亚马逊云科技

开发人员指南

Amazon Serverless Application Repository

Amazon Serverless Application Repository: 开发人员指南

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

Table of Contents

那是什么 Amazon Serverless Application Repository？

Amazon Serverless Application Repository 这使开发人员和企业可以轻松地在 Amazon 云中快速查找、部署和发布无服务器应用程序。有关无服务器应用程序的更多信息，请参阅网站上的无服务器计算和应用程序。 Amazon

您可以轻松地发布应用程序，将其与整个社区公开共享，或在团队中或在您的组织中私下共享。要发布无服务器应用程序（或应用程序），您可以使用 Amazon SAM 命令行界面 (Amazon SAM CLI) 或 Amazon SDKs 上传您的代码。 Amazon Web Services 管理控制台除了您的代码外，您还可以上传一个简单的清单文件，也称为 Amazon Serverless Application Model (Amazon SAM) 模板。有关的更多信息 Amazon SAM，请参阅《Amazon Serverless Application Model 开发人员指南》。

Amazon Serverless Application Repository 已与 Amazon Lambda 控制台深度集成。此集成意味着所有级别的开发人员都可以开始使用无服务器计算，而无需学习任何新的内容。可以使用类别关键字来浏览应用程序，例如 Web 和移动后端、数据处理应用程序或聊天自动程序。您还可以按名称、发布者或事件源搜索应用程序。要使用应用程序，您只需选择它，配置所有必需的字段，然后单击几次即可部署它。

在本指南中，您可以了解使用 Amazon Serverless Application Repository的两种方法：

- 发布应用程序— 配置和上传应用程序，使其可供其他开发人员使用，并发布应用程序的新版本。
- 部署应用程序— 浏览应用程序并查看有关它们的信息，包括源代码和自述文件。还可以安装、配置和部署您选择的应用程序。

后续步骤

- 有关向发布示例应用程序的教程 Amazon Serverless Application Repository，请参阅快速入门：发布应用程序。
- 有关从部署应用程序的说明 Amazon Serverless Application Repository，请参阅如何部署应用程序。

快速入门：发布应用程序

本指南将引导您完成下载、构建、测试示例无服务器应用程序并将其发布到 Amazon Serverless Application Repository 使用 Amazon SAM CLI 的步骤。您可以使用此示例应用程序作为开发和发布自己的无服务器应用程序的起点。

概述

以下步骤概述如何下载、构建和发布示例无服务器应用程序：

1. 初始化。使用 `sam init` 从模板下载示例应用程序。
2. 本地测试。使用在本地测试应用程序`sam local invoke` and/or `sam local start-api`。请注意，使用这些命令，即使您的 Lambda 函数是在本地调用的，它也会从云中的 Amazon 资源中读取和写入数据。 Amazon
3. 程序包。如果您对 Lambda 函数感到满意，请使用将 Lambda 函数、 Amazon SAM 模板和任何依赖项捆绑到部署包中。 Amazon CloudFormation `sam package`在此步骤中，您还将包含有关将上传到 Amazon Serverless Application Repository的应用程序的信息。
4. 发布。使用 `sam publish` 将应用程序发布到 Amazon Serverless Application Repository 。完成此步骤后，您可以使用查看您的应用程序 Amazon Serverless Application Repository 并将其部署到 Amazon 云端 Amazon Serverless Application Repository。

下一节中的示例Hello World 应用程序将指导您完成构建和发布无服务器应用程序的这些步骤。

Hello World 应用程序

在本练习中，您将下载并测试代表简单 API 后端的 Hello World 无服务器应用程序。它有一个支持 GET 操作的 Amazon API Gateway 终端节点和 Lambda 函数。当向终端节点发送 GET 请求时，API Gateway 会调用 Lambda 函数。然后， Amazon Lambda 执行该函数，该函数仅返回一条`hello world`消息。

该应用程序具有以下组件：

- 一个为 Hello World 应用程序定义两个 Amazon 资源的 Amazon SAM 模板：带有 GET 操作的 API Gateway 服务和一个 Lambda 函数。该模板还定义了 API Gateway GET 操作和 Lambda 函数之间的映射。

- 用 Python 编写的应用程序代码。

开始前的准备工作

请确保您具有本练习所需的设置：

- 您必须拥有一个拥有管理员权限的 IAM 用户的 Amazon 账户。请参阅设置 Amazon 账户。
- 您必须安装 Amazon SAM CLI（命令行界面）。请参见安装 C Amazon SAM LI。
- 必须安装版本 1.16.77 或更高版本。 Amazon CLI 请参阅安装 Amazon Command Line Interface。

步骤 1：初始化应用程序

在本节中，您将下载示例应用程序，其中包括 Amazon SAM 模板和应用程序代码。

初始化应用程序

1. 在 Amazon SAM CLI 命令提示符下运行以下命令。

```
sam init --runtime python3.6
```

2. 查看命令创建的目录的内容 (sam-app/)：

   - `template.yaml`— 定义 Hello World 应用程序所需的两个 Amazon 资源：一个 Lambda 函数和一个支持 GET 操作的 API Gateway 终端节点。模板还定义了两个资源之间的映射。
   - 与 Hello World 应用程序代码相关的内容：
     - `hello_world/`directory-包含应用程序代码，该代码`hello world`将在您运行时返回。

> **Note**
>
> 在本练习中，应用程序代码是用 Python 编写的，你可以在`init`命令中指定运行时间。Amazon Lambda 支持用于创建应用程序代码的其他语言。如果您指定了另一个受支持的运行时，`init` 命令将以指定的语言提供 Hello World 代码，以及一个您可以遵循该语言的 `README.md` 文件。有关支持的运行时的信息，请参阅 Lambda 执行环境和可用库。

步骤 2：在本地测试应用程序

现在，您已经在本地计算机上安装了该 Amazon SAM 应用程序，请按照以下步骤在本地对其进行测试。

在本地测试应用程序

1. 在本地启动 API 网关终端节点。您必须从包含该 `template.yaml` 文件的目录运行以下命令。

   ```
   sam-app> sam local start-api --region us-east-1
   ```

   该命令返回 API Gateway 终端节点，您可以向其发送请求以进行本地测试。
2. 测试应用程序。复制 API Gateway 端点 URL，将其粘贴到浏览器中，然后选择 Enter。API Gateway 终端节点网址示例如`http://127.0.0.1:3000/hello`下。

   API Gateway 在本地调用终端节点映射到的 Lambda 函数。Lambda 函数在本地 Docker 容器中执行并返回。`hello world`API Gateway 会向浏览器返回包含该文本的响应。

练习：更改消息字符串

成功测试示例应用程序后，您可以尝试进行简单的修改：更改返回的消息字符串。

1. 编辑 `/hello_world/app.py` 文件以将消息字符串从 `'hello world'` 更改为 `'Hello World!'`。
2. 在浏览器中重新加载测试 URL 并观察新字符串。

您会注意到您的新代码是动态加载的，而无需重新启动 `sam local` 进程。

步骤 3：打包应用程序

在本地测试应用程序后，您可以使用 Amazon SAM CLI 创建部署包和打包 Amazon SAM 模板。

> **Note**
>
> 在以下步骤中，您将为包含应用程序代码的 `hello_world/` 目录的内容创建一个 .zip 文件。此 .zip 文件是无服务器应用程序的部署包。有关更多信息，请参阅《Amazon Lambda 开发者指南》中的创建部署包 (Python)。

要创建 Lambda 部署包

1. 在 Amazon SAM 模板文件中添加一个Metadata部分，提供所需的应用程序信息。有关 Amazon SAM 模板Metadata部分的更多信息，请参阅《Amazon Serverless Application Model 开发者指南》中的 “Amazon SAM 模板元数据节属性”。

   以下是一个示例 Metadata 部分：

   ```
   Metadata:
     AWS::ServerlessRepo::Application:
       Name: my-app
       Description: hello world
       Author: user1
       SpdxLicenseId: Apache-2.0
       LicenseUrl: LICENSE.txt
       ReadmeUrl: README.md
       Labels: ['tests']
       HomePageUrl: https://github.com/user1/my-app-project
       SemanticVersion: 0.0.1
       SourceCodeUrl: https://github.com/user1/my-app-project
   ```

   LicenseUrl和ReadmeUrl属性可以是对本地文件的引用（如上例所示），也可以是指向已经托管这些项目的 Amazon S3 存储桶的链接。

2. 在要保存打包代码的位置创建 S3 存储桶。如果要使用现有 S3 存储桶，请跳过此步骤。

   ```
   sam-app> aws s3 mb s3://bucketname
   ```

3. 通过运行以下 CL package Amazon SAM I 命令创建 Lambda 函数部署包。

   ```
   sam-app> sam package \
       --template-file template.yaml \
       --output-template-file packaged.yaml \
       --s3-bucket bucketname
   ```

   此命令执行以下操作：

   - 压缩aws-sam/hello_world/目录的内容并将其上传到 Amazon S3。
   - 将部署包、自述文件和许可证文件上传到选项指定的 Amazon S3 存储桶。--s3-bucket
   - 输出一个名为 packaged.yaml 的新模板文件，您在下一步使用该文件将应用程序发布到 Amazon Serverless Application Repository。该packaged.yaml模板文件与原始模板文件

(template.yaml) 类似，但有一个关键区别——CodeUri、LicenseUrl、和ReadmeUrl属性指向包含相应项目的 Amazon S3 存储桶和对象。来自示例 packaged.yaml 模板文件的以下代码段显示了 CodeUri 属性：

```
HelloWorldFunction:
    Type: AWS::Serverless::Function # For more information about function
 resources, see https://github.com/awslabs/serverless-application-model/blob/
master/versions/2016-10-31.md#awsserverlessfunction
    Properties:
      CodeUri: s3://bucketname/fbd77a3647a4f47a352fcObjectGUID

...
```

步骤 4：发布应用程序

既然您已创建部署包，就可以使用它将应用程序发布到 Amazon Serverless Application Repository。

将无服务器应用程序发布到 Amazon Serverless Application Repository

- 执行以下命令以在 Amazon Serverless Application Repository 中发布新应用程序，并将第一个版本创建为 0.0.1。

```
sam-app> sam publish \
    --template packaged.yaml \
    --region us-east-1
```

> **Note**
>
> 默认情况下，应用程序将创建为私有应用程序。您必须先共享该应用程序，然后才能允许其他 Amazon 账户查看和部署您的应用程序。有关共享应用程序的更多详细信息，请参阅下面的 Next Steps (后续步骤)。

后续步骤

现在，您已经发布了示例应用程序，以下是您可能需要对其执行的几项操作。

- 查看您的应用程序 Amazon Serverless Application Repository — `sam publish` 命令的输出将包括一个 Amazon Serverless Application Repository 直接指向您的应用程序详情页面的链接。您也可以前往 Amazon Serverless Application Repository 登录页面并搜索您的应用程序。
- 共享您的应用程序-由于您的应用程序在默认情况下设置为私有，因此其他 Amazon 账户无法看到该应用程序。为了与他人共享您的应用程序，您必须将其公开，或者向特定的 Amazon 账户列表授予权限。有关使用共享应用程序的信息， Amazon CLI 请参阅Amazon Serverless Application Repository 应用程序策略示例。有关使用控制台共享应用程序的信息，请参阅共享应用程序。

更多信息

有关 Amazon SAM 模板`Metadata`部分`sam package`和 Amazon SAM CLI `sam publish` 命令的更多信息，请参阅《Amazon Serverless Application Model 开发人员指南》中的 “使用 Amazon SAM CLI 发布应用程序”。

发布应用程序

当您将无服务器应用程序发布到时 Amazon Serverless Application Repository，您可以将其提供给其他人查找和部署。

您首先使用 Amazon Serverless Application Model (Amazon SAM) 模板定义您的应用程序。定义应用程序时，必须考虑是否要求应用程序的使用者确认应用程序的功能。有关使用 Amazon SAM 和确认权能的更多信息，请参阅Amazon SAM 与 Amazon Serverless Application Repository。

您可以使用 Amazon Web Services 管理控制台、 Amazon SAM 命令行界面 (Amazon SAM CLI) 或 Amazon SDK 发布无服务器应用程序。要了解有关向发布应用程序的过程的更多信息 Amazon Serverless Application Repository，请参阅如何发布应用程序。

当你发布应用程序时，它最初设置为私有，这意味着只有创建它的 Amazon 账户才能使用它。要与他人共享您的应用程序，您必须将其设置为私下共享（仅与一组特定的 Amazon 账户共享）或公开共享（与所有人共享）。

当您将应用程序发布到 Amazon Serverless Application Repository 并将其设置为公共时，该服务将向所有区域的消费者提供该应用程序。当消费者将公共应用程序部署到除首次发布该应用程序的区域之外的区域时，该用户会将该应用程序的部署项目 Amazon Serverless Application Repository 复制到目标区域的 Amazon S3 存储桶中。它会更新 Amazon SAM 模板中使用这些项目的所有资源，以改为引用目标区域的 Amazon S3 存储桶中的文件。部署工件可以包括 Lambda 函数代码、API 定义文件等。

> **Note**
>
> 私有和私有共享的应用程序仅在创建它们 Amazon 所在的地区可用。所有 Amazon 地区都提供 @@ 公开共享的应用程序。要了解有关共享应用程序的更多信息，请参阅Amazon Serverless Application Repository 应用程序策略示例。

主题

Amazon SAM 与 Amazon Serverless Application Repository

Amazon Serverless Application Model (Amazon SAM) 是一个开源框架，可用于在其上 Amazon构建无服务器应用程序。有关使用构建无服务器应用程序 Amazon SAM 的更多信息，请参阅《Amazon Serverless Application Model 开发人员指南》。

在构建将发布到的应用程序时 Amazon Serverless Application Repository，必须考虑一组可用的受支持 Amazon 资源和策略模板。以下各节将更详细地介绍这些主题。

中支持的 Amazon 资源 Amazon Serverless Application Repository

Amazon Serverless Application Repository 支持由许多 Amazon SAM Amazon CloudFormation 资源组成的无服务器应用程序。要查看支持的 Amazon 资源的完整列表 Amazon Serverless Application Repository，请参阅支持的 Amazon 资源清单。

如果您想请求支持以获取其他 Amazon 资源，请联系 Suppor Amazon t。

Important

Amazon Serverless Application Repository 阻止发布包含以下过于宽泛的 IAM 权限模式的应用程序，这些模式不遵循最低权限原则：

- 将`AWSLambda_FullAccess`托管策略附加到 Lambda 函数
- 在内联 IAM 策略中`iam:*`对所有资源 (*) 授予`iam:AttachRolePolicy`、`iam:PutRolePolicy`或

要发布您的应用程序，请仅`AWSLambda_FullAccess`替换为应用程序所需的特定 Lambda 权限`iam:AttachRolePolicy`、范围`iam:PutRolePolicy`、和`iam:PassRole`权限， ARNs 而不是所有资源。有关指导，请参阅 IAM 安全最佳实践。

Important

如果您的应用程序模板包含以下任一自定义 IAM 角色或资源策略，则默认情况下，您的应用程序不会显示在搜索结果中。另外，客户需要确认应用程序的自定义 IAM 角色或资源策略，然后才能部署应用程序。有关更多信息，请参阅确认应用程序功能。
这适用于的资源列表是：

- IAM 角色：AWS::IAM::GroupAWS::IAM::InstanceProfile、AWS::IAM::Policy、和AWS::IAM::Role。
- 资源策略：AWS::Lambda::LayerVersion权限、AWS::Events::EventBus策略AWS::Lambda::Permission、AWS:: iam: Policy、、AWS::ApplicationAutoScaling::ScalingPolicyAWS::S3::BucketPolicy、AWS::SQS::QueueP AWS:: SN S:。TopicPolicy

如果您的应用程序包含AWS::Serverless::Application资源，则客户需要先确认该应用程序包含嵌套应用程序，然后才能部署该应用程序。有关嵌套应用程序的更多信息，请参阅《Amazon Serverless Application Model 开发人员指南》中的嵌套应用程序。有关确认功能的更多信息，请参阅确认应用程序功能。

策略模板

Amazon SAM 为您提供了策略模板列表，用于将 Lambda 函数的权限范围限定为应用程序所使用的资源。使用策略模板不需要额外的客户确认，即可搜索、浏览或部署应用程序。

有关标准 Amazon SAM 策略模板的列表，请参阅《Amazon Serverless Application Model 开发人员指南》中的Amazon SAM 策略模板。

支持的 Amazon 资源清单

这是支持的 Amazon 资源的完整列表 Amazon Serverless Application Repository。

- `AWS::AccessAnalyzer::Analyzer`
- `AWS::AmazonMQ::Broker`
- `AWS::AmazonMQ::Configuration`
- `AWS::AmazonMQ::ConfigurationAssociation`
- `AWS::ApiGateway::Account`
- `AWS::ApiGateway::ApiKey`
- `AWS::ApiGateway::Authorizer`
- `AWS::ApiGateway::BasePathMapping`
- `AWS::ApiGateway::ClientCertificate`

- AWS::ApiGateway::Deployment
- AWS::ApiGateway::DocumentationPart
- AWS::ApiGateway::DocumentationVersion
- AWS::ApiGateway::DomainName
- AWS::ApiGateway::GatewayResponse
- AWS::ApiGateway::Method
- AWS::ApiGateway::Model
- AWS::ApiGateway::RequestValidator
- AWS::ApiGateway::Resource
- AWS::ApiGateway::RestApi
- AWS::ApiGateway::Stage
- AWS::ApiGateway::UsagePlan
- AWS::ApiGateway::UsagePlanKey
- AWS::ApiGateway::VpcLink
- AWS::ApiGatewayV2::Api
- AWS::ApiGatewayV2::ApiMapping
- AWS::ApiGatewayV2::Authorizer
- AWS::ApiGatewayV2::DomainName
- AWS::ApiGatewayV2::Deployment
- AWS::ApiGatewayV2::Integration
- AWS::ApiGatewayV2::IntegrationResponse
- AWS::ApiGatewayV2::Model
- AWS::ApiGatewayV2::Route
- AWS::ApiGatewayV2::RouteResponse
- AWS::ApiGatewayV2::Stage
- AWS::AppSync::ApiKey
- AWS::AppSync::DataSource
- AWS::AppSync::GraphQLApi

- AWS::AppSync::GraphQLSchema
- AWS::AppSync::Resolver
- AWS::ApplicationAutoScaling::AutoScalingGroup
- AWS::ApplicationAutoScaling::LaunchConfiguration
- AWS::ApplicationAutoScaling::ScalableTarget
- AWS::ApplicationAutoScaling::ScalingPolicy
- AWS::Athena::NamedQuery
- AWS::Athena::WorkGroup
- AWS::CertificateManager::Certificate
- AWS::Chatbot::SlackChannelConfiguration
- AWS::CloudFormation::CustomResource
- AWS::CloudFormation::Interface
- AWS::CloudFormation::Macro
- AWS::CloudFormation::WaitConditionHandle
- AWS::CloudFront::CachePolicy
- AWS::CloudFront::CloudFrontOriginAccessIdentity
- AWS::CloudFront::Distribution
- AWS::CloudFront::Function
- AWS::CloudFront::OriginRequestPolicy
- AWS::CloudFront::ResponseHeadersPolicy
- AWS::CloudFront::StreamingDistribution
- AWS::CloudTrail::Trail
- AWS::CloudWatch::Alarm
- AWS::CloudWatch::AnomalyDetector
- AWS::CloudWatch::Dashboard
- AWS::CloudWatch::InsightRule
- AWS::CodeBuild::Project
- AWS::CodeCommit::Repository

