 管理控制台并打开 VPC 控制台，网址为：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择端点。
3. 选择 创建端点。
4. 对于服务类别，选择 AWS 服务。
5. 对于服务名称，选择 S3 on Outposts 服务（com.amazonaws.us-east-1.s3-outposts）。
6. 对于 VPC，选择您要从中访问 S3 on Outposts 的 VPC。
7. 对于子网，对于每个可用区选择一个您将从中访问 S3 on Outposts 的子网。您无法从同一可用区中选择多个子网。对于您选择的每个子网，将创建一个新的端点网络接口。默认情况下，将子网 IP 地址范围中的 IP 地址分配给端点网络接口。要为端点网络接口指定 IP 地址，请选择指定 IP 地址，然后输入子网地址范围中的 IPv6 地址。
8. 对于 IP 地址类型，选择双堆栈。同时将 IPv4 和 IPv6 地址分配给端点网络接口。仅当所有选定子网都具有 IPv4 和 IPv6 地址范围时，才支持此选项。
9. 对于安全组，选择要与 VPC 端点的端点网络接口关联的安全组。默认情况下，默认安全组与 VPC 相关联。
10. 对于策略，请选择完整访问权限以允许所有主体通过 VPC 端点对所有资源执行所有操作。否则，选择自定义来附加 VPC 端点策略，该策略控制主体通过 VPC 端点对资源执行操作的权限。该选项仅在服务支持 VPC 端点策略时可用。有关更多信息，请参阅 Endpoint policies。
11. （可选）若要添加标签，请选择添加新标签，然后输入该标签的键和值。
12. 选择创建端点。

Example– S3 on Outposts 桶策略

要允许 S3 on Outposts 与 VPC 端点进行交互，您可以更新您的 S3 on Outposts 策略，如下所示：

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3-outposts:*",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

AWS CLI

> **Note**
>
> 要在您的 VPC 端点上启用 IPv6 网络，必须为 S3 on Outposts 的 `SupportedIpAddressType` 筛选条件设置 `IPv6`。

以下示例使用 `create-vpc-endpoint` 命令创建新的双堆栈接口端点。

```
aws ec2 create-vpc-endpoint \
--vpc-id vpc-12345678 \
--vpc-endpoint-type Interface \
--service-name com.amazonaws.us-east-1.s3-outposts \
--subnet-id subnet-12345678 \
--security-group-id sg-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

根据 AWS PrivateLink 服务配置，新创建的端点连接可能需要先由 VPC 端点服务提供商接受，然后才能使用。有关更多信息，请参阅《AWS PrivateLink User Guide》中的 Accept and reject endpoint connection requests。

以下示例使用 `modify-vpc-endpoint` 命令将仅限 IPv4 的 VPC 端点更新为双堆栈端点。双堆栈端点同时允许访问 IPv4 和 IPv6 网络。

```
aws ec2 modify-vpc-endpoint \
```

```
--vpc-endpoint-id vpce-12345678 \
--add-subnet-ids subnet-12345678 \
--remove-subnet-ids subnet-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

有关如何为 AWS PrivateLink 启用 IPv6 网络的更多信息，请参阅 Expedite your IPv6 adoption with AWS PrivateLink services and endpoints。

使用 S3 on Outposts 双堆栈端点

S3 on Outposts 双堆栈端点支持通过 IPv6 和 IPv4 向 S3 on Outposts 桶发出的请求。本节介绍如何使用 S3 on Outposts 双堆栈端点。

主题

- S3 on Outposts 双堆栈端点
- 从 AWS CLI 使用双堆栈端点
- 从 AWS SDK 使用 S3 on Outposts 双堆栈端点

S3 on Outposts 双堆栈端点

当您向双堆栈端点发出请求时，S3 on Outposts 桶 URL 解析为 IPv6 或 IPv4 地址。有关如何通过 IPv6 访问 S3 on Outposts 桶的更多信息，请参阅通过 IPv6 向 S3 on Outposts 发出请求。

要通过双堆栈端点访问 S3 on Outposts 桶，请使用路径样式端点名称。S3 on Outposts 只支持区域双堆栈端点名称，这意味着，您必须在名称中指定区域。

