用户指南

AWS DevOps 代理人

AWS DevOps 代理人: 用户指南

Table of Contents

关于 AWS DevOps 代理

AWS DevOps Agent 是一种边境代理，可以解决并主动预防事件，从而不断提高可靠性和性能。

AWS DevOps 工程师以经验丰富的 DevOps 工程师的身份调查事故并确定运营方面的改进。

该代理的工作方式是：

- 了解您的资源及其关系。
- 使用您的可观察性工具、技能、代码存储库和 CI/CD 管道。
- 关联遥测、代码和部署数据，以了解应用程序资源之间的关系。
- 支持多云和混合环境中的应用程序。

主要 功能

AWS DevOps Agent 通过以下功能提供全面的事件响应和预防功能：

始终在线、自主的事件响应

AWS DevOps Agent 会在问题发生的那一刻即自动进行调查：

- 自动事件调查 — 收到警报或支持请求后立即开始调查
- AWS DevOps Agent Chat-在整个 Ag DevOps ent Space Web 应用程序中使用自然语言查询您的基础架构、分析系统运行状况并指导调查。聊天会根据你正在查看的页面提供情境感知响应，无论是询问拓扑中的资源、指导调查还是在 Prevention 中筛选建议。
- 详细的缓解计划 — 提供具体措施来解决事件、验证成功并在需要时恢复更改
- 自动协调事件 — 通过您首选的沟通渠道（例如 Slack 和 ServiceNow
- AWS 支持集成 — 直接根据调查创建 AWS 支持案例，并向 AWS 支持专家提供即时背景信息

防止未来事件

AWS DevOps 代理分析历史事件的模式，以帮助您从被动的消防转变为主动的运营改进：

- 有针对性的建议 — 提供具体、可操作的改进，以加强四个关键领域：可观察性（监控、警报、记录）、基础架构优化（自动扩展、容量调整）和部署管道增强（测试、验证）。

- 持续学习 — 根据团队的反馈完善建议

从您的 DevOps 工具中获得更多收益

AWS DevOps Agent 无需更改工作流程即可与您的现有工具集成：

- 应用程序资源映射-生成应用程序资源及其关系的拓扑图
- 内置集成 — 可与流行的可观察性工具（Amazon CloudWatch、Dynatrace、Datadog、New Relic 和 Splunk）、代码存储库和 CI/CD 管道（GitHub 操作和存储库、工作流程和存储库、工作流程和存储库）配合使用 GitLab
- 自定义工具集成 — 通过连接到您自己的模型上下文协议 (MCP) 服务器来扩展功能，以获取其他工具
- 对话式基础设施查询 — 使用自然语言查询 AWS 资源、系统指标和警报状态，无需浏览多个控制台。Chat 可以理解上下文，并保留对话历史记录以备后续问题之用。

AWS DevOps 代理的工作原理

AWS DevOps 代理通过双控制台架构运行。管理员使用 AWS 管理控制台来创建和管理代理空间、配置集成以及设置访问控制。运营团队使用 AWS DevOps 代理 Web 应用程序进行 day-to-day事件响应和调查活动。在 Web 应用程序中，操作员可以与代理调查进行交互，浏览跨账户应用程序拓扑，并了解可观察性、代码、管道和基础设施架构的预防性改进。要了解更多信息，请参阅the section called “主动预防事故”。

该服务围绕代理空间进行组织，代理空间是定义 AWS DevOps 代理可以访问和调查的内容的逻辑容器。每个代理空间都包含您的 AWS 账户配置、第三方工具集成和访问权限。要了解更多信息，请参阅the section called “什么是 DevOps 代理空间？”。

AWS DevOps Agent 会自动构建映射您的资源及其关系的应用程序拓扑。此拓扑有助于服务在调查期间了解您的应用程序架构。要了解更多信息，请参阅the section called “什么是 DevOps 代理拓扑？”。

优势

- 缩短平均解决时间 (MTTR) — 自主调查立即开始，将事件解决时间从几小时缩短到几分钟
- 防止事件反复发生 — 针对性建议可解决根本原因并增强系统弹性
- 提高运营效率 — 将您的团队从重复的调查任务中解放出来，专注于创新
- 在现有工作流程中工作 — 无需中断即可与现有工具和流程集成