- AWS::CodePipeline::CustomActionType
- AWS::CodePipeline::Pipeline
- AWS::CodePipeline::Webhook
- AWS::CodeStar::GitHubRepository
- AWS::CodeStarNotifications::NotificationRule
- AWS::Cognito::IdentityPool
- AWS::Cognito::IdentityPoolRoleAttachment
- AWS::Cognito::UserPool
- AWS::Cognito::UserPoolClient
- AWS::Cognito::UserPoolDomain
- AWS::Cognito::UserPoolGroup
- AWS::Cognito::UserPoolResourceServer
- AWS::Cognito::UserPoolUser
- AWS::Cognito::UserPoolUserToGroupAttachment
- AWS::Config::AggregationAuthorization
- AWS::Config::ConfigRule
- AWS::Config::ConfigurationAggregator
- AWS::Config::ConfigurationRecorder
- AWS::Config::DeliveryChannel
- AWS::Config::RemediationConfiguration
- AWS::DataPipeline::Pipeline
- AWS::DynamoDB::Table
- AWS::EC2::EIP
- AWS::EC2::InternetGateway
- AWS::EC2::NatGateway
- AWS::EC2::Route
- AWS::EC2::RouteTable
- AWS::EC2::SecurityGroup
- AWS::EC2::SecurityGroupEgress

- AWS::EC2::SecurityGroupIngress
- AWS::EC2::Subnet
- AWS::EC2::SubnetRouteTableAssociation
- AWS::EC2::VPC
- AWS::EC2::VPCGatewayAttachment
- AWS::EC2::VPCPeeringConnection
- AWS::ECR::Repository
- AWS::Elasticsearch::Domain
- AWS::Events::EventBus
- AWS::Events::EventBusPolicy
- AWS::Events::Rule
- AWS::EventSchemas::Discoverer
- AWS::EventSchemas::Registry
- AWS::EventSchemas::Schema
- AWS::Glue::Classifier
- AWS::Glue::Connection
- AWS::Glue::Crawler
- AWS::Glue::Database
- AWS::Glue::DevEndpoint
- AWS::Glue::Job
- AWS::Glue::Partition
- AWS::Glue::SecurityConfiguration
- AWS::Glue::Table
- AWS::Glue::Trigger
- AWS::Glue::Workflow
- AWS::IAM::Group
- AWS::IAM::InstanceProfile
- AWS::IAM::ManagedPolicy
- AWS::IAM::OIDCProvider

- AWS::IAM::Policy
- AWS::IAM::Role
- AWS::IAM::ServiceLinkedRole
- AWS::IoT::Certificate
- AWS::IoT::Policy
- AWS::IoT::PolicyPrincipalAttachment
- AWS::IoT::Thing
- AWS::IoT::ThingPrincipalAttachment
- AWS::IoT::TopicRule
- AWS::KMS::Alias
- AWS::KMS::Key
- AWS::Kinesis::Stream
- AWS::Kinesis::StreamConsumer
- AWS::Kinesis::Streams
- AWS::KinesisAnalytics::Application
- AWS::KinesisAnalytics::ApplicationOutput
- AWS::KinesisFirehose::DeliveryStream
- AWS::Lambda::Alias
- AWS::Lambda::EventInvokeConfig
- AWS::Lambda::EventSourceMapping
- AWS::Lambda::Function
- AWS::Lambda::LayerVersion
- AWS::Lambda::LayerVersionPermission
- AWS::Lambda::Permission
- AWS::Lambda::Version
- AWS::Location::GeofenceCollection
- AWS::Location::Map
- AWS::Location::PlaceIndex
- AWS::Location::RouteCalculator

- AWS::Location::Tracker
- AWS::Location::TrackerConsumer
- AWS::Logs::Destination
- AWS::Logs::LogGroup
- AWS::Logs::LogStream
- AWS::Logs::MetricFilter
- AWS::Logs::SubscriptionFilter
- AWS::Route53::HealthCheck
- AWS::Route53::HostedZone
- AWS::Route53::RecordSet
- AWS::Route53::RecordSetGroup
- AWS::S3::Bucket
- AWS::S3::BucketPolicy
- AWS::SNS::Subscription
- AWS::SNS::Topic
- AWS::SNS::TopicPolicy
- AWS::SQS::Queue
- AWS::SQS::QueuePolicy
- AWS::SSM::Association
- AWS::SSM::Document
- AWS::SSM::MaintenanceWindowTask
- AWS::SSM::Parameter
- AWS::SSM::PatchBaseline
- AWS::SSM::ResourceDataSync
- AWS::SecretsManager::ResourcePolicy
- AWS::SecretsManager::RotationSchedule
- AWS::SecretsManager::Secret
- AWS::SecretsManager::SecretTargetAttachment
- AWS::Serverless::Api

- AWS::Serverless::Application
- AWS::Serverless::Function
- AWS::Serverless::HttpApi
- AWS::Serverless::LayerVersion
- AWS::Serverless::SimpleTable
- AWS::Serverless::StateMachine
- AWS::ServiceDiscovery::HttpNamespace
- AWS::ServiceCatalog::CloudFormationProvisionedProduct
- AWS::ServiceDiscovery::Instance
- AWS::ServiceDiscovery::PrivateDnsNamespace
- AWS::ServiceDiscovery::PublicDnsNamespace
- AWS::ServiceDiscovery::Service
- AWS::SES::ReceiptRule
- AWS::SES::ReceiptRuleSet
- AWS::StepFunctions::Activity
- AWS::StepFunctions::StateMachine
- AWS::Wisdom::Assistant
- AWS::Wisdom::AssistantAssociation
- AWS::Wisdom::KnowledgeBase

如何发布应用程序

本节为您提供使用 Amazon SAM CLI 或将无服务器应用程序发布到的过程。 Amazon Serverless Application Repository Amazon Web Services 管理控制台它还向您展示如何共享您的应用程序以允许其他人部署它，以及从 Amazon Serverless Application Repository中删除您的应用程序。

> **Important**
>
> 您在发布应用程序时输入的信息未加密。此信息包括作者姓名等数据。如果您有不希望存储或公开的个人身份信息，我们建议您不要在发布应用程序时输入此类信息。

发布应用程序 (Amazon CLI)

向发布应用程序的最简单方法 Amazon Serverless Application Repository 是使用一组 Amazon SAM CLI 命令。有关更多信息，请参阅 Amazon Serverless Application Model (Amazon SAM) 开发者指南中的使用 Amazon SAM CLI 发布应用程序。

发布新应用程序（控制台）

本节介绍如何使用 Amazon Web Services 管理控制台 向发布新应用程序 Amazon Serverless Application Repository。有关发布现有应用程序的新版本的说明，请参阅发布现有应用程序的新版本。

先决条件

在将应用程序发布到之前 Amazon Serverless Application Repository，您需要满足以下条件：

- 有效的 Amazon 账户。
- 定义所用 Amazon 资源的有效 Amazon Serverless Application Model (Amazon SAM) 模板。有关 Amazon SAM 模板的更多信息，请参阅Amazon SAM 模板基础知识。
- 您使用 Amazon CloudFormation `package`命令为应用程序创建的软件包 Amazon CLI。此命令将您的 Amazon SAM 模板引用的本地工件（本地路径）打包。有关更多详细信息，请参阅 Amazon CloudFormation 文档中的软件包。
- 指向应用程序源代码的 URL（如果您需要公开发布应用程序）。
- 一个 readme.txt 文件。此文件应描述客户如何使用您的应用程序，以及如何在将其部署到自己的 Amazon 账户中之前对其进行配置。
- 来自 SPDX 网站的 license.txt 文件或有效的许可证标识符。请注意，只有当您想要公开共享您的应用程序时，才需要许可证。如果您要将应用程序保持为私有或仅私下共享，则无需指定许可证。
- 有效的 Amazon S3 存储桶策略，用于向服务授予在您打包应用程序时上传到 Amazon S3 的项目的读取权限。要设置此策略，请按照下列步骤操作：

  1. 打开 Amazon S3 控制台，网址为 https://console.aws.amazon.com/s3/。
  2. 选择用于打包您的应用程序的 Amazon S3 存储桶。
  3. 选择权限选项卡。
  4. 选择存储桶策略按钮。
  5. 将以下策略语句粘贴到 Bucket policy editor (存储桶策略编辑器) 中。请务必在元素中替换您的存储桶名称，在`Resource`元素中替换您的 Amazon 账户 ID。`Condition``Condition`元素中

的表达式确保 Amazon Serverless Application Repository 只有从指定 Amazon 账户访问应用程序的权限。有关策略声明的更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素参考。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service":  "serverlessrepo.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::bucketname/*",
            "Condition" : {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

6. 选择保存按钮。

过程

使用以下步骤在 Amazon Serverless Application Repository 中创建新应用程序。

要在中创建新应用程序 Amazon Serverless Application Repository

1. 打开 Amazon Serverless Application Repository 控制台并选择 Publish applications (发布应用程序)。
2. 在 Publish an application (发布应用程序) 页面上，输入以下应用程序信息，然后选择 Publish application (发布应用程序)：

属性	必需	说明
应用程序名称	TRUE	应用程序的名称。

属性	必需	说明
		最小长度 = 1。最大长度 = 140。 模式："[a-zA-Z0-9\\-]+";
作者	TRUE	发布应用程序的作者的姓名。 最小长度 = 1。最大长度 = 127。 模式："^[a-z0-9](([a-z0-9]|-(?!-))*[a-z0-9])?$";
主页	FALSE	包含有关应用程序的更多信息的 URL，例如应用程序 GitHub 存储库的位置。
描述	TRUE	关于应用程序的描述。 最小长度 = 1。最大长度 = 256。
标签	FALSE	改善在搜索中发现应用程序的结果的标签。 最小长度 = 1。最大长度 = 127。最大标签数量：10。 模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

属性	必需	说明
Spdx 许可证（下拉列表）	FALSE	从下拉列表中选择包含 SPDX 网站上可用的许可证的有效许可证标识符。在下拉列表中选择一个项目将填充其下方的 License (许可证) 文本框。注意：在下拉列表中选择许可证将替换 License (许可证) 文本框中的内容，并放弃您所做的任何手动编辑。
许可证	FALSE	上传 .txt 许可证文件，或从上一行中描述的 Spdx license (Spdx 许可证) 下拉菜单中选择一个许可证。从 Spdx license (Spdx 许可证) 下拉列表中选择许可证会自动填充 License (许可证) 文本框。上传许可证文件或从 Spdx license (Spdx 许可证) 下拉菜单中选择一个许可证文件后，您可以手动编辑此文本框的内容。但是，如果从下拉列表中选择了另一个 Spdx license (Spdx 许可证) ，则会丢弃您所做的任何手动编辑。 这是一个可选字段，但您必须提供许可证才能公开共享应用程序。

属性	必需	说明
Readme (自述文件)	FALSE	上传自述文件的内容，该文件可以是文本或记录格式。这些内容显示在 Amazon Serverless Application Repository中应用程序的详细信息页面上。上传文件后，您可以手动编辑此文本框的内容。
Semantic version	FALSE	应用程序的语义版本。有关更多信息，请参阅语义版本控制网站。 您必须为此属性提供一个值，才能使您的应用程序变为公有的。
Source code Url (源代码 Url)	FALSE	指向应用程序源代码的公共存储库的链接。
SAM 模板	TRUE	定义所用 Amazon 资源的有效 Amazon Serverless Application Model (Amazon SAM) 模板。

共享应用程序

已发布的应用程序可能已设置以下三个类别之一的权限：

- 私有（默认）-使用同一账户创建且未与任何其他 Amazon 账户共享的应用程序。只有共享您 Amazon 账户的消费者才有权部署私有应用程序。
- 私下共享 — 发布者已明确与一组特定 Amazon 帐户或 Amazon 组织中的 Amazon 帐户共享的应用程序。消费者有权部署已与其 Amazon 账户或 Amazon 组织共享的应用程序。有关的更多信息 Amazon Organizations，请参阅《Amazon Organizations 用户指南》。
- 公开共享-发布者与所有人共享的应用程序。所有使用者都有权部署任何公开共享的应用程序。

将应用程序发布到后 Amazon Serverless Application Repository，该应用程序默认设置为私有。本节向您展示如何与特定 Amazon 账户或 Amazon 组织私下共享应用程序，或者如何与所有人公开共享应用程序。

通过控制台共享应用程序

您可以通过两种方式与他人共享您的应用程序：1) 与特定 Amazon 账户或 Amazon 组织内的 Amazon 账户共享，或者 2) 与所有人公开共享。有关的更多信息 Amazon Organizations，请参阅《Amazon Organizations 用户指南》。

选项 1：与 Amazon 组织内的特定 Amazon 账户或账户共享您的应用程序

1. 打开 Amazon Serverless Application Repository 控制台。
2. 在导航窗格上，选择 Published Applications (已发布的应用程序)，以显示您已创建的应用程序的列表。
3. 选择要共享的应用程序。
4. 选择 Sharing (共享) 选项卡。
5. 在 Application policy statements (应用程序策略语句) 部分中，选择 Create Statement (创建语句) 按钮。
6. 在 Statement Configuration (语句配置) 窗口中，根据您希望共享应用程序的方式填写各个字段。

   > **Note**
   >
   > 如果您与组织共享，则只能指定您的 Amazon 账户所属的组织。如果您尝试指定一个您不是其成员的 Amazon 组织，则会出现错误。
   > 要与您的 Amazon 组织共享您的申请，您必须确认该`UnshareApplication`操作将添加到您的政策声明中，以防将来需要撤销共享。

7. 选择保存按钮。

选项 2：与所有人公开共享您的应用程序

1. 打开 Amazon Serverless Application Repository 控制台。
2. 在导航窗格上，选择 Published Applications (已发布的应用程序)，以显示您已创建的应用程序的列表。
3. 选择要共享的应用程序。

4. 选择 Sharing (共享) 选项卡。
5. 在 Public Sharing (公开共享) 部分，选择 Edit (编辑) 按钮。
6. 在 Public sharing (公开共享) 下，选择 Enabled (已启用) 单选按钮。
7. 在文本框中键入应用程序的名称，然后选择 Save (保存) 按钮。

> **Note**
>
> 要公开共享应用程序，它必须同时设置了 `SemanticVersion` 和 `LicenseUrl` 属性。

通过共享应用程序 Amazon CLI

要使用共享应用程序， Amazon CLI 您可以使用`put-application-policy`命令授予权限，指定要与之共享的 Amazon 账户作为委托人。

有关使用 Amazon CLI 共享应用程序的更多信息，请参阅Amazon Serverless Application Repository 应用程序策略示例。

取消共享应用程序

取消与 Amazon 组织共享应用程序的选项有两种：

1. 应用程序的发布者可以使用 `put-application-policy` 命令删除权限。
2. 来自组织管理账户的用户可以对与该 Amazon 组织共享的任何应用程序执行取消共享应用程序操作，即使该应用程序是由其他账户的用户发布的。

> **Note**
>
> 当通过 “取消共享应用程序” 操作取消与 Amazon 组织共享应用程序时，该应用程序将无法再次与 Amazon 组织共享。

有关的更多信息 Amazon Organizations，请参阅《Amazon Organizations 用户指南》。

发布者删除权限

发布者通过控制台删除权限

要通过取消共享应用程序 Amazon Web Services 管理控制台，请删除与其他 Amazon 账户共享该应用程序的政策声明。为此，请按照以下步骤操作：

1. 打开 Amazon Serverless Application Repository 控制台。
2. 在左侧导航窗格中，选择 Available Applications (可用的应用程序)。
3. 选择要取消共享的应用程序。
4. 选择 Sharing (共享) 选项卡。
5. 在 Application policy statements (应用程序策略语句) 部分中，选择与要取消共享的账户共享应用程序的策略语句。
6. 选择删除。
7. 此时会显示确认消息。再次选择删除。

发布者通过删除权限 Amazon CLI

要通过取消共享应用程序 Amazon CLI，发布者可以使用`put-application-policy`命令移除或以其他方式更改权限，将该应用程序设为私有，或者与其他 Amazon 账户组共享。

有关使用 Amazon CLI 更改权限的更多信息，请参阅Amazon Serverless Application Repository 应用程序策略示例。

管理账号取消共享应用程序

管理账号通过控制台取消与 Amazon 组织共享的应用程序

要通过取消共享 Amazon 组织中的应用程序 Amazon Web Services 管理控制台，管理账户中的用户可以执行以下操作：

1. 打开 Amazon Serverless Application Repository 控制台。
2. 在左侧导航窗格中，选择 Available Applications (可用的应用程序)。
3. 在应用程序的磁贴中，选择 Unshare (取消共享)。
4. 在取消共享消息框中，通过输入组织 ID 和应用程序名称，然后选择 Save (保存)，以确认您要取消共享应用程序。

管理账户取消与 Amazon 组织共享应用程序的权限 Amazon CLI

要取消与 Amazon 组织的共享应用程序，管理账户中的用户可以运行该`aws serverlessrepo unshare-application`命令。

以下命令从 Amazon 组织取消共享应用程序，其中*`application-id`*是应用程序的 Amazon 资源名称 (ARN)*`organization-id`*，也是 Amazon 组织 ID：

```
aws serverlessrepo unshare-application --application-id application-id --organization-id organization-id
```

删除应用程序

您可以使用 Amazon Web Services 管理控制台 或 Amazon SAM CLI 从中删除应用程序。 Amazon Serverless Application Repository

删除应用程序（控制台）

要通过删除已发布的应用程序 Amazon Web Services 管理控制台，请执行以下操作。

1. 打开 Amazon Serverless Application Repository 控制台。
2. 对于 My Applications (我的应用程序)，请选择要删除的应用程序。
3. 在应用程序的详细信息页面中，选择 Delete application (删除应用程序)。
4. 选择 Delete application (删除应用程序) 来完成删除。

删除应用程序 (Amazon CLI)

要使用删除已发布的应用程序 Amazon CLI，请运行`aws serverlessrepo delete-application`命令。

以下命令删除应用程序，其中 *`application-id`* 是应用程序的 Amazon 资源名称 (ARN)：

```
aws serverlessrepo delete-application --application-id application-id
```

发布现有应用程序的新版本

本节介绍如何使用 Amazon SAM CLI 或，向 Amazon Serverless Application Repository 发布现有应用程序的新版本 Amazon Web Services 管理控制台。有关发布新应用程序的说明，请参阅如何发布应用程序。

发布现有应用程序的新版本 (Amazon CLI)

发布现有应用程序的新版本的最简单方法是使用一组 Amazon SAM CLI 命令。有关更多信息，请参阅 Amazon Serverless Application Model (Amazon SAM) 开发者指南中的使用 Amazon SAM CLI 发布应用程序。

发布现有应用程序的新版本（控制台）

要发布以前发布的应用程序的新版本，请按照下列步骤操作：

1. 打开 Amazon Serverless Application Repository 控制台。
2. 在导航窗格上，选择 My Applications (我的应用程序) 以显示您已创建的应用程序的列表。
3. 选择要为其发布新版本的应用程序。
4. 选择 새 버전 발행。
5. 在 Versions (版本) 中，输入以下应用程序信息：

属性	必需	说明
Semantic version	TRUE	应用程序的语义版本。有关更多信息，请参阅语义版本控制网站。 您必须为此属性提供一个值，才能使您的应用程序变为公有的。
Source code Url (源代码 Url)	FALSE	指向应用程序源代码的公共存储库的链接。
SAM 模板	TRUE	定义所用 Amazon 资源的有效 Amazon Serverless Application Model (Amazon SAM) 模板。