对于双堆栈路径样式的 FIP 端点，请使用以下命名约定：

```
s3-outposts-fips.region.api.aws
```

对于双堆栈非 FIPS 端点，请使用以下命名约定：

```
s3-outposts.region.api.aws
```

> **Note**
>
> S3 on Outposts 不支持虚拟托管样式的端点名称。

从 AWS CLI 使用双堆栈端点

本节提供用于向双堆栈端点发出请求的 AWS CLI 命令示例。有关设置 AWS CLI 的说明，请参阅通过 AWS CLI 和适用于 Java 的 SDK 开始使用。

在 AWS Config 文件内的配置文件中将配置值 `use_dualstack_endpoint` 设置为 `true`，从而将 `s3` 和 `s3api` AWS CLI 命令发出的所有 Amazon S3 请求都定向到指定区域的双堆栈端点。您可以在配置文件或命令中使用 `--region` 选项指定区域。

通过 AWS CLI 使用双堆栈端点时，仅支持 `path` 寻址样式。在配置文件中设置的寻址样式确定桶名称是在主机名中还是在 URL 中。有关更多信息，请参阅《AWS CLI 用户指南》中的 s3outposts。

要通过 AWS CLI 使用双栈端点，请将 `--endpoint-url` 参数与 `http://s3.dualstack.region.amazonaws.com` 或 `https://s3-outposts-fips.region.api.aws` 端点一起用于任何 `s3control` 或 `s3outposts` 命令。

例如：

```
$  aws s3control list-regional-buckets --endpoint-url https://s3-
outposts.region.api.aws
```

从 AWS SDK 使用 S3 on Outposts 双堆栈端点

本节提供一些示例，介绍如何使用 AWS SDK 来访问双堆栈端点。

AWS SDK for Java 2.x双堆栈端点示例

以下示例显示了当使用 AWS SDK for Java 2.x 创建 S3 on Outposts 客户端时，如何使用 `S3ControlClient` 和 `S3OutpostsClient` 类来启用双堆栈端点。有关为 Amazon S3 on Outposts 创建和测试有效 Java 示例的说明，请参阅通过 AWS CLI 和适用于 Java 的 SDK 开始使用。

Example– 创建启用双堆栈端点的 S3ControlClient 类

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsRequest;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsResponse;
import software.amazon.awssdk.services.s3control.model.S3ControlException;
```

```
public class DualStackEndpointsExample1 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");
        String accountId = "111122223333";
        String navyId = "9876543210";

        try {
            // Create an S3ControlClient with dual-stack endpoints enabled.
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                                                             .region(clientRegion)
                                                             .dualstackEnabled(true)
                                                             .build();
            ListRegionalBucketsRequest listRegionalBucketsRequest =
 ListRegionalBucketsRequest.builder()

       .accountId(accountId)

       .outpostId(navyId)

       .build();

            ListRegionalBucketsResponse listBuckets =
 s3ControlClient.listRegionalBuckets(listRegionalBucketsRequest);
            System.out.printf("ListRegionalBuckets Response: %s%n",
 listBuckets.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3ControlException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
    }
}
```

Example– 创建启用双堆栈端点的 S3OutpostsClient

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3outposts.S3OutpostsClient;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsRequest;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsResponse;
import software.amazon.awssdk.services.s3outposts.model.S3OutpostsException;


public class DualStackEndpointsExample2 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");

        try {
            // Create an S3OutpostsClient with dual-stack endpoints enabled.
            S3OutpostsClient s3OutpostsClient = S3OutpostsClient.builder()
                                                                .region(clientRegion)
                                                                .dualstackEnabled(true)
                                                                .build();
            ListEndpointsRequest listEndpointsRequest =
 ListEndpointsRequest.builder().build();

            ListEndpointsResponse listEndpoints =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
            System.out.printf("ListEndpoints Response: %s%n",
 listEndpoints.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3OutpostsException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
```

```
    }
}
```

如果要在 Windows 上使用 AWS SDK for Java 2.x，可能必须设置以下 Java 虚拟机（JVM）属性：

```
java.net.preferIPv6Addresses=true
```