6. 选择发布版本。

经过验证的作者徽章

中@@ 经过验证的作者 Amazon Serverless Application Repository 是指那些作为合理而 Amazon 谨慎的服务提供商，对请求者提供的信息进行了真诚的审查，并确认请求者的身份与所声称的相同。

经过验证的作者的应用程序会显示经过验证的作者徽章以及指向作者公开个人资料的链接。经过验证的作者徽章将显示在搜索结果和应用程序详情页面上。

请求经过验证的作者徽章

你可以通过向 serverlessrepo-verified-author@amazon .com 发送电子邮件来申请 Amazon Serverless Application Repository 被批准为经过验证的作者。您需要提供以下信息：

- 作者姓名
- Amazon 账号
- 可公开访问的个人资料链接，例如您的 LinkedIn个人资料 GitHub 或个人资料

提交经过验证的作者徽章的申请后，您将在几天 Amazon 内收到回复。在您的请求获得批准之前，可能会要求您提供更多信息。

您的请求获得批准后，可能会在一天内为您的应用程序显示经过验证的作者徽章。

> **Note**
>
> 所有与 Amazon 账户和作者姓名都匹配的应用程序都会显示经过验证的作者徽章。由于 Amazon 帐户可以有多个作者，因此徽章不会显示在具有不同作者姓名的应用程序上。要在具有不同作者姓名的应用程序上显示作者徽章，您必须为该作者提交另一个请求。

共享 Lambda 图层

如果您已在 Lambda 层中实现了功能，则可能需要在不托管该层的全局实例的情况下共享该层。通过以这种方式共享层，其他用户可以将层的实例部署到自己的账户。这样可以防止客户端应用程序依赖于层的全局实例。 Amazon Serverless Application Repository 使您能够以这种方式轻松共享 Lambda 层。

有关 Lambda 层的更多信息，请参阅Amazon Lambda 开发者指南中的Amazon Lambda 层。

工作方式

以下是使用 Amazon Serverless Application Repository共享层的步骤。这允许在用户的 Amazon 账户中创建图层的副本。

1. 使用包含您的层作为资源的 Amazon SAM 模板来定义无服务器应用程序，即AWS::Serverless::LayerVersion或AWS::Lambda::LayerVersion资源。
2. 将您的应用程序发布到 Amazon Serverless Application Repository并共享（公开或私下）。
3. 客户部署了您的应用程序，该应用程序会在自己的 Amazon 账户中创建您的图层副本。现在，客户可以在其客户端应用程序中在其 Amazon 账户中引用该层的 Amazon 资源名称 (ARN)。

示例

以下是包含您要共享的 Lambda 层的应用程序的示例 Amazon SAM 模板：

```
Resources:
  SharedLayer:
    Type: AWS::Serverless::LayerVersion
    Properties:
      LayerName: shared-layer
      ContentUri: source/layer-code/
      CompatibleRuntimes:
        - python3.7
Outputs:
  LayerArn:
    Value: !Ref SharedLayer
```

当客户从部署您的应用程序时 Amazon Serverless Application Repository，将在他们的 Amazon 账户中创建一个层。层的 ARN 如下所示：

```
arn:aws:lambda:us-east-1:012345678901:layer:shared-layer:1
```

客户现在可以在自己的客户端应用程序中引用此 ARN，如下例所示：

```
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: python3.7
```

```
      CodeUrl: source/app-code/
      Layers:
        - arn:aws:lambda:us-east-1:012345678901:layer:shared-layer:1
```

部署应用程序

此部分帮助您了解如何查找和部署已发布到 Amazon Serverless Application Repository的无服务器应用程序。您可以通过访问公共网站来浏览公开发布的应用程序，而无需 Amazon 注册帐户。或者，您可以从 Amazon Lambda 控制台中浏览应用程序。

某些应用程序具有经过验证的作者徽章，并附有指向作者个人资料的链接。如果 Amazon 作者以合理和谨慎的服务提供商的身份对请求者提供的信息进行了真诚的审查，并确认请求者的身份与所声称的相同，则该作者被视为经过验证的作者。

在部署应用程序之前 Amazon Serverless Application Repository，请参阅以下主题以了解应用程序部署权限和应用程序功能。

主题

应用程序部署权限

要在中部署应用程序 Amazon Serverless Application Repository，您必须拥有执行此操作的权限。您有权部署的应用程序分为三类：

- 私有 — 使用同一个账户创建且未与任何其他账户共享的应用程序。您有权部署使用您的 Amazon 账户创建的应用程序。
- 私下共享 — 发布者已明确与一组特定 Amazon 账户共享的应用程序。您有权部署已与您的 Amazon 账户共享的应用程序。
- 公开共享-发布者与所有人共享的应用程序。您有权部署任何公开共享的应用程序。

您只能搜索和浏览您有权限的应用程序。其中包括使用您的 Amazon 账户创建、与您的 Amazon 账户私下共享以及公开共享的应用程序。不会为您显示所有其他应用程序。

> **Important**
>
> 包含嵌套应用程序的应用程序继承嵌套应用程序的共享限制。例如，假设一个应用程序是公开共享的，但它包含一个仅与创建父应用程序的 Amazon 账户私下共享的嵌套应用程序。在这种

情况下，如果您的 Amazon 账户无权部署嵌套应用程序，则无法部署父应用程序。有关嵌套应用程序的更多信息，请参阅《Amazon Serverless Application Model 开发人员指南》中的嵌套应用程序。

应用程序功能：IAM 角色、资源策略和嵌套应用程序

在部署应用程序之前， Amazon Serverless Application Repository 会检查应用程序模板中是否有 IAM 角色、 Amazon 资源策略以及模板指定应创建的嵌套应用程序。IAM 资源，例如具有完全访问权限的 IAM 角色，可以修改您 Amazon 账户中的任何资源。因此，建议您在继续之前检查与应用程序关联的权限，以便您不会无意中创建具有升级权限的资源。为确保已完成此操作，您必须先确认该应用程序包含功能，然后 Amazon Serverless Application Repository 才能代表您部署该应用程序。

应用程序可能包含以下四个功能中的任何功能：`CAPABILITY_IAM`、`CAPABILITY_NAMED_IAM`、`CAPABILITY_RESOURCE_POLICY` 和 `CAPABILITY_AUTO_EXPAND`。

以下资源需要您指定`CAPABILITY_IAM`或`CAPABILITY_NAMED_IAM`：AWS::IAM::Group、AWS::IAM::InstanceProfileAWS::IAM和AWS::IAM::Role。如果应用程序包含具有自定义名称的 IAM 资源，您必须指定 `CAPABILITY_NAMED_IAM`。有关如何指定功能的示例，请参阅查找并确认应用程序功能 (Amazon CLI)。

以下资源需要您指定`CAPABILITY_RESOURCE_POLICY`：AWS::Lambda::LayerVersion权限、、AWS::Events::EventBus策略AWS::Lambda::Permission、AWS:: iam:Policy、、、AWS::ApplicationAutoScaling::ScalingPolicy、AWS::S3::BucketPolicy和。AWS::SQS::QueuePo

包含一个或多个嵌套应用程序的应用程序要求您指定 `CAPABILITY_AUTO_EXPAND`。有关嵌套应用程序的更多信息，请参阅《Amazon Serverless Application Model 开发人员指南》中的嵌套应用程序。

> **Note**
>
> Amazon Serverless Application Repository 对新发布的应用程序实施限制，以便在从公共存储库部署时为您提供保护。具体而言， Amazon Serverless Application Repository 阻止发布将`AWSLambda_FullAccess`托管策略附加到 Lambda 函数的应用程序，或者对内联 IAM 策略中的所有资源授予`iam:AttachRolePolicy`或`iam:*`的应用程序。`iam:PutRolePolicy`这些控件是对本主题中描述的能力确认流程的补充。

查找并确认应用程序功能（控制台）

您可以在Amazon Serverless Application Repository 网站上找到可用的应用程序，也可以通过 Lambda 控制台（在 Amazon Serverless Application Repository 选项卡下的“创建函数”页面上）找到可用的应用程序。 Amazon Serverless Application Repository

默认情况下，要求确认用于创建自定义 IAM 角色或资源策略的功能的应用程序不会显示在搜索结果中。要搜索包含这些功能的应用程序，您必须选中 Show apps that create custom IAM roles or resource policies (显示创建自定义 IAM 角色或资源策略的应用程序) 复选框。

在您选择应用程序时，可以在 Permissions (权限) 选项卡下查看应用程序的功能。要部署应用程序，您需要选中 I acknowledge this application creates custom IAM roles or resource policies (我确认此应用程序创建自定义 IAM 角色或资源策略) 复选框。如果您不确认这些功能，则会看到以下错误消息：需要确认。要进行部署，请选中配置应用程序参数部分中的复选框。

查看应用程序功能 (Amazon CLI)

要使用查看应用程序的功能 Amazon CLI，您首先需要该应用程序的 Amazon 资源名称 (ARN)。然后，您可以执行以下命令：

```
aws serverlessrepo get-application \
--application-id application-arn
```

requiredCapabilities 响应属性包含您需要确认然后才能部署应用程序的应用程序功能列表。请注意，如果 requiredCapabilities 属性为空，则应用程序没有所需功能。

如何部署应用程序

本节向您介绍使用 Amazon Web Services 管理控制台 或从部署无服务器应用程序的过程。 Amazon Serverless Application Repository Amazon CLI

部署新应用程序（控制台）

本节介绍如何 Amazon Serverless Application Repository 使用部署新应用程序 Amazon Web Services 管理控制台。有关部署现有应用程序的新版本的说明，请参阅更新应用程序。

浏览、搜索和部署应用程序

使用以下步骤在中查找、配置和部署应用程序。 Amazon Serverless Application Repository

要在中查找和配置应用程序 Amazon Serverless Application Repository

1. 打开 Amazon Serverless Application Repository 公有主页，或打开 Amazon Lambda 控制台。选择 Create function (创建函数)，然后选择 Browse serverless app repository (浏览无服务器应用程序存储库)。
2. 浏览或搜索应用程序。

   > **Note**
   >
   > 要显示包含自定义 IAM 角色或资源策略的应用程序，请选中 Show apps that create custom IAM roles or resource policies (显示创建自定义 IAM 角色或资源策略的应用程序) 复选框。有关自定义 IAM 角色和资源策略的更多信息，请参阅确认应用程序功能。

3. 选择一个应用程序以查看详细信息，例如其权限、功能以及 Amazon 客户部署该应用程序的次数。

   将显示您尝试部署应用程序的 Amazon 区域的部署计数。
4. 在应用程序详细信息页面上，通过查看 Amazon SAM 模板、许可证和自述文件来查看应用程序的权限和应用程序资源。在此页上，您还可以找到公开共享的应用程序的 Source code URL (源代码 URL) 链接。如果应用程序包含任何嵌套应用程序，您还可以在此页面上查看嵌套应用程序的详细信息。
5. 在 Application settings (应用程序设置) 部分中配置应用程序。有关配置特定应用程序的指导，请参阅该应用程序的自述文件。

   例如，配置要求可能包括指定您希望应用程序有权访问的资源的名称。这样的资源可能是亚马逊 DynamoDB 表、亚马逊 S3 存储桶或 Amazon API Gateway API API。
6. 选择部署。这样做您会进入 Deployment status 页。

   > **Note**
   >
   > 如果应用程序具有需要确认的功能，您必须选中 I acknowledge this application creates custom IAM roles or resource policies (我确认此应用程序创建自定义 IAM 角色或资源策略) 复选框，然后才能部署应用程序。否则将导致出现错误。有关自定义 IAM 角色和资源策略的更多信息，请参阅确认应用程序功能。

7. 在 Deployment status (部署状态) 页面上，您可以查看部署的进度。在等待部署完成时，您可以搜索和浏览其他应用程序，然后通过 Lambda 控制台返回此页面。

成功部署应用程序后，您可以查看和管理使用现有 Amazon 工具创建的资源。

部署新应用程序 (Amazon CLI)

本节介绍如何使用从部署新应用程序 Amazon CLI。 Amazon Serverless Application Repository 有关部署现有应用程序的新版本的说明，请参阅更新应用程序。

查找并确认应用程序功能 (Amazon CLI)

要使用确认应用程序的功能 Amazon CLI，请执行以下步骤：

1. 查看应用程序的功能。使用以下 Amazon CLI 命令查看应用程序的功能：

   ```
   aws serverlessrepo get-application \
   --application-id application-arn
   ```

   requiredCapabilities 响应属性包含您需要确认然后才能部署应用程序的应用程序功能列表。您也可以使用中的 GetApplication API Amazon SDKs 来获取这些数据。

2. 创建变更集。创建 Amazon CloudFormation 变更集时，必须提供一组必需的权能。例如，使用以下 Amazon CLI 命令通过确认应用程序的功能来部署应用程序：

   ```
   aws serverlessrepo create-cloud-formation-change-set \
   --application-id application-arn \
   --stack-name unique-name-for-cloud-formation-stack \
   --capabilities list-of-capabilities
   ```

   成功执行此命令后，将返回更改集 ID。下一步需要更改集 ID。您也可以使用中的 CreateCloudFormationChangeSet API Amazon SDKs 来创建变更集。

   例如，以下 Amazon CLI 命令确认包含具有自定义名称的AWS::IAM::Role资源和一个或多个嵌套应用程序的应用程序：

   ```
   aws serverlessrepo create-cloud-formation-change-set \
   --application-id application-arn \
   --stack-name unique-name-for-cloud-formation-stack \
   --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND
   ```

3. 执行变更集。执行更改集将实际执行部署。提供在上一步中创建更改集时返回的更改集 ID。

   以下示例 Amazon CLI 命令执行应用程序变更集来部署应用程序：

```
aws cloudformation execute-change-set \
--change-set-name changeset-id-arn
```

您也可以使用中的 ExecuteChangeSet API Amazon SDKs 来执行变更集。

删除应用程序堆栈

要删除之前使用部署的应用程序 Amazon Serverless Application Repository，请按照与删除 Amazon CloudFormation 堆栈相同的步骤进行操作：

- Amazon Web Services 管理控制台：要使用删除应用程序 Amazon Web Services 管理控制台，请参阅Amazon CloudFormation 用户指南中的在 Amazon CloudFormation 控制台上删除堆栈。
- Amazon CLI：要使用删除应用程序 Amazon CLI，请参阅Amazon CloudFormation 用户指南中的删除堆栈。

更新应用程序

在部署了中的应用程序后 Amazon Serverless Application Repository，您可能需要对其进行更新。例如，您可能希望更改应用程序设置，或者您可能希望将应用程序更新到已发布的最新版本。

以下各节介绍如何使用 Amazon Web Services 管理控制台 或部署应用程序的新版本 Amazon CLI。

更新应用程序（控制台）

要更新之前部署的应用程序，请使用与部署新应用程序相同的过程，并提供与最初部署该应用程序相同的应用程序名称。特别是，在应用程序 Amazon Serverless Application Repository 名称前`serverlessrepo-`加上。但是，要部署应用程序的新版本，您需要提供原始应用程序名称而不在前面附加 `serverlessrepo-`。

例如，如果您部署了具有名称 `MyApplication` 的应用程序，则堆栈名称将为 `serverlessrepo-MyApplication`。要更新该应用程序，您需要`MyApplication`再次提供名称，请不要指定的完整堆栈名称。`serverlessrepo-MyApplication`

对于所有其他应用程序设置，您可以保持与之前部署相同的值，也可以提供新值。

更新应用程序 (Amazon CLI)

要更新之前部署的应用程序，请使用与部署新应用程序相同的过程，并提供与最初部署该应用程序所用的相同 *--stack-name*。特别是，在堆栈 Amazon Serverless Application Repository 名称前面serverlessrepo-加上。但是，要部署应用程序的新版本，您需要提供原始堆栈名称而不在前面附加 serverlessrepo-。

例如，如果您部署了具有堆栈名称 MyApplication 的应用程序，则创建的堆栈名称将为 serverlessrepo-MyApplication。要更新该应用程序，您需要MyApplication再次提供名称，请不要指定的完整堆栈名称。serverlessrepo-MyApplication

安全性 Amazon Serverless Application Repository

云安全 Amazon 是重中之重。作为 Amazon 客户，您可以受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

安全是双方共同承担 Amazon 的责任。责任共担模式将其描述为云的 安全性和云中 的安全性：

- 云安全 — Amazon 负责保护在 Amazon 云中运行 Amazon 服务的基础架构。 Amazon 还为您提供可以安全使用的服务。作为 Amazon 合规性计划的一部分，第三方审核人员将定期测试和验证安全性的有效性。要了解适用于 Amazon Serverless Application Repository的合规性计划，请参阅合规性计划范围内的Amazon 服务。
- 云端安全-您的责任由您使用的 Amazon 服务决定。您还需要对其他因素负责，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 Amazon Serverless Application Repository时应用责任共担模式。以下主题向您介绍如何配置 Amazon Serverless Application Repository 以满足您的安全和合规性目标。您还将学习如何使用其他 Amazon 服务来帮助您监控和保护您的 Amazon Serverless Application Repository 资源。

主题

中的数据保护 Amazon Serverless Application Repository

Amazon 责任共担模式适用于保护 Amazon Serverless Application Repository中的数据。如本模型所述 Amazon ，负责保护运行所有内容的全球基础架构 Amazon Web Services 云。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 Amazon Web Services 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅中国地区法律条款。

出于数据保护目的，我们建议您保护 Amazon Web Services 账户 凭证并使用 Amazon IAM Identity Center 或 Amazon Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 用于 SSL/TLS 与 Amazon 资源通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用设置 API 和用户活动日志 Amazon CloudTrail。有关使用 CloudTrail 跟踪捕获 Amazon 活动的信息，请参阅《Amazon CloudTrail 用户指南》中的使用跟 CloudTrail 踪。
- 使用 Amazon 加密解决方案以及其中的所有默认安全控件 Amazon Web Services 服务。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果您在 Amazon 通过命令行界面或 API 进行访问时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》https://www.amazonaws.cn/compliance/fips/。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您使用控制台、API Amazon Serverless Application Repository 或 SDK 或以其他 Amazon Web Services 服务 方式使用控制台 Amazon CLI、API 或 Amazon SDK 的情况。在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供 URL，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

传输中加密

Amazon Serverless Application Repository API 端点仅支持通过 HTTPS 进行安全连接。当您使用 Amazon Web Services 管理控制台、 Amazon SDK 或 Amazon Serverless Application Repository API 管理 Amazon Serverless Application Repository 资源时，所有通信都使用传输层安全 (TLS) 进行加密。

有关 API 终端节点的完整列表，请参阅中的Amazon 区域和终端节点Amazon Web Services 一般参考。

静态加密

会 Amazon Serverless Application Repository 加密您上传到的文件 Amazon Serverless Application Repository，包括部署包和图层档案。

的 Identity and Access Management Amazon Serverless Application Repository

Amazon Identity and Access Management (IAM) Amazon Web Services 服务 可帮助管理员安全地控制对 Amazon 资源的访问权限。IAM 管理员控制谁可以进行身份验证（登录）和授权（拥有权限）使用 Amazon Serverless Application Repository 资源。您可以使用 IAM Amazon Web Services 服务 ，无需支付额外费用。

要大致了解 IAM 的工作原理，请参阅 IAM 用户指南中的了解 IAM 的工作原理。

主题

受众

您的使用方式 Amazon Identity and Access Management (IAM) 因您的角色而异：

- 服务用户：如果您无法访问功能，请从管理员处请求权限（请参阅Amazon Serverless Application Repository 身份和访问疑难解答）
- 服务管理员：确定用户访问权限并提交权限请求（请参阅如何 Amazon Serverless Application Repository 与 IAM 配合使用）
- IAM 管理员：编写用于管理访问权限的策略（请参阅Amazon Serverless Application Repository 基于身份的策略示例）

使用身份进行身份验证

身份验证是您 Amazon 使用身份凭证登录的方式。您必须以 IAM 用户身份进行身份验证 Amazon Web Services 账户根用户，或者通过担任 IAM 角色进行身份验证。

对于编程访问， Amazon 提供 SDK 和 CLI 来对请求进行加密签名。有关更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的Amazon 签名版本 4。

Amazon Web Services 账户 root 用户

创建时 Amazon Web Services 账户，首先会有一个名为 Amazon Web Services 账户 root 用户的登录身份，该身份可以完全访问所有资源 Amazon Web Services 服务 和资源。我们强烈建议不要使用根用户进行日常任务。有关要求根用户凭证的任务，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

IAM 用户和群组

IAM 用户是对某个人员或应用程序具有特定权限的一个身份。建议使用临时凭证，而非具有长期凭证的 IAM 用户。有关更多信息，请参阅 IAM 用户指南中的要求人类用户使用身份提供商的联合身份验证才能 Amazon 使用临时证书进行访问。

IAM 组指定一组 IAM 用户，便于更轻松地对大量用户进行权限管理。有关更多信息，请参阅《IAM 用户指南》中的 IAM 用户使用案例。

IAM 角色

IAM 角色是具有特定权限的身份，可提供临时凭证。您可以通过从用户切换到 IAM 角色（控制台）或调用 Amazon CLI 或 Amazon API 操作来代入角色。有关更多信息，请参阅《IAM 用户指南》中的担任角色的方法。

IAM 角色对于联合用户访问、临时 IAM 用户权限、跨账户访问、跨服务访问以及在 Amazon EC2 上运行的应用程序非常有用。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

使用策略管理访问

您可以 Amazon 通过创建策略并将其附加到 Amazon 身份或资源来控制中的访问权限。策略定义了与身份或资源关联时的权限。 Amazon 在委托人提出请求时评估这些政策。大多数策略都以 JSON 文档的 Amazon 形式存储在中。有关 JSON 策略文档的更多信息，请参阅《IAM 用户指南》中的 JSON 策略概述。

管理员使用策略，通过定义哪个主体可以在什么条件下对哪些资源执行哪些操作来指定谁有权访问什么。

默认情况下，用户和角色没有权限。IAM 管理员创建 IAM 策略并将其添加到角色中，然后用户可以担任这些角色。IAM 策略定义权限，与执行操作所用的方法无关。

基于身份的策略

基于身份的策略是您附加到身份（用户、组或角色）的 JSON 权限策略文档。这些策略控制身份可以执行什么操作、对哪些资源执行以及在什么条件下执行。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

基于身份的策略可以是内联策略（直接嵌入到单个身份中）或托管策略（附加到多个身份的独立策略）。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。您必须在基于资源的策略中指定主体。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 Amazon 托管策略。

访问控制列表 (ACLs)

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3 和 Amazon VPC 就是支持的服务示例 ACLs。 Amazon WAF要了解更多信息 ACLs，请参阅《亚马逊简单存储服务开发者指南》中的访问控制列表 (ACL) 概述。

其他策略类型

Amazon 支持其他策略类型，这些策略类型可以设置更常见的策略类型授予的最大权限：

- 权限边界 – 设置基于身份的策略可以授予 IAM 实体的最大权限。有关更多信息，请参阅《 IAM 用户指南》中的 IAM 实体的权限边界。
- 服务控制策略 (SCPs)-在中指定组织或组织单位的最大权限 Amazon Organizations。有关更多信息，请参阅《Amazon Organizations 用户指南》中的服务控制策略。
- 资源控制策略 (RCPs)-设置账户中资源的最大可用权限。有关更多信息，请参阅《Amazon Organizations 用户指南》中的资源控制策略 (RCPs)。
- 会话策略 – 在为角色或联合用户创建临时会话时，作为参数传递的高级策略。有关更多信息，请参阅《IAM 用户指南》中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 Amazon 确定是否允许请求，请参阅 IAM 用户指南中的策略评估逻辑。

如何 Amazon Serverless Application Repository 与 IAM 配合使用

在使用 IAM 管理对的访问权限之前 Amazon Serverless Application Repository，您应该了解有哪些 IAM 功能可供使用 Amazon Serverless Application Repository。

要大致了解 IAM 的工作原理，请参阅 IAM 用户指南中的了解 IAM 的工作原理。要全面了解这些 Amazon 服务 Amazon Serverless Application Repository 和其他服务如何与 IAM 配合使用，请参阅 IAM 用户指南中的与 IAM 配合使用的Amazon 服务。

主题

- Amazon Serverless Application Repository 基于身份的策略
- Amazon Serverless Application Repository 应用程序政策
- 基于 Amazon Serverless Application Repository 标签的授权
- Amazon Serverless Application Repository IAM 角色

Amazon Serverless Application Repository 基于身份的策略

通过使用 IAM 基于身份的策略，可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。Amazon Serverless Application Repository 支持特定的操作、资源和条件键。要了解在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》 中的 IAM JSON 策略元素参考。

下面介绍权限策略示例。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:CreateApplication"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "CreateApplicationVersion",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:CreateApplicationVersion"
            ],
            "Resource": "arn:aws:serverlessrepo:us-
east-1:111122223333:applications/application-name"
        }
    ]
}
```

该策略包含两条语句：

- 第一条语句授予`serverlessrepo:CreateApplication`对所有 Amazon Serverless Application Repository 资源 Amazon Serverless Application Repository 执行操作的权限，`Resource`值由通配符 (*) 指定。
- 第二条语句使用应用程序的 Amazon Amazon 资源名称 (ARN) 来授予对资源 Amazon Serverless Application Repository 执行`serverlessrepo:CreateApplicationVersion`操作的 Amazon Serverless Application Repository 权限。通过 `Resource` 值来指定应用程序。

该策略不指定 `Principal` 元素，因为在基于身份的策略中，您未指定获取权限的委托人。附加了策略的用户是隐式委托人。向 IAM 角色附加权限策略后，该角色的信任策略中标识的委托人将获取权限。

有关显示所有 Amazon Serverless Application Repository API 操作及其适用的 Amazon 资源的表格，请参阅Amazon Serverless Application Repository API 权限：操作和资源参考。

操作

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。在策略中包含操作以授予执行关联操作的权限。

中的策略操作在操作前 Amazon Serverless Application Repository 使用以下前缀:`serverlessrepo:`. 例如，要授予某人 Amazon Serverless Application Repository `SearchApplications`通

过 API 操作运行 Amazon Serverless Application Repository 实例的权限，您需要将该`serverlessrepo:SearchApplications`操作包含在他们的策略中。策略语句必须包含 `Action` 或 `NotAction` 元素。 Amazon Serverless Application Repository 定义了自己的一组操作，这些操作描述了您可以使用此服务执行的任务。

要在单个语句中指定多项操作，请使用逗号将它们隔开，如下所示：

```
"Action": [
      "serverlessrepo:action1",
      "serverlessrepo:action2"
]
```

您也可以使用通配符（*) 指定多个操作。例如，要指定以单词 `List` 开头的所有操作，包括以下操作：

```
"Action": "serverlessrepo:List*"
```

要查看 Amazon Serverless Application Repository 操作列表，请参阅 IAM 用户指南 Amazon Serverless Application Repository中的定义操作。

资源

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。作为最佳实践，请使用其 Amazon 资源名称（ARN）指定资源。对于不支持资源级权限的操作，请使用通配符 (*) 指示语句应用于所有资源。

```
"Resource": "*"
```

在中 Amazon Serverless Application Repository，主要 Amazon 资源是 Amazon Serverless Application Repository 应用程序。 Amazon Serverless Application Repository 应用程序具有与其关联的唯一 Amazon 资源名称 (ARNs)，如下表所示。

Amazon 资源类型	Amazon 资源名称 (ARN) 格式
应用程序	arn:: serverless存储库::: applicat *partition* ions/ *region* *account-id* *application-name*

有关格式的更多信息 ARNs，请参阅 Amazon 资源名称 (ARNs) 和 Amazon 服务命名空间。

以下是一个策略示例，该策略授予对所有 Amazon 资源的`serverlessrepo:ListApplications`操作权限。在当前的实现中， Amazon Serverless Application Repository 不支持通过使用某些 API 操作的 Amazon Amazon 资源 ARNs （也称为资源级权限）来识别特定资源。在这些情况下，您必须指定通配符 (*)。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListExistingApplications",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:ListApplications"
            ],
            "Resource": "*"
        }
    ]
}
```

有关显示所有 Amazon Serverless Application Repository API 操作及其适用的 Amazon 资源的表格，请参阅Amazon Serverless Application Repository API 权限：操作和资源参考。

条件键

Amazon Serverless Application Repository 不提供任何特定于服务的条件密钥，但它确实支持使用一些全局条件密钥。要查看所有 Amazon 全局条件键，请参阅 IAM 用户指南中的Amazon 全局条件上下文密钥。

示例

要查看 Amazon Serverless Application Repository 基于身份的策略的示例，请参阅。Amazon Serverless Application Repository 基于身份的策略示例

Amazon Serverless Application Repository 应用程序政策

应用程序策略决定了指定的委托人或 PrincipalOrg 可以对应用程序执行的 Amazon Serverless Application Repository 操作。

您可以向与 Amazon Serverless Application Repository 应用程序关联的策略添加权限。附加到 Amazon Serverless Application Repository 应用程序的权限策略称为应用程序策略。应用程序策略是基于 IAM 资源的策略的扩展。主要资源是 Amazon Serverless Application Repository 应用程序。您可以使用 Amazon Serverless Application Repository 应用程序策略来管理应用程序部署权限。

Amazon Serverless Application Repository 应用程序策略主要由发布者用来向消费者授予部署其应用程序的权限以及相关操作（例如搜索和查看这些应用程序的详细信息）。发布者可以将应用程序权限设置为以下三个类别：

- 私有 — 使用同一个账户创建且未与任何其他账户共享的应用程序。您有权部署使用您的 Amazon 账户创建的应用程序。
- 私下共享 — 发布者已明确与一组特定 Amazon 账户或 Amazon Organizations 共享的应用程序。您有权部署已与您的 Amazon 账户或 Amazon 组织共享的应用程序。
- 公开共享-发布者与所有人共享的应用程序。您有权部署任何公开共享的应用程序。

您可以使用、或 Amazon CLI Amazon SDKs，来授予权限 Amazon Web Services 管理控制台。

示例

要查看管理 Amazon Serverless Application Repository 应用程序策略的示例，请参阅Amazon Serverless Application Repository 应用程序策略示例。

基于 Amazon Serverless Application Repository 标签的授权

Amazon Serverless Application Repository 不支持根据标签控制对资源或操作的访问权限。

Amazon Serverless Application Repository IAM 角色

I AM 角色是您的 Amazon 账户中具有特定权限的实体。

将临时证书与 Amazon Serverless Application Repository

您可以使用临时凭证进行联合身份登录，来担任 IAM 角色或担任跨账户角色。您可以通过调用AssumeRole或之类的 Amazon STS API 操作来获取临时安全证书GetFederationToken。

Amazon Serverless Application Repository 支持使用临时证书。

服务相关角色

Amazon Serverless Application Repository 不支持与服务相关的角色。

服务角色

Amazon Serverless Application Repository 不支持服务角色。

Amazon Serverless Application Repository 基于身份的策略示例

默认情况下，IAM 用户和角色没有创建或修改 Amazon Serverless Application Repository 资源的权限。他们也无法使用 Amazon Web Services 管理控制台 Amazon CLI、或 Amazon API 执行任务。IAM 管理员必须创建 IAM 策略，以便为用户和角色授予权限以对所需的指定资源执行特定的 API 操作。然后，管理员必须将这些策略附加到需要这些权限的 IAM 用户或组。

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅 I AM 用户指南中的 “在 JSON” 选项卡上创建策略。

主题

策略最佳实践

基于身份的策略非常强大。它们决定是否有人可以在您的账户中创建、访问或删除 Amazon Serverless Application Repository 资源。这些操作可能会给您的 Amazon 账户带来费用。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- 授予最低权限：创建自定义策略时，仅授予执行任务所需的许可。最开始只授予最低权限，然后根据需要授予其它权限。这样做比起一开始就授予过于宽松的权限而后再尝试收紧权限来说更为安全。有关更多信息，请参阅 IAM 用户指南中的授予最低权限。
- 为敏感操作启用 MFA – 为增强安全性，要求 IAM 用户使用多重身份验证（MFA) 来访问敏感资源或 API 操作。有关更多信息，请参阅 IAM 用户指南中的在 Amazon中使用多重身份验证（MFA）。
- 使用策略条件来增强安全性 – 在切实可行的范围内，定义基于身份的策略在哪些情况下允许访问资源。例如，您可编写条件来指定请求必须来自允许的 IP 地址范围。您也可以编写条件，以便仅允许

指定日期或时间范围内的请求，或者要求使用 SSL 或 MFA。有关更多信息，请参阅 IAM 用户指南中的 IAM JSON 策略元素：条件。

使用控制 Amazon Serverless Application Repository 台

Amazon Serverless Application Repository 控制台为您提供了一个用于发现和管理 Amazon Serverless Application Repository 应用程序的集成环境。该控制台提供的功能和工作流程通常需要管理 Amazon Serverless Application Repository 应用程序的权限，此外还需要中记录的特定于 API 的权限。Amazon Serverless Application Repository API 权限：操作和资源参考

有关使用 Amazon Serverless Application Repository 控制台所需权限的更多信息，请参阅客户托管策略示例。

允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上或使用 Amazon CLI 或 Amazon API 以编程方式完成此操作的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
```

```
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

客户托管策略示例

此部分中的示例提供了一组可附加到用户的示例策略。如果您是首次创建策略，建议您先在账户中创建 IAM 用户并按顺序将策略附加到用户。您还可以使用这些示例创建单个自定义策略（其中包括执行多个操作的权限），然后将其附加到用户。

有关如何向用户关联策略的更多信息，请参阅 IAM 用户指南中的向用户添加权限。

示例

- 发布者示例 1：允许发布者列出应用程序
- 发布者示例 2：允许发布者查看应用程序或应用程序版本的详细信息
- 发布者示例 3：允许发布者创建应用程序或应用程序版本
- 发布者示例 4：允许发布者创建应用程序策略以与其他人共享应用程序
- 使用者示例 1：允许使用者搜索应用程序
- 使用者示例 2：允许使用者查看应用程序的详细信息
- 使用者示例 3：允许使用者部署应用程序
- 使用者示例 4：拒绝访问部署资产
- 使用者示例 5：防止使用者搜索和部署公有应用程序

发布者示例 1：允许发布者列出应用程序

您的账户中的 IAM 用户必须先具有 `serverlessrepo:ListApplications` 操作权限，然后才能在控制台中查看任何内容。当您授予这些权限时，控制台可以显示在该用户所属的特定 Amazon 区域中创建的 Amazon 账户中的 Amazon Serverless Application Repository 应用程序列表。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListExistingApplications",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:ListApplications"
            ],
            "Resource": "*"
        }
    ]
}
```

发布者示例 2：允许发布者查看应用程序或应用程序版本的详细信息

用户可以选择 Amazon Serverless Application Repository 应用程序并查看该应用程序的详细信息。此类详细信息包括作者、说明、版本和其他配置信息。为此，用户需要 Amazon Serverless Application Repository的 `serverlessrepo:ListApplicationVersions` 和 `serverlessrepo:GetApplication` API 操作的权限。

在以下示例中，为将其 Amazon 资源名称 (ARN) 指定为 `Resource` 值的特定应用程序授予这些权限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:GetApplication",
                "serverlessrepo:ListApplicationVersions"
            ],
            "Resource": "arn:aws:serverlessrepo:us-
east-1:111122223333:applications/application-name"
        }
    ]
}
```

发布者示例 3：允许发布者创建应用程序或应用程序版本

如果要允许用户拥有创建 Amazon Serverless Application Repository 应用程序的权限，则需要向`serverlessrepo:CreateApplication`和`serverlessrepo:CreateApplicationVersions`操作授予权限，如以下策略所示。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:CreateApplication",
                "serverlessrepo:CreateApplicationVersion"
            ],
            "Resource": "*"
        }
    ]
}
```

发布者示例 4：允许发布者创建应用程序策略以与其他人共享应用程序

要使用户与其他人共享应用程序，您必须向这些用户授予创建应用程序策略的权限，如以下策略所示。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ShareApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:PutApplicationPolicy",
```

```
                "serverlessrepo:GetApplicationPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

使用者示例 1：允许使用者搜索应用程序

要使使用者能够搜索应用程序，您必须向他们授予以下权限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "SearchApplications",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:SearchApplications"
            ],
            "Resource": "*"
        }
    ]
}
```

使用者示例 2：允许使用者查看应用程序的详细信息

用户可以选择 Amazon Serverless Application Repository 应用程序并查看应用程序的详细信息，例如作者、描述、版本和其他配置信息。为此，用户必须具有执行以下 Amazon Serverless Application Repository 操作的权限。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:GetApplication",
                "serverlessrepo:ListApplicationVersions"
            ],
            "Resource": "*"
        }
    ]
}
```

使用者示例 3：允许使用者部署应用程序

要使使用者能够部署应用程序，您必须向他们授予执行许多操作的权限。以下策略为客户提供了所需权限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DeployApplication",
            "Effect": "Allow",
            "Action": [
                "serverlessrepo:CreateCloudFormationChangeSet",
                "cloudformation:CreateChangeSet",
                "cloudformation:ExecuteChangeSet",
                "cloudformation:DescribeStacks"

            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> 部署应用程序可能需要权限才能使用其他 Amazon 资源。由于 Amazon Serverless Application Repository 使用的底层部署机制与相同，因此有关更多信息 Amazon CloudFormation，请参阅使用 Ident Amazon ity and Access Management 控制访问权限。如需有关解决与权限相关的部署问题的帮助，请参阅问题排查：IAM 权限不足。

使用者示例 4：拒绝访问部署资产

当应用程序与 Amazon 账户私下共享时，默认情况下，该账户中的所有用户都可以访问同一账户中所有其他用户的部署资产。以下策略禁止账户中的用户访问存储在 Amazon S3 存储桶中的部署资产 Amazon Serverless Application Repository。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeploymentAssetAccess",
            "Effect": "Deny",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::awsserverlessrepo-changesets*/*"
            ]
        }
    ]
}
```

使用者示例 5：防止使用者搜索和部署公有应用程序

您可以阻止用户对应用程序执行某些操作。

以下策略通过将 `serverlessrepo:applicationType` 指定为 `public` 来应用于公有应用程序。它可以通过将 `Effect` 指定为 `Deny` 来阻止用户执行许多操作。有关可用条件键的更多信息 Amazon

Serverless Application Repository，请参阅的操作、资源和条件键 Amazon Serverless Application Repository。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Condition": {
                "StringEquals": {
                    "serverlessrepo:applicationType": "public"
                }
            },
            "Action": [
                "serverlessrepo:SearchApplications",
                "serverlessrepo:GetApplication",
                "serverlessrepo:CreateCloudFormationTemplate",
                "serverlessrepo:CreateCloudFormationChangeSet",
                "serverlessrepo:ListApplicationVersions",
                "serverlessrepo:ListApplicationDependencies"
            ],
            "Resource": "*",
            "Effect": "Deny"
        }
    ]
}
```

> **Note**
>
> 此政策声明也可以用作服务控制策略并应用于 Amazon 组织。有关服务控制策略的更多信息，请参阅《Amazon Organizations 用户指南》中的服务控制策略。

Amazon Serverless Application Repository 应用程序策略示例

附加到 Amazon Serverless Application Repository 应用程序的权限策略称为应用程序策略。应用程序策略决定了指定的委托人或 PrincipalOrg 可以对应用程序执行的 Amazon Serverless Application Repository 操作。

Amazon Serverless Application Repository 应用程序是中的主要 Amazon 资源 Amazon Serverless Application Repository。 Amazon Serverless Application Repository 应用程序策略主要由发布者用来向消费者授予部署其应用程序的权限以及相关操作（例如搜索和查看这些应用程序的详细信息）。

发布者可以将应用程序权限设置为以下三个类别：

- 私有 — 使用同一个账户创建且未与任何其他账户共享的应用程序。只有共享您 Amazon 账户的消费者才有权部署私有应用程序。
- 私下共享 — 发布者已明确与一组特定 Amazon 帐户或 Amazon 组织中的 Amazon 帐户共享的应用程序。消费者有权部署已与其 Amazon 账户或 Amazon 组织共享的应用程序。有关 Amazon 组织的更多信息，请参阅《Amazon Organizations 用户指南》。
- 公开共享-发布者与所有人共享的应用程序。所有使用者都有权部署任何公开共享的应用程序。

> **Note**
>
> 对于私有共享的应用程序， Amazon Serverless Application Repository 仅支持Amazon 账户作为委托人。发布者可以将一个 Amazon 账户中的所有用户作为一个群组授予或拒绝使用某个 Amazon Serverless Application Repository 应用程序。发布者不能授予或拒绝 Amazon 账户内的个人用户使用 Amazon Serverless Application Repository 应用程序。

有关使用设置应用程序权限的说明 Amazon Web Services 管理控制台，请参阅共享应用程序。

有关使用 Amazon CLI 和示例设置应用程序权限的说明，请参阅以下各节。

应用程序权限（Amazon CLI 和 Amazon SDKs）

使用 Amazon CLI 或 Amazon SDKs 为 Amazon Serverless Application Repository 应用程序设置权限时，可以指定以下操作：

Action	说明
GetApplication	授予查看有关应用程序的信息的权限。
CreateCloudFormationChangeSet	授予部署应用程序的权限。 注意：此操作不会授予除部署之外的任何其他权限。

Action	说明
CreateCloudFormationTemplate	授予为应用程序创建 Amazon CloudFormation 模板的权限。
ListApplicationVersions	授予列出应用程序的版本的权限。
ListApplicationDependencies	授予列出包含应用程序中嵌套的应用程序的权限。
SearchApplications	授予搜索应用程序的权限。
部署	此操作启用表中前面列出的所有操作。也就是说，它授予查看应用程序、部署应用程序、列出版本以及搜索应用程序的权限。

应用程序策略示例

以下示例演示如何使用 Amazon CLI授予权限。有关如何使用授予权限的信息 Amazon Web Services 管理控制台，请参阅共享应用程序。

本节中的所有示例都使用以下 Amazon CLI 命令来管理与 Amazon Serverless Application Repository 应用程序关联的权限策略：

- put-application-policy
- get-application-policy

主题

- 示例 1：与其他账户共享应用程序
- 示例 2：公开共享应用程序
- 示例 3：使应用程序成为私有的
- 示例 4：指定多个账户和权限
- 示例 5：与 Amazon 组织中的所有账户共享应用程序
- 示例 6：与 Amazon 组织中的某些账户共享应用程序
- 示例 7：检索应用程序策略
- 示例 8：允许特定账户嵌套应用程序

示例 1：与其他账户共享应用程序

要与其他特定账户共享应用程序，但不允许与他人共享，请指定要与之共享的 Amazon 账户 ID 作为委托人。这也称为将应用程序设置为私下共享。为此，请使用以下 Amazon CLI 命令。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=account-id,Actions=Deploy
```

> **Note**
>
> 私下共享的应用程序只能在创建应用程序的同一 Amazon 区域中使用。

示例 2：公开共享应用程序

要使应用程序公开，请通过将“*”指定为委托人来与每个人共享应用程序，如以下示例所示。公开共享的应用程序在所有区域中都可用。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=*,Actions=Deploy
```

> **Note**
>
> 要公开共享应用程序，它必须同时设置了 `SemanticVersion` 和 `LicenseUrl` 属性。

示例 3：使应用程序成为私有的

您可以将应用程序设为私有，这样它就不会与任何人共享，只能由拥有该应用程序的 Amazon 账户进行部署。为此，您需要从策略中清除委托人和操作，该策略还会删除 Amazon 组织内其他账户部署应用程序的权限。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements '[]'
```

> **Note**
>
> 私有应用程序只能在创建应用程序的同一 Amazon 区域中使用。

示例 4：指定多个账户和权限

您可以授予多个权限，也可以一次向多个 Amazon 账户授予这些权限。为此，您可以将列表指定为委托人和操作，如以下示例所示。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=account-id-1,account-
id-2,Actions=GetApplication,CreateCloudFormationChangeSet
```

示例 5：与 Amazon 组织中的所有账户共享应用程序

可以向 Amazon 组织内的所有用户授予权限。您可以通过指定组织 ID 来执行此操作，如以下示例所示。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=*,PrincipalOrgIDs=org-id,Actions=Deploy,UnshareApplication
```

有关 Amazon 组织的更多信息，请参阅《Amazon Organizations 用户指南》。

> **Note**
>
> 您只能指定您的 Amazon 账户所属的 Amazon 组织。如果您尝试指定一个您不是其成员的 Amazon 组织，则会出现错误。
>
> 要与您的 Amazon 组织共享您的应用程序，您必须包含`UnshareApplication`操作权限，以防将来需要撤消共享。

示例 6：与 Amazon 组织中的某些账户共享应用程序

可以向 Amazon 组织内的特定账户授予权限。为此，您可以将 Amazon 账户列表指定为委托人，并指定您的组织 ID，如下例所示。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=account-id-1,account-id-2,PrincipalOrgIDs=org-
id,Actions=Deploy,UnshareApplication
```

> **Note**
>
> 您只能指定您的 Amazon 账户所属的 Amazon 组织。如果您尝试指定一个您不是其成员的 Amazon 组织，则会出现错误。
> 要与您的 Amazon 组织共享您的应用程序，您必须包含UnshareApplication操作权限，以防将来需要撤消共享。

示例 7：检索应用程序策略

要查看应用程序的当前策略，例如，要查看它当前是否共享，您可以使用 get-application-policy 命令，如以下示例所示。

```
aws serverlessrepo get-application-policy \
--region region \
--application-id application-arn
```

示例 8：允许特定账户嵌套应用程序

允许任何人嵌套公有应用程序。如果您希望只允许特定账户嵌套您的应用程序，则必须设置以下最低权限，如以下示例所示。

```
aws serverlessrepo put-application-policy \
--region region \
--application-id application-arn \
--statements Principals=account-id-1,account-
id-2,Actions=GetApplication,CreateCloudFormationTemplate
```

Amazon Serverless Application Repository API 权限：操作和资源参考

在设置访问控制和编写可附加到 IAM 身份的权限策略（基于身份的策略）时，您可以将下表作为参考。该每个 Amazon Serverless Application Repository API 操作、您可以授予执行该操作的权限的相

应操作以及您可以授予权限的 Amazon 资源。您可以在策略的 `Action` 字段中指定这些操作，并在策略的 `Resource` 字段中指定资源值。

要指定操作，请在 API 操作名称之前使用 `serverlessrepo:` 前缀（例如，`serverlessrepo:ListApplications`）。

操作	URI	方法	Amazon 资源 (ARNs)
操作： ListApplications 所需权限：无服务器存储库：ListApplications	/applications	GET	*
操作： CreateApplication 所需权限：无服务器存储库：CreateApplication	/applications	POST	*
操作： GetApplication 所需权限：无服务器存储库：GetApplication	/应用程序/*application-id*	GET	arn: aws: serverless 存储库::: applications/*region account-id application-name*
操作： DeleteApplication 所需权限：无服务器存储库：DeleteApplication	/应用程序/*application-id*	DELETE	arn: aws: serverless 存储库::: applications/*region account-id application-name*
操作： UpdateApplication	/应用程序/*application-id*	PATCH	arn: aws: serverless 存储库::: applications/*region account-i*

操作	URI	方法	Amazon 资源 (ARNs)
所需权限：无服务器存储库：UpdateApplication			*d application-name*
操作： CreateCloudFormationChangeSet 所需权限：无服务器存储库：CreateCloudFormationChangeSet	/applications/ /changesets *application-id*	POST	arn: aws: serverless存储库::: applications/ *region account-id application-name*
操作： GetApplicationPolicy 所需权限：无服务器存储库：GetApplicationPolicy	/applications/ /政策 *application-id*	GET	arn: aws: serverless存储库::: applications/ *region account-id application-name*
操作： PutApplicationPolicy 所需权限：无服务器存储库：PutApplicationPolicy	/applications/ /政策 *application-id*	PUT	arn: aws: serverless存储库::: applications/ *region account-id application-name*
操作： ListApplicationVersions 所需权限：无服务器存储库：ListApplicationVersions	/应用程序/ /版本 *application-id*	GET	arn: aws: serverless存储库::: applications/ *region account-id application-name*

操作	URI	方法	Amazon 资源 (ARNs)
操作： CreateApplicationVersion 所需权限：无服务器存储库：CreateApplicationVersion	/应用程序/ /版本/ *application-id semantic-version*	PUT	arn: aws: serverless 存储库::: applications/ *region account-id application-name*
操作： ListApplicationDependencies 所需权限：无服务器存储库：ListApplicationDependencies	/applications/ /依赖关系 *application-id*	GET	arn: aws: serverless 存储库::: applications/ *region account-id application-name*
操作： SearchApplications 所需权限：无服务器存储库：SearchApplications	不适用	不适用	*

Amazon Serverless Application Repository 身份和访问疑难解答

使用以下信息来帮助您诊断和修复在使用 Amazon Serverless Application Repository 和 IAM 时可能遇到的常见问题。

主题

- 我无权在中执行操作 Amazon Serverless Application Repository
- 我无权执行 iam：PassRole
- 我是一名管理员，想允许其他人访问 Amazon Serverless Application Repository
- 我想允许 Amazon 账户以外的人访问我的 Amazon Serverless Application Repository 资源

我无权在中执行操作 Amazon Serverless Application Repository

如果 Amazon Web Services 管理控制台 告诉您您无权执行某项操作，则必须联系管理员寻求帮助。管理员是指提供用户名和密码的人员。

当 `mateojackson` IAM 用户尝试使用控制台查看有关应用程序的详细信息但没有`serverlessrepo:GetApplication`权限时，就会出现以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 serverlessrepo:GetApplication on resource: my-example-application
```

在这种情况下，Mateo 请求管理员更新其策略，以允许他使用 `serverlessrepo:GetApplication` 操作访问 *my-example-application* 资源。

我无权执行 iam：PassRole

如果您收到一个错误，表明您无权执行 `iam:PassRole` 操作，则必须更新策略以允许您将角色传递给。 Amazon Serverless Application Repository

有些 Amazon Web Services 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 `marymajor` 的 IAM 用户尝试使用控制台在 Amazon Serverless Application Repository中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 `iam:PassRole` 操作。

如果您需要帮助，请联系您的 Amazon 管理员。您的管理员是提供登录凭证的人。

我是一名管理员，想允许其他人访问 Amazon Serverless Application Repository

要允许其他人访问 Amazon Serverless Application Repository，您必须向需要访问的人员或应用程序授予权限。如果使用 Amazon IAM Identity Center 管理人员和应用程序，则可以向用户或组分配权限集来定义其访问权限级别。权限集会自动创建 IAM 策略并将其分配给与人员或应用程序关联的 IAM 角色。有关更多信息，请参阅《Amazon IAM Identity Center 用户指南》中的权限集。

如果未使用 IAM Identity Center，则必须为需要访问的人员或应用程序创建 IAM 实体（用户或角色）。然后，您必须将策略附加到实体，以便在 Amazon Serverless Application Repository中向其授予正确的权限。授予权限后，向用户或应用程序开发人员提供凭证。他们将使用这些凭证访问 Amazon。要了解有关创建 IAM 用户、组、策略和权限的更多信息，请参阅《IAM 用户指南》中的 IAM 身份和 IAM 中的策略和权限。

我想允许 Amazon 账户以外的人访问我的 Amazon Serverless Application Repository 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以代入角色。对于支持基于资源的策略或访问控制列表 (ACLs) 的服务，您可以使用这些策略向人们授予访问您的资源的权限。

要了解更多信息，请参阅以下内容：

- 要了解是否 Amazon Serverless Application Repository 支持这些功能，请参阅如何 Amazon Serverless Application Repository 与 IAM 配合使用。
- 要了解如何提供对您拥有的资源的访问权限 Amazon Web Services 账户 ，请参阅 IAM 用户指南中的向您拥有 Amazon Web Services 账户 的另一个 IAM 用户提供访问权限。
- 要了解如何向第三方提供对您的资源的访问权限 Amazon Web Services 账户，请参阅 IAM 用户指南中的向第三方提供访问权限。 Amazon Web Services 账户
- 要了解如何通过身份联合验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（身份联合验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

中的日志和监控 Amazon Serverless Application Repository

监控是维护 Amazon 解决方案可靠性、可用性和性能的重要组成部分。您应该从 Amazon 解决方案的所有部分收集监控数据，以便在出现多点故障时可以更轻松地进行调试。 Amazon 提供了多种用于监控您的 Amazon Serverless Application Repository 资源和应对潜在事件的工具，例如：

Amazon CloudTrail 日志

> 与 Amazon CloudTrail一项服务集成，该服务提供用户、角色或 Amazon 服务在中采取的操作的记录 Amazon Serverless Application Repository。 Amazon Serverless Application Repository CloudTrail 捕获 Amazon Serverless Application Repository 作为事件的所有 API 调用。

主题

使用记录 Amazon Serverless Application Repository API 调用 Amazon CloudTrail

Amazon Serverless Application Repository 与集成 Amazon CloudTrail，后者是一项服务，用于记录用户、角色或 Amazon 服务在中采取的操作 Amazon Serverless Application Repository。 CloudTrail 捕获 Amazon Serverless Application Repository 作为事件的所有 API 调用。捕获的调用包括来自 Amazon Serverless Application Repository 控制台的调用和对 Amazon Serverless Application Repository API 操作的代码调用。

如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括的事件 Amazon Serverless Application Repository。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向提出的请求 Amazon Serverless Application Repository。还可以确定发出请求的源 IP 地址、请求方、发出请求的时间以及其他详细信息。

要了解更多信息 CloudTrail，请参阅《Amazon CloudTrail 用户指南》。

Amazon Serverless Application Repository 中的信息 CloudTrail

CloudTrail 在您创建 Amazon 账户时已在您的账户上启用。当活动发生在中时 Amazon Serverless Application Repository，该活动会与其他 Amazon 服务 CloudTrail 事件一起记录在事件历史记录中。您可以在自己的 Amazon 账户中查看、搜索和下载最近发生的事件。有关更多信息，请参阅使用事件历史记录查看 CloudTrail 事件。

要持续记录您 Amazon 账户中的事件，包括的事件 Amazon Serverless Application Repository，请创建跟踪。跟踪允许 CloudTrail将日志文件传输到 Amazon S3 存储桶。默认情况下，当您在控制台中创建跟踪时，该跟踪将应用于所有 Amazon 区域。跟踪记录 Amazon 分区中所有 Amazon 区域的事件，并将日志文件传送到您指定的 Amazon S3 存储桶。此外，您可以配置其他 Amazon 服务，以进一步分析和处理 CloudTrail 日志中收集的事件数据。有关更多信息，请参阅下列内容：

- 创建跟踪概述
- CloudTrail 支持的服务和集成
- 配置 Amazon SNS 通知 CloudTrail
- 接收来自多个区域的 CloudTrail 日志文件和接收来自多个账户的 CloudTrail 日志文件

所有 Amazon Serverless Application Repository 操作都由 “Amazon Serverless Application Repository 资源” 页面记录 CloudTrail 并记录在案。例如，对CreateApplicationUpdateApplications、和ListApplications操作的调用会在 CloudTrail 日志文件中生成条目。

每个事件或日志条目都包含有关生成请求的人员信息。身份信息有助于您确定以下内容：

- 请求是使用根证书还是 Amazon Identity and Access Management (IAM) 用户凭证发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 Amazon 服务发出。

有关更多信息，请参阅 CloudTrail userIdentity 元素。

了解 Amazon Serverless Application Repository 日志文件条目

跟踪是一种配置，允许将事件作为日志文件传输到您指定的 Amazon S3 存储桶。 CloudTrail 日志文件包含一个或多个日志条目。事件代表来自任何来源的单个请求，包括有关请求的操作、操作的日期和时间、请求参数等的信息。 CloudTrail 日志文件不是公共 API 调用的有序堆栈跟踪，因此它们不会按任何特定的顺序出现。

以下示例显示了演示该CreateApplication操作的 CloudTrail 日志条目。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "Root",
        "principalId": "999999999999",
        "arn": "arn:aws:iam::999999999999:root",
        "accountId": "999999999999",
        "accessKeyId": "ASIAUVPLBDH76HEXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-07-30T16:40:42Z"
            }
        },
        "invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2018-07-30T17:37:37Z",
    "eventSource": "serverlessrepo.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-east-1",
```

```
    "sourceIPAddress": "72.21.217.161",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
        "licenseBody": "<content of license>",
        "sourceCodeUrl": "<sample url>",
        "spdxLicenseId": "<sample license id>",
        "readmeBody": "<content of readme>",
        "author": "<author name>",
        "templateBody": "<content of SAM template>",
        "name": "<application name>",
        "semanticVersion": "<version>",
        "description": "<content of description>",
        "homePageUrl": "<sample url>",
        "labels": [
            "<label1>",
            "<label2>"
        ]
    },
    "responseElements": {
        "licenseUrl": "<url to access content of license>",
        "readmeUrl": "<url to access content of readme>",
        "spdxLicenseId": "<sample license id>",
        "creationTime": "2018-07-30T17:37:37.045Z",
        "author": "<author name>",
        "name": "<application name>",
        "description": "<content of description>",
        "applicationId": "arn:aws:serverlessrepo:us-
east-1:999999999999:applications/<application name>",
        "homePageUrl": "<sample url>",
        "version": {
            "applicationId": "arn:aws:serverlessrepo:us-
east-1:999999999999:applications/<application name>",
            "semanticVersion": "<version>",
            "sourceCodeUrl": "<sample url>",
            "templateUrl": "<url to access content of SAM template>",
            "creationTime": "2018-07-30T17:37:37.027Z",
            "parameterDefinitions": [
                {
                    "name": "<parameter name>",
                    "description": "<parameter description>",
                    "type": "<parameter type>"
                }
            ]
        },
```

```
        "labels": [
            "<label1>",
            "<label2>"
        ]
    },
    "requestID": "3f50d899-941f-11e8-ab18-01063f863be5",
    "eventID": "a66a6490-d388-4a4f-8c7b-9d6ec61ab262",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "999999999999"
}
```

的合规性验证 Amazon Serverless Application Repository

Amazon Serverless Application Repository 作为多个合规计划的一部分，第三方审计师会评估其安全 Amazon 性和合规性。这些合规性计划包括 SOC、PCI、FedRAMP 等。

有关特定合规计划范围内的 Amazon 服务列表，请参阅按合规计划划分的范围内的Amazon 服务。有关一般信息，请参阅 Amazon 合规性计划。

您可以使用下载第三方审计报告 Amazon Artifact。有关更多信息，请参阅在 Artifac t 中 Amazon 下载报告。

您在使用时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。 Amazon Serverless Application Repository Amazon 提供了以下资源来帮助实现合规性：

- 安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在上部署以安全为重点和以合规为重点的基准环境的步骤。 Amazon
- Amazon 合规资源 — 此工作簿和指南集可能适用于您所在的行业和所在地。
- Amazon Config— 该 Amazon 服务评估您的资源配置在多大程度上符合内部实践、行业指导方针和法规。
- Amazon Security Hub CSPM— 此 Amazon 服务可全面了解您的安全状态 Amazon ，帮助您检查是否符合安全行业标准和最佳实践。

中的韧性 Amazon Serverless Application Repository

Amazon 全球基础设施是围绕 Amazon 区域和可用区构建的。 Amazon 区域提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络相连。利用可用区，您可以设计和操作

在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础架构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 Amazon 区域和可用区的更多信息，请参阅Amazon 全球基础设施。

中的基础设施安全 Amazon Serverless Application Repository

作为一项托管服务 Amazon Serverless Application Repository ，受 Amazon 全球网络安全的保护。有关 Amazon 安全服务以及如何 Amazon 保护基础设施的信息，请参阅Amazon 云安全。要使用基础设施安全的最佳实践来设计您的 Amazon 环境，请参阅 S Amazon ecurity Pillar Well-Architected Framework 中的基础设施保护。

您可以使用 Amazon 已发布的 API 调用 Amazon Serverless Application Repository 通过网络进行访问。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

Amazon Serverless Application Repository 使用接口端点进行访问 (Amazon PrivateLink)

您可以使用 Amazon PrivateLink 在您的 VPC 和之间创建私有连接 Amazon Serverless Application Repository。您可以像在 VPC 中 Amazon Serverless Application Repository 一样进行访问，无需使用互联网网关、NAT 设备、VPN 连接或 Amazon Direct Connect 连接。VPC 中的实例不需要公有 IP 地址即可访问 Amazon Serverless Application Repository。

您可以通过创建由 Amazon PrivateLink提供支持的接口端点来建立此私有连接。我们将在您为接口端点启用的每个子网中创建一个端点网络接口。这些是请求者托管的网络接口，用作发往 Amazon Serverless Application Repository的流量的入口点。

有关更多信息，请参阅《Amazon PrivateLink 指南》中的通过 Amazon PrivateLink访问 Amazon Web Services 服务。

的注意事项 Amazon Serverless Application Repository

在为设置接口终端节点之前 Amazon Serverless Application Repository，请查看Amazon PrivateLink 指南中的注意事项。

Amazon Serverless Application Repository 支持通过接口端点调用其所有 API 操作。

为创建接口终端节点 Amazon Serverless Application Repository

您可以创建用于 Amazon Serverless Application Repository 使用 Amazon VPC 控制台或 Amazon Command Line Interface (Amazon CLI) 的接口终端节点。有关更多信息，请参阅《Amazon PrivateLink 指南》中的创建接口端点。

Amazon Serverless Application Repository 使用以下服务名称创建接口终端节点：

```
com.amazonaws.region.serverlessrepo
```

如果为接口端点启用私有 DNS，则可使用其默认区域 DNS 名称向 Amazon Serverless Application Repository 发出 API 请求。例如 `serverlessrepo.us-east-1.amazonaws.com`。

为 VPC 端点创建端点策略

端点策略是一种 IAM 资源，您可以将其附加到接口端点。默认终端节点策略允许 Amazon Serverless Application Repository 通过接口终端节点进行完全访问。要控制允许 Amazon Serverless Application Repository 从您的 VPC 访问权限，请将自定义终端节点策略附加到接口终端节点。

端点策略指定以下信息：

- 可执行操作的主体（Amazon Web Services 账户、IAM 用户和 IAM 角色）。
- 可执行的操作。
- 可对其执行操作的资源。

有关更多信息，请参阅《Amazon PrivateLink 指南》中的使用端点策略控制对服务的访问权限。

示例：用于 Amazon Serverless Application Repository 操作的 VPC 终端节点策略

以下是自定义端点策略的示例。当您将此策略附加到接口终端节点时，它会向所有资源的所有委托人授予对所列 Amazon Serverless Application Repository 操作的访问权限。以下示例允许所有用户通过 VPC 终端节点创建应用程序。

```
{
   "Statement": [
      {
         "Principal": "*",
```

```
         "Effect": "Allow",
         "Action": [
            "serverlessrepo:CreateApplication"
         ],
         "Resource":"*"
      }
   ]
}
```

Amazon Serverless Application Repository 配额

Amazon Serverless Application Repository 有一个 Amazon 账户在每个 Amazon 地区可以拥有的公共应用程序数量的配额。此配额适用于每个区域，并且可以增加。要请求提高限制，请使用支持中心控制台。

资源	默认配额
公共应用程序（每个 Amazon 区域的每个 Amazon 账户）	100

以下配额应用于可供代码包和应用程序策略使用的存储。您不能更改这些配额。

资源	限额
代码包的免费 Amazon S3 存储空间（每个 Amazon 区域按 Amazon 账户计算）	5 GB
应用程序策略长度	6,144 个字符

疑难解答 Amazon Serverless Application Repository

使用时 Amazon Serverless Application Repository，在创建、更新或删除应用程序时可能会遇到问题。使用此部分可帮助解决您可能遇到的常见问题。您还可以在 Amazon Serverless Application Repository 论坛上搜索答案和发布问题。

> **Note**
>
> 中的应用程序通过使用进行部署 Amazon CloudFormation。 Amazon Serverless Application Repository 有关疑难解答 Amazon CloudFormation 问题的信息，请参阅《Amazon CloudFormation 故障排除指南》。

主题

您无法使应用程序成为公有

如果您无法使应用程序成为公有，则可能是缺少由开源代码促进会 (OSI) 批准的应用程序的许可证文件。

为使应用程序成为公有，您需要一个 OSI 批准的许可证文件，还有一个成功发布的应用程序版本，以及该版本的源代码 URL。在应用程序创建后，您不能更新应用程序的许可证。

如果由于缺少许可证文件而无法使应用程序成为公有，请删除该应用程序并创建一个新的同名应用程序。确保您为其提供了由开源代码促进会 (OSI) 组织批准的一个或多个开源许可证。

已超过配额

如果您收到指示超出配额的错误消息，请检查您是否达到了资源配额。有关 Amazon Serverless Application Repository 配额，请参阅Amazon Serverless Application Repository 配额。

已更新的自述文件没有立即显示

当您使应用程序成为公有时，应用程序的内容可能需要 24 小时才能更新。如果您遇到超过 24 小时的延迟，请尝试联系 Supp Amazon ort 寻求帮助。有关详细信息，请参阅以下内容。

由于 IAM 权限不足，您无法部署应用程序

要部署 Amazon Serverless Application Repository 应用程序，您需要访问 Amazon Serverless Application Repository 资源和 Amazon CloudFormation 堆栈的权限。您可能还需要权限才能使用应用程序中描述的基础服务。例如，如果您要创建亚马逊 S3 存储桶或亚马逊 DynamoDB 表，则需要访问亚马逊 S3 或 DynamoDB 的权限。

如果您遇到此类问题，请查看您的 Amazon Identity and Access Management (IAM) 策略并确认您拥有必要的权限。有关更多信息，请参阅使用 Identity and Acces s Managem Amazon ent 控制访问权限。

您无法将同一应用程序部署两次

您提供的应用程序名称将用作 Amazon CloudFormation 堆栈的名称。如果您在部署应用程序时遇到问题，请确保没有同名的现有 Amazon CloudFormation 堆栈。如果您这样做，请提供不同的应用程序名称或删除现有堆栈以部署同名的应用程序。

为何我的应用程序不能公开使用

默认情况下，应用程序是私有的。要使应用程序成为公有，请遵循此处的步骤。

联系 支持

有些情况下，您可能无法在本部分中或通过 Amazon Serverless Application Repository 论坛找到故障排除解决方案。如果您有 Amazon Premium Support，则可以在支持部门创建技术Amazon 支持案例。

在联系 Su Amazon pport 之前，请务必获取您有疑问的应用程序的 Amazon 资源名称 (ARN)。您可以在 Amazon Serverless Application Repository 控制台中找到应用程序 ARN。

操作

REST API 包括以下操作

- CreateApplication

  创建应用程序，可以选择包含一个 Amazon SAM 文件，以便在同一次调用中创建第一个应用程序版本。
- CreateApplicationVersion

  创建应用程序版本。
- CreateCloudFormationChangeSet

  为给定的应用程序创建 Amazon CloudFormation 更改集。
- CreateCloudFormationTemplate

  创建 Amazon CloudFormation 模板。
- DeleteApplication

  删除指定的应用程序。
- GetApplication

  获取指定的应用程序。
- GetApplicationPolicy

  检索应用程序的策略。
- GetCloudFormationTemplate

  获取指定的 Amazon CloudFormation 模板。
- ListApplicationDependencies

  检索嵌套在包含应用程序中的应用程序列表。
- ListApplications

  列出请求者所拥有的应用程序。
- ListApplicationVersions

  列出指定应用程序的版本。

- PutApplicationPolicy

  设置应用程序的权限策略。有关此操作支持的操作列表，请参阅应用程序权限。

- UnshareApplication

  取消与 Amazon 组织共享的应用程序。

  此操作只能从组织的主账户调用。

- UpdateApplication

  更新指定的应用程序。

资源

REST API 包括以下资源。

主题

Applications

URI

`/applications`

HTTP 方法

GET

操作 ID：`ListApplications`

列出请求者所拥有的应用程序。

查询参数

Name	Type	必需	描述
`maxItems`	字符串	False	要退货的商品总数。

Name	Type	必需	描述
nextToken	字符串	False	指定从何处开始分页的令牌。

响应

状态代码	响应模型	说明
200	ApplicationPage	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

POST

操作 ID：CreateApplication

创建应用程序，可以选择包含一个 Amazon SAM 文件，以便在同一次调用中创建第一个应用程序版本。

响应

状态代码	响应模型	说明
201	Application	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
409	ConflictException	该资源已存在。

状态代码	响应模型	说明
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

POST 架构

```
{
  "name": "string",
  "description": "string",
  "author": "string",
  "spdxLicenseId": "string",
  "licenseBody": "string",
  "licenseUrl": "string",
  "readmeBody": "string",
  "readmeUrl": "string",
  "labels": [
    "string"
  ],
  "homePageUrl": "string",
  "semanticVersion": "string",
  "templateBody": "string",
  "templateUrl": "string",
```

```
  "sourceCodeUrl": "string",
  "sourceCodeArchiveUrl": "string"
}
```

响应正文

ApplicationPage 架构

```
{
  "applications": [
    {
      "applicationId": "string",
      "name": "string",
      "description": "string",
      "author": "string",
      "spdxLicenseId": "string",
      "labels": [
        "string"
      ],
      "creationTime": "string",
      "homePageUrl": "string"
    }
  ],
  "nextToken": "string"
}
```

Application 架构

```
{
  "applicationId": "string",
  "name": "string",
  "description": "string",
  "author": "string",
  "isVerifiedAuthor": boolean,
  "verifiedAuthorUrl": "string",
  "spdxLicenseId": "string",
  "licenseUrl": "string",
  "readmeUrl": "string",
  "labels": [
    "string"
  ],
  "creationTime": "string",
```

```
  "homePageUrl": "string",
  "version": {
    "applicationId": "string",
    "semanticVersion": "string",
    "sourceCodeUrl": "string",
    "sourceCodeArchiveUrl": "string",
    "templateUrl": "string",
    "creationTime": "string",
    "parameterDefinitions": [
      {
        "name": "string",
        "defaultValue": "string",
        "description": "string",
        "type": "string",
        "noEcho": boolean,
        "allowedPattern": "string",
        "constraintDescription": "string",
        "minValue": integer,
        "maxValue": integer,
        "minLength": integer,
        "maxLength": integer,
        "allowedValues": [
          "string"
        ],
        "referencedByResources": [
          "string"
        ]
      }
    ],
    "requiredCapabilities": [
      enum
    ],
    "resourcesSupported": boolean
  }
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ConflictException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

Application

有关应用程序的详细信息。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

name

应用程序的名称。

最小长度 = 1。最大长度 =140

模式："[a-zA-Z0-9\\-]+";

类型：字符串
必需：True

description

关于应用程序的描述。

最小长度 = 1。最大长度 =256

类型：字符串
必需：True

author

发布应用程序的作者的姓名。

最小长度 = 1。最大长度 = 127。

模式 “^ [a-z0-9] ([a-z0-9] |-(? ! -)) * [a-z0-9]) ？ $”;

类型：字符串
必需：True

isVerifiedAuthor

指定此应用程序的作者是否已通过验证。这意味着，作为一个合理而谨慎的服务提供商， Amazon 它已对请求者提供的信息进行了真诚的审查，并确认请求者的身份与所声称的相同。

类型：布尔值
必填项：False

verifiedAuthorUrl

经过验证的作者的公开个人资料网址。此 URL 由作者提交。

类型：字符串
必填项：False

spdxLicenseId

来自 https://spdx.org/licenses/ 的有效标识符。

类型：字符串
必填项：False

licenseUrl

指向应用程序许可证文件的链接，该文件与您的应用程序的 spdxLicenseId 值相匹配。

最大大小 5 MB

类型：字符串
必填项：False

readmeUrl

指向 Markdown 语言的自述文件链接，其中包含对应用程序及其工作原理的更详细描述。

最大大小 5 MB

类型：字符串
必填项：False

labels

用于改善搜索结果中应用程序发现率的标签。

最小长度 = 1。最大长度 = 127。标签的最大数量：10

模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

类型：string 类型的数组
必填项：False

creationTime

此资源的创建日期和时间。

类型：字符串
必填项：False

homePageUrl

包含有关应用程序的更多信息（例如应用程序 GitHub 存储库的位置）的 URL。

类型：字符串
必填项：False

version

有关应用程序的版本信息。

类型：版本
必需：False

ApplicationPage

应用程序详细信息列表。

applications

一系列应用程序摘要。

类型：ApplicationSummary 类型的数组
必需：True

nextToken

用于请求下一页结果的令牌。

类型：字符串
必填项：False

ApplicationSummary

有关应用程序的详细信息摘要。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

name

应用程序的名称。

最小长度 = 1。最大长度 =140

模式："[a-zA-Z0-9\\-]+";

类型：字符串
必需：True

description

关于应用程序的描述。

最小长度 = 1。最大长度 =256

类型：字符串
必需：True

author

发布应用程序的作者的姓名。

最小长度 = 1。最大长度 = 127。

模式 “^ [a-z0-9] ([a-z0-9] |-(? ! -)) * [a-z0-9]) ? $”;

类型：字符串
必需：True

spdxLicenseId

来自 https://spdx.org/licenses/ 的有效标识符。

类型：字符串
必填项：False

labels

用于改善搜索结果中应用程序发现率的标签。

最小长度 = 1。最大长度 = 127。标签的最大数量：10

模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

类型：string 类型的数组
必填项：False

creationTime

此资源的创建日期和时间。

类型：字符串
必填项：False

homePageUrl

包含有关应用程序的更多信息（例如应用程序 GitHub 存储库的位置）的 URL。

类型：字符串

必填项：False

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串

必填项：False

errorCode

400

类型：字符串

必填项：False

Capability

部署某些应用程序时必须指定的值。

```
CAPABILITY_IAM
CAPABILITY_NAMED_IAM
CAPABILITY_AUTO_EXPAND
CAPABILITY_RESOURCE_POLICY
```

ConflictException

该资源已存在。

message

该资源已存在。

类型：字符串
必填项：False

errorCode

409

类型：字符串
必填项：False

CreateApplicationInput

创建应用程序请求。

name

您要发布的应用程序的名称。

最小长度 = 1。最大长度 =140

模式："[a-zA-Z0-9\\-]+";

类型：字符串
必需：True

description

关于应用程序的描述。

最小长度 = 1。最大长度 =256

类型：字符串
必需：True

author

发布应用程序的作者的姓名。

最小长度 = 1。最大长度 = 127。

模式 “^ [a-z0-9] ([a-z0-9] |-(? ! -)) * [a-z0-9]) ? $”;

类型：字符串
必需：True

spdxLicenseId

来自 https://spdx.org/licenses/ 的有效标识符。

类型：字符串
必填项：False

licenseBody

一个本地文本文件，其中包含与您的应用程序的 spdxLicenseId 值相匹配的应用程序许可证。该文件的格式为`file://<path>/<filename>`。

最大大小 5 MB

您只能指定`licenseBody`和中的一个`licenseUrl`；否则，将出现错误。

类型：字符串
必填项：False

licenseUrl

指向 S3 对象的链接，其中包含与您的应用程序的 spdxLicenseID 值相匹配的应用程序许可证。

最大大小 5 MB

您只能指定`licenseBody`和中的一个`licenseUrl`；否则，将出现错误。

类型：字符串
必填项：False

readmeBody

Markdown 语言的本地文本自述文件，其中包含对应用程序及其工作原理的更详细描述。该文件的格式为`file://<path>/<filename>`。

最大大小 5 MB

您只能指定readmeBody和中的一个readmeUrl；否则，将出现错误。

类型：字符串
必填项：False

readmeUrl

指向 Markdown 语言中的 S3 对象的链接，其中包含对应用程序及其工作原理的更详细描述。

最大大小 5 MB

您只能指定readmeBody和中的一个readmeUrl；否则，将出现错误。

类型：字符串
必填项：False

labels

用于改善搜索结果中应用程序发现率的标签。

最小长度 = 1。最大长度 = 127。标签的最大数量：10

模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

类型：string 类型的数组
必填项：False

homePageUrl

包含有关应用程序的更多信息（例如应用程序 GitHub 存储库的位置）的 URL。

类型：字符串
必填项：False

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必填项：False

templateBody

应用程序的本地原始打包 Amazon SAM 模板文件。该文件的格式为`file://<path>/<filename>`。

只能指定`templateBody`和中的一个`templateUrl`；否则会出现错误。

类型：字符串
必填项：False

templateUrl

指向包含应用程序打包 Amazon SAM 模板的 S3 对象的链接。

只能指定`templateBody`和中的一个`templateUrl`；否则会出现错误。

类型：字符串
必填项：False

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

sourceCodeArchiveUrl

指向 S3 对象的链接，其中包含此版本应用程序的源代码的 ZIP 存档。

最大大小 50 MB

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

ParameterDefinition

应用程序支持的参数。

name

参数的名称。

类型：字符串
必需：True

defaultValue

模板适当类型的值，用于在创建堆栈时未指定值的情况下。如果您定义参数的约束，则必须指定一个符合这些约束的值。

类型：字符串
必填项：False

description

描述参数的字符串，最多 4,000 个字符。

类型：字符串
必填项：False

type

参数的类型。

有效值：`String | Number | List<Number> | CommaDelimitedList`

`String`: 一个字面字符串。

例如，用户可以指定`"MyUserName"`。

`Number`: 整数或浮点数。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变为字符串。

例如，用户可以指定`"8888"`。

`List<Number>`: 用逗号分隔的整数或浮点数组。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变成字符串列表。

例如，用户可以指定 “80,20”，然后`Ref`得出结果。`["80","20"]`

`CommaDelimitedList`：用逗号分隔的文字字符串数组。字符串的总数应比逗号总数多 1。此外，每个成员字符串都经过空格修剪。

例如，用户可以指定 “测试、开发、生产”，然后输入`Ref`结果。`["test","dev","prod"]`

类型：字符串
必填项：False

noEcho

每当有人调用描述堆栈时，是否要屏蔽参数值。如果将该值设置为 true，则使用星号 (*****) 掩盖参数值。

类型：布尔值
必填项：False

allowedPattern

一个正则表达式，表示要允许 `String` 类型使用的模式。

类型：字符串
必填项：False

constraintDescription

用于在违反约束时说明该约束的字符串。例如，在没有约束条件描述的情况下，具有允许的 [A-Za-z0-9]+ 模式的参数会在用户指定无效值时显示以下错误消息：

```
Malformed input-Parameter MyParameter must match pattern [A-Za-z0-9]+
```

通过添加约束条件描述（例如“必须仅包含大写和小写字母和数字”），可以显示以下自定义错误消息：

```
Malformed input-Parameter MyParameter must contain only uppercase and
lowercase letters and numbers.
```

类型：字符串
必填项：False

minValue

一个数值，用于确定要允许Number类型使用的最小数值。

类型：整数
必需：False

maxValue

一个数值，用于确定要允许Number类型使用的最大数值。

类型：整数
必需：False

minLength

一个整数值，用于确定要允许String类型使用的最小字符数。

类型：整数
必需：False

maxLength

一个整数值，用于确定要允许的String类型的最大字符数。

类型：整数
必需：False

allowedValues

包含参数允许值列表的阵列。

类型：string 类型的数组
必填项：False

referencedByResources

使用此参数的 Amazon SAM 资源列表。

类型：string 类型的数组
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

Version

应用程序版本详情。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

sourceCodeArchiveUrl

指向 S3 对象的链接，其中包含此版本应用程序的源代码的 ZIP 存档。

最大大小 50 MB

类型：字符串
必填项：False

templateUrl

指向应用程序打包 Amazon SAM 模板的链接。

类型：字符串
必需：True

creationTime

此资源的创建日期和时间。

类型：字符串
必需：True

parameterDefinitions

应用程序支持的参数类型数组。

类型：ParameterDefinition 类型的数组
必需：True

requiredCapabilities

在部署某些应用程序之前必须指定的值列表。某些应用程序可能包含可能影响您 Amazon 账户权限的资源，例如，通过创建新 Amazon Identity and Access Management (IAM) 用户。对于这些应用程序，必须通过指定此参数来明确确认其功能。

唯一有效的值是CAPABILITY_IAM、CAPABILITY_NAMED_IAMCAPABILITY_RESOURCE_POLICY、和CAPABILITY_AUTO_EXPAND。

以下资源需要您指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM：AWS::IAM::Group、AWS::IAM::InstanceProfileAWS::IAM和AWS::IAM::Role。如果应用程序包含 IAM 资源，则可以指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM。如果应用程序包含具有自定义名称的 IAM 资源，您必须指定 CAPABILITY_NAMED_IAM。

以下资源需要您指定CAPABILITY_RESOURCE_POLICY：AWS::Lambda::Permission、AWS:: iam:Policy、、AWS::ApplicationAutoScaling::ScalingPolicy、AWS::S3::BucketPolicy和。AWS::SQS::QueuePolic

包含一个或多个嵌套应用程序的应用程序要求您指定 CAPABILITY_AUTO_EXPAND。

如果您的应用程序模板包含上述任何资源，我们建议您在部署之前查看与该应用程序关联的所有权限。如果您没有为需要功能的应用程序指定此参数，则调用将失败。

类型：Capability 类型的数组

必需：True

resourcesSupported

检索该应用程序所在的区域是否支持此应用程序中包含的所有 Amazon 资源。

类型：布尔值

必需：True

应用程序应用程序 ID

URI

`/applications/applicationId`

HTTP 方法

GET

操作 ID：`GetApplication`

获取指定的应用程序。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

查询参数

Name	Type	必需	描述
semanticVersion	字符串	False	要获取的应用程序的语义版本。

响应

状态代码	响应模型	说明
200	Application	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

DELETE

操作 ID：DeleteApplication

删除指定的应用程序。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
204	无	成功
400	BadRequestException	请求中的参数之一无效。

状态代码	响应模型	说明
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
409	ConflictException	该资源已存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

PATCH

操作 ID：UpdateApplication

更新指定的应用程序。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	Application	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
409	ConflictException	该资源已存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

架构

请求正文

PATCH 架构

```
{
  "description": "string",
  "author": "string",
  "readmeBody": "string",
  "readmeUrl": "string",
```

```
  "labels": [
    "string"
  ],
  "homePageUrl": "string"
}
```

响应正文

Application 架构

```
{
  "applicationId": "string",
  "name": "string",
  "description": "string",
  "author": "string",
  "isVerifiedAuthor": boolean,
  "verifiedAuthorUrl": "string",
  "spdxLicenseId": "string",
  "licenseUrl": "string",
  "readmeUrl": "string",
  "labels": [
    "string"
  ],
  "creationTime": "string",
  "homePageUrl": "string",
  "version": {
    "applicationId": "string",
    "semanticVersion": "string",
    "sourceCodeUrl": "string",
    "sourceCodeArchiveUrl": "string",
    "templateUrl": "string",
    "creationTime": "string",
    "parameterDefinitions": [
      {
        "name": "string",
        "defaultValue": "string",
        "description": "string",
        "type": "string",
        "noEcho": boolean,
        "allowedPattern": "string",
        "constraintDescription": "string",
        "minValue": integer,
        "maxValue": integer,
```

```
        "minLength": integer,
        "maxLength": integer,
        "allowedValues": [
          "string"
        ],
        "referencedByResources": [
          "string"
        ]
      }
    ],
    "requiredCapabilities": [
      enum
    ],
    "resourcesSupported": boolean
  }
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ConflictException 架构

```
{
```

```
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

Application

有关应用程序的详细信息。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

name

应用程序的名称。

最小长度 = 1。最大长度 =140

模式："[a-zA-Z0-9\\-]+";

类型：字符串
必需：True

description

关于应用程序的描述。

最小长度 = 1。最大长度 =256

类型：字符串
必需：True

author

发布应用程序的作者的姓名。

最小长度 = 1。最大长度 = 127。

模式 “^ [a-z0-9] ([a-z0-9] |-(? ! -)) * [a-z0-9]) ? $”;

类型：字符串
必需：True

isVerifiedAuthor

指定此应用程序的作者是否已通过验证。这意味着，作为一个合理而谨慎的服务提供商， Amazon 它已对请求者提供的信息进行了真诚的审查，并确认请求者的身份与所声称的相同。

类型：布尔值
必填项：False

verifiedAuthorUrl

经过验证的作者的公开个人资料网址。此 URL 由作者提交。

类型：字符串
必填项：False

spdxLicenseId

来自 https://spdx.org/licenses/ 的有效标识符。

类型：字符串
必填项：False

licenseUrl

指向应用程序许可证文件的链接，该文件与您的应用程序的 spdxLicenseId 值相匹配。

最大大小 5 MB

类型：字符串
必填项：False

readmeUrl

指向 Markdown 语言的自述文件链接，其中包含对应用程序及其工作原理的更详细描述。

最大大小 5 MB

类型：字符串
必填项：False

labels

用于改善搜索结果中应用程序发现率的标签。

最小长度 = 1。最大长度 = 127。标签的最大数量：10

模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

类型：string 类型的数组
必填项：False

creationTime

此资源的创建日期和时间。

类型：字符串
必填项：False

homePageUrl

包含有关应用程序的更多信息（例如应用程序 GitHub 存储库的位置）的 URL。

类型：字符串

必填项：False

version

有关应用程序的版本信息。

类型：版本
必需：False

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

Capability

部署某些应用程序时必须指定的值。

```
CAPABILITY_IAM
CAPABILITY_NAMED_IAM
CAPABILITY_AUTO_EXPAND
CAPABILITY_RESOURCE_POLICY
```

ConflictException

该资源已存在。

message

该资源已存在。

类型：字符串
必填项：False

errorCode

409

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

ParameterDefinition

应用程序支持的参数。

name

参数的名称。

类型：字符串
必需：True

defaultValue

模板适当类型的值，用于在创建堆栈时未指定值的情况下。如果您定义参数的约束，则必须指定一个符合这些约束的值。

类型：字符串
必填项：False

description

描述参数的字符串，最多 4,000 个字符。

类型：字符串
必填项：False

type

参数的类型。

有效值：`String | Number | List<Number> | CommaDelimitedList`

`String`: 一个字面字符串。

例如，用户可以指定`"MyUserName"`。

`Number`: 整数或浮点数。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变为字符串。

例如，用户可以指定`"8888"`。

`List<Number>`: 用逗号分隔的整数或浮点数组。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变成字符串列表。

例如，用户可以指定 “80,20”，然后`Ref`得出结果。`["80","20"]`

`CommaDelimitedList`：用逗号分隔的文字字符串数组。字符串的总数应比逗号总数多 1。此外，每个成员字符串都经过空格修剪。

例如，用户可以指定 “测试、开发、生产”，然后输入`Ref`结果。`["test","dev","prod"]`

类型：字符串
必填项：False

noEcho

每当有人调用描述堆栈时，是否要屏蔽参数值。如果将该值设置为 true，则使用星号 (*****) 掩盖参数值。

类型：布尔值
必填项：False

allowedPattern

一个正则表达式，表示要允许 String 类型使用的模式。

类型：字符串
必填项：False

constraintDescription

用于在违反约束时说明该约束的字符串。例如，在没有约束条件描述的情况下，具有允许的 [A-Za-z0-9]+ 模式的参数会在用户指定无效值时显示以下错误消息：

```
Malformed input-Parameter MyParameter must match pattern [A-Za-z0-9]+
```

通过添加约束条件描述（例如 “必须仅包含大写和小写字母和数字”），可以显示以下自定义错误消息：

```
Malformed input-Parameter MyParameter must contain only uppercase and
lowercase letters and numbers.
```

类型：字符串
必填项：False

minValue

一个数值，用于确定要允许Number类型使用的最小数值。

类型：整数
必需：False

maxValue

一个数值，用于确定要允许Number类型使用的最大数值。

类型：整数
必需：False

minLength

一个整数值，用于确定要允许String类型使用的最小字符数。

类型：整数
必需：False

maxLength

一个整数值，用于确定要允许的String类型的最大字符数。

类型：整数
必需：False

allowedValues

包含参数允许值列表的阵列。

类型：string 类型的数组
必填项：False

referencedByResources

使用此参数的 Amazon SAM 资源列表。

类型：string 类型的数组
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

UpdateApplicationInput

更新应用程序请求。

description

关于应用程序的描述。

最小长度 = 1。最大长度 =256

类型：字符串
必填项：False

author

发布应用程序的作者的姓名。

最小长度 = 1。最大长度 = 127。

模式 “^ [a-z0-9] ([a-z0-9] |-(? ! -)) * [a-z0-9]) ? $”;

类型：字符串
必填项：False

readmeBody

Markdown 语言的文本自述文件，其中包含对应用程序及其工作原理的更详细描述。

最大大小 5 MB

类型：字符串
必填项：False

readmeUrl

指向 Markdown 语言的自述文件链接，其中包含对应用程序及其工作原理的更详细描述。

最大大小 5 MB

类型：字符串
必填项：False

labels

用于改善搜索结果中应用程序发现率的标签。

最小长度 = 1。最大长度 = 127。标签的最大数量：10

模式："^[a-zA-Z0-9+\\-_:\\/@]+$";

类型：string 类型的数组
必填项：False

homePageUrl

包含有关应用程序的更多信息（例如应用程序 GitHub 存储库的位置）的 URL。

类型：字符串
必填项：False

Version

应用程序版本详情。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串

必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

sourceCodeArchiveUrl

指向 S3 对象的链接，其中包含此版本应用程序的源代码的 ZIP 存档。

最大大小 50 MB

类型：字符串
必填项：False

templateUrl

指向应用程序打包 Amazon SAM 模板的链接。

类型：字符串
必需：True

creationTime

此资源的创建日期和时间。

类型：字符串

必需：True

parameterDefinitions

应用程序支持的参数类型数组。

类型：ParameterDefinition 类型的数组
必需：True

requiredCapabilities

在部署某些应用程序之前必须指定的值列表。某些应用程序可能包含可能影响您 Amazon 账户权限的资源，例如，通过创建新 Amazon Identity and Access Management (IAM) 用户。对于这些应用程序，必须通过指定此参数来明确确认其功能。

唯一有效的值是CAPABILITY_IAM、CAPABILITY_NAMED_IAMCAPABILITY_RESOURCE_POLICY、和CAPABILITY_AUTO_EXPAND。

以下资源需要您指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM：AWS::IAM::Group、AWS::IAM::InstanceProfileAWS::IAM和AWS::IAM::Role。如果应用程序包含 IAM 资源，则可以指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM。如果应用程序包含具有自定义名称的 IAM 资源，您必须指定 CAPABILITY_NAMED_IAM。

以下资源需要您指定CAPABILITY_RESOURCE_POLICY：AWS::Lambda::Permission、AWS:: iam:Policy、、AWS::ApplicationAutoScaling::ScalingPolicy、AWS::S3::BucketPolicy和。AWS::SQS::QueuePolic

包含一个或多个嵌套应用程序的应用程序要求您指定 CAPABILITY_AUTO_EXPAND。

如果您的应用程序模板包含上述任何资源，我们建议您在部署之前查看与该应用程序关联的所有权限。如果您没有为需要功能的应用程序指定此参数，则调用将失败。

类型：Capability 类型的数组
必需：True

resourcesSupported

检索该应用程序所在的区域是否支持此应用程序中包含的所有 Amazon 资源。

类型：布尔值

必需：True

应用程序应用程序 ID 变更集

URI

`/applications/applicationId/changesets`

HTTP 方法

POST

操作 ID：`CreateCloudFormationChangeSet`

为给定的应用程序创建 Amazon CloudFormation 更改集。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
201	ChangeSetDetails	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

POST 架构

```
{
  "stackName": "string",
  "semanticVersion": "string",
  "templateId": "string",
  "parameterOverrides": [
    {
      "name": "string",
      "value": "string"
    }
  ],
  "capabilities": [
    "string"
  ],
  "changeSetName": "string",
  "clientToken": "string",
  "description": "string",
  "notificationArns": [
    "string"
  ],
```

```
  "resourceTypes": [
    "string"
  ],
  "rollbackConfiguration": {
    "rollbackTriggers": [
      {
        "arn": "string",
        "type": "string"
      }
    ],
    "monitoringTimeInMinutes": integer
  },
  "tags": [
    {
      "key": "string",
      "value": "string"
    }
  ]
}
```

响应正文

ChangeSetDetails 架构

```
{
  "applicationId": "string",
  "semanticVersion": "string",
  "changeSetId": "string",
  "stackId": "string"
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
```

```
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ChangeSetDetails

变更集的详细信息。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

changeSetId

变更集的亚马逊资源名称 (ARN)。

长度限制：最小长度为 1。

模式：ARN：[-a-za-z0-9：/] *

类型：字符串
必需：True

stackId

堆栈的唯一 ID。

类型：字符串
必需：True

CreateCloudFormationChangeSetInput

创建应用程序更改集请求。

stackName

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必填项：False

templateId

返回的 UUID。 CreateCloudFormationTemplate

模式：[0-9a-fa-f] {8}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {12}

类型：字符串
必填项：False

parameterOverrides

应用程序参数的参数值列表。

类型：ParameterValue 类型的数组
必填项：False

capabilities

在部署某些应用程序之前必须指定的值列表。某些应用程序可能包含可能影响您 Amazon 账户权限的资源，例如，通过创建新 Amazon Identity and Access Management (IAM) 用户。对于这些应用程序，必须通过指定此参数来明确确认其功能。

唯一有效的值是CAPABILITY_IAM、CAPABILITY_NAMED_IAMCAPABILITY_RESOURCE_POLICY、和CAPABILITY_AUTO_EXPAND。

以下资源需要您指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM：AWS::IAM::Group、AWS::IAM::InstanceProfileAWS::IAM和AWS::IAM::Role。如果应用程序包含 IAM 资源，则可以指

定CAPABILITY_IAM或CAPABILITY_NAMED_IAM。如果应用程序包含具有自定义名称的 IAM 资源，您必须指定 CAPABILITY_NAMED_IAM。

以下资源要求您指定CAPABILITY_RESOURCE_POLICY:AWS::Lambda::Permission、AWS:: iam: Policy、、AWS::ApplicationAutoScaling::ScalingPolicyAWS::S3::BucketPolicy、AWS::SQS::QueuePolicy和AWS:: SN S:。TopicPolicy

包含一个或多个嵌套应用程序的应用程序要求您指定 CAPABILITY_AUTO_EXPAND。

如果您的应用程序模板包含上述任何资源，我们建议您在部署之前查看与该应用程序关联的所有权限。如果您没有为需要功能的应用程序指定此参数，则调用将失败。

类型：string 类型的数组
必填项：False

changeSetName

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：字符串
必填项：False

clientToken

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：字符串
必填项：False

description

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：字符串
必填项：False

notificationArns

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：string 类型的数组
必填项：False

resourceTypes

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：string 类型的数组
必填项：False

rollbackConfiguration

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：RollbackConfiguration
必需：False

tags

此属性与 Amazon CloudFormation CreateChangeSetAPI 的同名参数相对应。

类型：Tag 类型的数组
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

ParameterValue

应用程序的参数值。

name

与参数关联的键。如果您没有为特定参数指定键和值，则 Amazon CloudFormation 使用模板中指定的默认值。

类型：字符串
必需：True

value

与参数关联的输入值。

类型：字符串
必需：True

RollbackConfiguration

此属性对应于Amazon CloudFormation RollbackConfiguration数据类型。

rollbackTriggers

此属性对应于Amazon CloudFormation RollbackConfiguration数据类型的同名内容。

类型：RollbackTrigger 类型的数组
必填项：False

monitoringTimeInMinutes

此属性对应于Amazon CloudFormation RollbackConfiguration数据类型的同名内容。

类型：整数
必需：False

RollbackTrigger

此属性对应于Amazon CloudFormation RollbackTrigger数据类型。

arn

此属性对应于Amazon CloudFormation RollbackTrigger数据类型的同名内容。

类型：字符串
必需：True

type

此属性对应于Amazon CloudFormation RollbackTrigger数据类型的同名内容。

类型：字符串
必需：True

Tag

此属性对应于Amazon CloudFormation 标签数据类型。

key

此属性对应于Amazon CloudFormation 标签数据类型的同名内容。

类型：字符串
必需：True

value

此属性对应于Amazon CloudFormation 标签数据类型的同名内容。

类型：字符串
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

Applications applicationId Dependencies

URI

`/applications/applicationId/dependencies`

HTTP 方法

GET

操作 ID：`ListApplicationDependencies`

检索嵌套在包含应用程序中的应用程序列表。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

查询参数

Name	Type	必需	描述
nextToken	字符串	False	指定从何处开始分页的令牌。
maxItems	字符串	False	要退货的商品总数。
semanticVersion	字符串	False	要获取的应用程序的语义版本。

响应

状态代码	响应模型	说明
200	ApplicationDependencyPage	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。

状态代码	响应模型	说明
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

响应正文

ApplicationDependencyPage 架构

```
{
  "dependencies": [
    {
      "applicationId": "string",
      "semanticVersion": "string"
    }
  ],
  "nextToken": "string"
```

```
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

ApplicationDependencyPage

嵌套在应用程序中的应用程序摘要列表。

dependencies

嵌套在应用程序中的应用程序摘要数组。

类型：ApplicationDependencySummary 类型的数组
必需：True

nextToken

用于请求下一页结果的令牌。

类型：字符串
必填项：False

ApplicationDependencySummary

嵌套应用程序摘要。

applicationId

嵌套应用程序的亚马逊资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

嵌套应用程序的语义版本。

类型：字符串
必需：True

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

应用程序应用程序 ID 政策

URI

`/applications/applicationId/policy`

HTTP 方法

GET

操作 ID：`GetApplicationPolicy`

检索应用程序的策略。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	ApplicationPolicy	成功
400	BadRequestException	请求中的参数之一无效。

状态代码	响应模型	说明
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

PUT

操作 ID：PutApplicationPolicy

设置应用程序的权限策略。有关此操作支持的操作列表，请参阅应用程序权限。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	ApplicationPolicy	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。

状态代码	响应模型	说明
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

PUT 架构

```
{
  "statements": [
    {
      "statementId": "string",
      "principals": [
        "string"
      ],
      "actions": [
```

```
        "string"
      ],
      "principalOrgIDs": [
        "string"
      ]
    }
  ]
}
```

响应正文

ApplicationPolicy 架构

```
{
  "statements": [
    {
      "statementId": "string",
      "principals": [
        "string"
      ],
      "actions": [
        "string"
      ],
      "principalOrgIDs": [
        "string"
      ]
    }
  ]
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
```

```
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

ApplicationPolicy

应用于应用程序的政策声明。

statements

应用于应用程序的一系列策略声明。

类型：ApplicationPolicyStatement 类型的数组
必需：True

ApplicationPolicyStatement

应用于应用程序的政策声明。

statementId

语句的唯一 ID。

类型：字符串
必填项：False

principals

IDs 要与之共享应用程序的 Amazon 帐户数组，或* 用于公开该应用程序。

类型：string 类型的数组
必需：True

actions

有关此操作支持的操作列表，请参阅应用程序权限。

类型：string 类型的数组
必需：True

principalOrgIDs

要与之共享应用程序的 Amazon Organizations ID。

类型：string 类型的数组
必填项：False

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

Applications applicationId Templates

URI

`/applications/applicationId/templates`

HTTP 方法

POST

操作 ID：`CreateCloudFormationTemplate`

创建 Amazon CloudFormation 模板。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
201	TemplateDetails	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

POST 架构

```
{
  "semanticVersion": "string"
}
```

响应正文

TemplateDetails 架构

```
{
  "templateId": "string",
  "templateUrl": "string",
  "applicationId": "string",
  "semanticVersion": "string",
  "status": enum,
  "creationTime": "string",
  "expirationTime": "string"
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

CreateCloudFormationTemplateInput

创建模板请求。

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

TemplateDetails

模板的详细信息。

templateId

返回的 UUID。 CreateCloudFormationTemplate

模式：[0-9a-fa-f] {8}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {12}

类型：字符串
必需：True

templateUrl

指向模板的链接，可用于使用部署应用程序 Amazon CloudFormation。

类型：字符串
必需：True

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

status

模板创建工作流程的状态。

可能的值：PREPARING | ACTIVE | EXPIRED

类型：字符串
必需：True
价值观：PREPARING | ACTIVE | EXPIRED

creationTime

此资源的创建日期和时间。

类型：字符串
必需：True

expirationTime

此模板的到期日期和时间。模板在创建 1 小时后过期。

类型：字符串
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

Applications applicationId Templates templateId

URI

`/applications/applicationId/templates/templateId`

HTTP 方法

GET

操作 ID：`GetCloudFormationTemplate`

获取指定的 Amazon CloudFormation 模板。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。
templateId	字符串	True	返回的 UUID。CreateCloudFormationTemplate 模式：[0-9a-fa-f] {8}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {12}

响应

状态代码	响应模型	说明
200	TemplateDetails	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。

状态代码	响应模型	说明
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。
templateId	字符串	True	返回的 UUID。CreateCloudFormationTemplate 模式：[0-9a-fa-f] {8}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {12}

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

响应正文

TemplateDetails 架构

```
{
  "templateId": "string",
  "templateUrl": "string",
  "applicationId": "string",
  "semanticVersion": "string",
  "status": enum,
  "creationTime": "string",
  "expirationTime": "string"
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
```

```
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串

必填项：False

TemplateDetails

模板的详细信息。

templateId

返回的 UUID。 CreateCloudFormationTemplate

模式：[0-9a-fa-f] {8}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {4}\-[0-9a-fa-f] {12}

类型：字符串

必需：True

templateUrl

指向模板的链接，可用于使用部署应用程序 Amazon CloudFormation。

类型：字符串

必需：True

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串

必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串

必需：True

status

模板创建工作流程的状态。

可能的值：PREPARING | ACTIVE | EXPIRED

类型：字符串
必需：True
价值观：PREPARING | ACTIVE | EXPIRED

creationTime

此资源的创建日期和时间。

类型：字符串
必需：True

expirationTime

此模板的到期日期和时间。模板在创建 1 小时后过期。

类型：字符串
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串

必填项：False

Applications applicationId Unshare

URI

`/applications/applicationId/unshare`

HTTP 方法

POST

操作 ID：UnshareApplication

取消与 Amazon 组织共享的应用程序。

此操作只能从组织的主账户调用。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
204	无	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。

状态代码	响应模型	说明
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

POST 架构

```
{
  "organizationId": "string"
}
```

响应正文

BadRequestException 架构

```
{
  "message": "string",
```

```
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串

必填项：False

UnshareApplicationInput

取消共享应用程序请求。

organizationId

要取消共享应用程序的 Amazon Organizations ID。

类型：字符串
必需：True

应用程序应用程序 ID 版本

URI

`/applications/applicationId/versions`

HTTP 方法

GET

操作 ID：`ListApplicationVersions`

列出指定应用程序的版本。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

查询参数

Name	Type	必需	描述
maxItems	字符串	False	要退货的商品总数。

Name	Type	必需	描述
nextToken	字符串	False	指定从何处开始分页的令牌。

响应

状态代码	响应模型	说明
200	ApplicationVersionPage	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
404	NotFoundException	请求中指定的资源（例如访问策略声明）不存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

响应正文

ApplicationVersionPage 架构

```
{
  "versions": [
    {
      "applicationId": "string",
      "semanticVersion": "string",
      "sourceCodeUrl": "string",
      "creationTime": "string"
    }
  ],
  "nextToken": "string"
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

NotFoundException 架构

```
{
```

```
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

ApplicationVersionPage

应用程序的版本摘要列表。

versions

应用程序的版本摘要数组。

类型：VersionSummary 类型的数组
必需：True

nextToken

用于请求下一页结果的令牌。

类型：字符串
必填项：False

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

NotFoundException

请求中指定的资源（例如访问策略声明）不存在。

message

请求中指定的资源（例如访问策略声明）不存在。

类型：字符串
必填项：False

errorCode

404

类型：字符串
必填项：False

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

VersionSummary

应用程序版本摘要。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

creationTime

此资源的创建日期和时间。

类型：字符串
必需：True

应用程序 applicationID 版本语义版本

URI

`/applications/applicationId/versions/semanticVersion`

HTTP 方法

PUT

操作 ID：`CreateApplicationVersion`

创建应用程序版本。

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。
semanticVersion	字符串	True	新版本的语义版本。

响应

状态代码	响应模型	说明
201	Version	成功
400	BadRequestException	请求中的参数之一无效。
403	ForbiddenException	客户端未通过身份验证。
409	ConflictException	该资源已存在。
429	TooManyRequestsException	客户端每单位时间发送的请求数超过了允许的请求数。
500	InternalServerErrorException	该 Amazon Serverless Application Repository 服务遇到了内部错误。

OPTIONS

路径参数

Name	Type	必需	描述
applicationId	字符串	True	应用程序的 Amazon 资源名称（ARN）。
semanticVersion	字符串	True	新版本的语义版本。

响应

状态代码	响应模型	说明
200	无	200 条回复

架构

请求正文

PUT 架构

```
{
  "templateBody": "string",
  "templateUrl": "string",
  "sourceCodeUrl": "string",
  "sourceCodeArchiveUrl": "string"
}
```

响应正文

Version 架构

```
{
  "applicationId": "string",
  "semanticVersion": "string",
  "sourceCodeUrl": "string",
  "sourceCodeArchiveUrl": "string",
```

```
  "templateUrl": "string",
  "creationTime": "string",
  "parameterDefinitions": [
    {
      "name": "string",
      "defaultValue": "string",
      "description": "string",
      "type": "string",
      "noEcho": boolean,
      "allowedPattern": "string",
      "constraintDescription": "string",
      "minValue": integer,
      "maxValue": integer,
      "minLength": integer,
      "maxLength": integer,
      "allowedValues": [
        "string"
      ],
      "referencedByResources": [
        "string"
      ]
    }
  ],
  "requiredCapabilities": [
    enum
  ],
  "resourcesSupported": boolean
}
```

BadRequestException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ForbiddenException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

ConflictException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

TooManyRequestsException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

InternalServerErrorException 架构

```
{
  "message": "string",
  "errorCode": "string"
}
```

Properties

BadRequestException

请求中的参数之一无效。

message

请求中的参数之一无效。

类型：字符串
必填项：False

errorCode

400

类型：字符串

必填项：False

Capability

部署某些应用程序时必须指定的值。

```
CAPABILITY_IAM
CAPABILITY_NAMED_IAM
CAPABILITY_AUTO_EXPAND
CAPABILITY_RESOURCE_POLICY
```

ConflictException

该资源已存在。

message

该资源已存在。

类型：字符串
必填项：False

errorCode

409

类型：字符串
必填项：False

CreateApplicationVersionInput

创建版本请求。

templateBody

应用程序的原始打包 Amazon SAM 模板。

类型：字符串
必填项：False

templateUrl

指向应用程序打包 Amazon SAM 模板的链接。

类型：字符串
必填项：False

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

sourceCodeArchiveUrl

指向 S3 对象的链接，其中包含此版本应用程序的源代码的 ZIP 存档。

最大大小 50 MB

类型：字符串
必填项：False

ForbiddenException

客户端未通过身份验证。

message

客户端未通过身份验证。

类型：字符串
必填项：False

errorCode

403

类型：字符串
必填项：False

InternalServerErrorException

该 Amazon Serverless Application Repository 服务遇到了内部错误。

message

该 Amazon Serverless Application Repository 服务遇到了内部错误。

类型：字符串
必填项：False

errorCode

500

类型：字符串
必填项：False

ParameterDefinition

应用程序支持的参数。

name

参数的名称。

类型：字符串
必需：True

defaultValue

模板适当类型的值，用于在创建堆栈时未指定值的情况下。如果您定义参数的约束，则必须指定一个符合这些约束的值。

类型：字符串
必填项：False

description

描述参数的字符串，最多 4,000 个字符。

类型：字符串
必填项：False

type

参数的类型。

有效值：`String | Number | List<Number> | CommaDelimitedList`

`String`: 一个字面字符串。

例如，用户可以指定`"MyUserName"`。

`Number`: 整数或浮点数。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变为字符串。

例如，用户可以指定`"8888"`。

`List<Number>`: 用逗号分隔的整数或浮点数组。 Amazon CloudFormation 将参数值验证为数字。但是，当您在模板的其他地方使用该参数时（例如，通过使用`Ref`内部函数），参数值将变成字符串列表。

例如，用户可以指定 “80,20”，然后`Ref`得出结果。`["80","20"]`

`CommaDelimitedList`：用逗号分隔的文字字符串数组。字符串的总数应比逗号总数多 1。此外，每个成员字符串都经过空格修剪。

例如，用户可以指定 “测试、开发、生产”，然后输入`Ref`结果。`["test","dev","prod"]`

类型：字符串
必填项：False

noEcho

每当有人调用描述堆栈时，是否要屏蔽参数值。如果将该值设置为 true，则使用星号 (*****) 掩盖参数值。

类型：布尔值
必填项：False

allowedPattern

一个正则表达式，表示要允许 String 类型使用的模式。

类型：字符串
必填项：False

constraintDescription

用于在违反约束时说明该约束的字符串。例如，在没有约束条件描述的情况下，具有允许的 [A-Za-z0-9]+ 模式的参数会在用户指定无效值时显示以下错误消息：

```
Malformed input-Parameter MyParameter must match pattern [A-Za-z0-9]+
```

通过添加约束条件描述（例如“必须仅包含大写和小写字母和数字”），可以显示以下自定义错误消息：

```
Malformed input-Parameter MyParameter must contain only uppercase and
lowercase letters and numbers.
```

类型：字符串
必填项：False

minValue

一个数值，用于确定要允许Number类型使用的最小数值。

类型：整数
必需：False

maxValue

一个数值，用于确定要允许Number类型使用的最大数值。

类型：整数
必需：False

minLength

一个整数值，用于确定要允许String类型使用的最小字符数。

类型：整数
必需：False

maxLength

一个整数值，用于确定要允许的String类型的最大字符数。

类型：整数
必需：False

allowedValues

包含参数允许值列表的阵列。

类型：string 类型的数组
必填项：False

referencedByResources

使用此参数的 Amazon SAM 资源列表。

类型：string 类型的数组
必需：True

TooManyRequestsException

客户端每单位时间发送的请求数超过了允许的请求数。

message

客户端每单位时间发送的请求数超过了允许的请求数。

类型：字符串
必填项：False

errorCode

429

类型：字符串
必填项：False

Version

应用程序版本详情。

applicationId

应用程序 Amazon 资源名称 (ARN)。

类型：字符串
必需：True

semanticVersion

应用程序的语义版本：

https://semver.org/

类型：字符串
必需：True

sourceCodeUrl

指向应用程序源代码的公共存储库的链接，例如特定 GitHub 提交的 URL。

类型：字符串
必填项：False

sourceCodeArchiveUrl

指向 S3 对象的链接，其中包含此版本应用程序的源代码的 ZIP 存档。

最大大小 50 MB

类型：字符串
必填项：False

templateUrl

指向应用程序打包 Amazon SAM 模板的链接。

类型：字符串
必需：True

creationTime

此资源的创建日期和时间。

类型：字符串
必需：True

parameterDefinitions

应用程序支持的参数类型数组。

类型：ParameterDefinition 类型的数组
必需：True

requiredCapabilities

在部署某些应用程序之前必须指定的值列表。某些应用程序可能包含可能影响您 Amazon 账户权限的资源，例如，通过创建新 Amazon Identity and Access Management (IAM) 用户。对于这些应用程序，必须通过指定此参数来明确确认其功能。

唯一有效的值是CAPABILITY_IAM、CAPABILITY_NAMED_IAMCAPABILITY_RESOURCE_POLICY、和CAPABILITY_AUTO_EXPAND。

以下资源需要您指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM：AWS::IAM::Group、AWS::IAM::InstanceProfileAWS::IAM和AWS::IAM::Role。如果应用程序包含 IAM 资源，则可以指定CAPABILITY_IAM或CAPABILITY_NAMED_IAM。如果应用程序包含具有自定义名称的 IAM 资源，您必须指定 CAPABILITY_NAMED_IAM。

以下资源要求您指定CAPABILITY_RESOURCE_POLICY：AWS::Lambda::Permission、AWS:: iam:Policy、、AWS::ApplicationAutoScaling::ScalingPolicy、AWS::S3::BucketPolicy和。AWS::SQS::QueuePolic

包含一个或多个嵌套应用程序的应用程序要求您指定 CAPABILITY_AUTO_EXPAND。

如果您的应用程序模板包含上述任何资源，我们建议您在部署之前查看与该应用程序关联的所有权限。如果您没有为需要功能的应用程序指定此参数，则调用将失败。

类型：Capability 类型的数组
必需：True

resourcesSupported

检索该应用程序所在的区域是否支持此应用程序中包含的所有 Amazon 资源。

类型：布尔值
必需：True

文档历史记录

- API 版本：最新
- 上次文档更新日期：2020 年 3 月 10 日

下表描述此《Amazon Serverless Application Repository 开发人员指南》每次发布时进行的重要更改。要获得本文档的更新通知，您可以订阅 RSS 源。

变更	说明	日期
已发布应用程序的 IAM 权限控制	Amazon Serverless Application Repository 现在禁止发布将`AWSLambda_FullAccess` 托管策略附加到 Lambda 函数`iam:AttachRolePolicy` `iam:PutRolePolicy` 、授予内联 IAM 策略中的所有资源或`iam:*`授予或的新应用程序。有关更多信息，请参阅Amazon SAM 与 Amazon Serverless Application Repository和应用程序功能一起使用。	2026年4月15日
应用程序共享和限制访问权限的更新	增加了对与组织中的账户共享应用程序以及限制 Amazon 账户和 Amazon 组织访问公共应用程序的支持。 Amazon 有关与组织中的用户共享应用程序的更多示例，请参阅Amazon Serverless Application Repository 应用程序策略示例。有关限制访问公有应用程序的示例，请参阅 Amazon Serverless Applicati	2020 年 3 月 10 日

	on Repository 基于身份的策略示例。	
新的支持的资源	添加了对一些额外资源的支持。有关支持的资源的完整列表，请参阅支持的 Amazon 资源列表。	2020 年 1 月 17 日
中国地区	Amazon Serverless Application Repository 现已在中国区域、北京和宁夏推出。有关 Amazon Serverless Application Repository 区域和终端节点的更多信息，请参阅中的区域和终端节点Amazon Web Services 一般参考。	2020 年 1 月 15 日
更新了 “安全” 部分，使其与其他 Amazon 服务保持一致。	关更多信息，请参阅安全性。	2020 年 1 月 2 日
简化了发布应用程序的流程	Amazon SAM CLI 中的新`sam publish`命令简化了在中发布无服务器应用程序的 Amazon Serverless Application Repository过程。有关下载和发布示例应用程序的 end-to-end教程，请参阅快速入门：发布应用程序。有关发布已在 Amazon 云端开发和测试的应用程序的说明，请参阅通过 Amazon SAM CLI 发布应用程序。	2018 年 12 月 21 日
嵌套应用程序和层支持	增加了对嵌套应用程序和层的支持。这包括对支持的 Amazon 资源和确认应用程序功能的更新。	2018 年 11 月 29 日

使用自定义 IAM 角色和资源策略发布应用程序	增加了对使用自定义 IAM 角色和资源策略发布应用程序的支持。这包括 “使用应用程序” 和 “发布应用程序” 工作流程的更新以及《Amazon Serverless Application Repository 开发人员指南》中对支持的 Amazon 资源和 API 参考的更新。	2018 年 11 月 16 日
策略模板更新	《Amazon Serverless Application Repository 开发者指南》中对支持的策略模板进行了更新。	2018 年 9 月 26 日
文档更新	在《Amazon Serverless Application Repository 开发者指南》中添加了身份验证和访问控制主题。	2018 年 7 月 2 日
公开发行	的公开发布 Amazon Serverless Application Repository，现已在 14 个 Amazon 地区推出。有关可用 Amazon 区域和 Amazon Serverless Application Repository 终端节点的 Amazon Serverless Application Repository 更多信息，请参阅中的区域和终端节点Amazon Web Services 一般参考。	2018 年 2 月 20 日
新指南	这是《Amazon Serverless Application Repository 开发者指南》的第一个预览版。	2017 年 11 月 30 日

Amazon 词汇表

有关最新 Amazon 术语，请参阅《Amazon Web Services 词汇表 参考资料》中的Amazon 词汇表。

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。