什么是 DevOps 代理 Web 应用程序？

AWS DevOps 代理使用双控制台架构，将管理功能与 day-to-day操作活动分开。这种设计使管理员能够配置服务，而运营团队则专注于事件响应和预防。

控制台

AWS DevOps 代理提供两个不同的接口：

- AWS 管理控制台-管理员使用 AWS 管理控制台来设置和管理 AWS DevOps 代理。在此控制台中，您可以the section called “创建代理空间”连接 AWS 服务和第三方工具，并管理组织的访问权限。
- DevOps 代理 Web 应用程序- 运营团队使用 A DevOps gent Space Web 应用程序进行日常事件响应活动。这个独立的应用程序提供了一个界面，待命工程师可以在其中启动调查、通过自然语言聊天与代理互动、查看应用程序拓扑以及查看事件预防建议。

Web 应用程序功能

代 DevOps 理 Web 应用程序提供以下主要功能：

- 事件响应- 您可以在该页面上创建和跟踪事件调查，并生成缓解计划以解决事件。
- 事件预防 — 在 “预防” 页面中，您可以在这里找到改善可观察性态势、交付流程和基础设施架构的建议，以防止将来发生事件。
- 拓扑- 拓扑页面提供了账户资源及其在关联账户中所有资源之间的关系的交互式可视化表示。您可以使用 “显示” 下拉列表在 “系统”、“容器” 和 “资源” 视图之间切换，查看具有不同详细级别的拓扑。
- 技能 — 模块化指令集，可扩展 AWS DevOps Agent 的专业能力。技能包括针对您的基础架构量身定制的领域知识、调查方法和工具配置。每种技能都支持特定的工具，并且只有在与调查相关的情况下才会逐步披露指令。
- 自然语言聊天界面 — Chat 是一款由 AI 驱动的对话助手，可在整个 Web 应用程序中使用，它使您能够使用自然语言查询基础架构、分析系统运行状况和进行调查。Chat 会根据您正在查看的页面提供情境感知响应。

身份验证

AWS DevOps 代理支持灵活的身份验证方法，以适应不同的组织需求：

- IAM Identity Center 集成（用户访问）— 组织可以使用 AWS 身份中心（IAM 身份中心）来集中管理用户对 A DevOps gent Space 网络应用程序的访问权限。IAM Identity Center 可以通过标准的

OIDC 和 SAML 协议与外部身份提供商联合，包括 Okta、Ping Identity 和 Microsoft Entra ID 等提供商。此方法支持您的身份提供商提供的多因素身份验证。

- 外部身份提供商 (IdP) 身份验证 — 组织可以将兼容 OIDC 的身份提供商（例如 Okta 或 Microsoft Entra ID）直接连接到 Agent Space 网络应用程序，而无需使用 IAM 身份中心。用户通过 IdP 使用其公司凭据登录。有关设置说明，请参阅the section called “设置外部身份提供商 (IdP) 身份验证”。
- IAM 身份验证链接（管理员访问权限）— 另一种方法允许您使用现有的控制台会话从 AWS 管理控制台直接访问 Web 应用程序。在实现完整的 Identity Center 集成之前，此选项很有用，但会话限制在 10 分钟以内。

什么是 DevOps 代理空间？

DevOps 代理空间是一个逻辑容器，用于定义 AWS DevOps 代理可以访问的工具和基础架构。每个 Agent Space 都独立运营，拥有自己的 AWS 账户访问权限、第三方集成和用户权限。

代理空间代表了 AWS DevOps 代理在事件响应期间可以访问和调查的边界。创建 Agent Space 时，您可以定义代理可以访问哪些 AWS 帐户、可以连接到哪些外部工具以及组织中的哪些用户可以与代理进行交互。

每个代理空间都充当 AWS DevOps 代理的独立部署。您可以通过 AWS 管理控制台配置座席空间，而您的运营团队则使用座席空间的 Web 应用程序在该空间内进行调查和查看建议。

代理空间是如何隔离的

Agent Spaces 保持隔离，以确保安全并防止跨不同环境或团队的意外访问：

- AWS 账户隔离 — 每个代理空间都使用专用 IAM 角色，这些角色仅授予对特定 AWS 账户和资源的访问权限。代理无法访问为代理空间明确配置的 AWS 资源之外的资源。
- 用户访问隔离-您可以控制哪些用户或组可以访问每个代理空间。这使您可以将访问权限与您的组织结构保持一致，从而确保团队仅与其指定的代理空间进行交互。
- 数据隔离 — 调查数据、事件历史记录和建议在每个代理空间中单独维护。来自一个代理空间的信息不可见，也无法从另一个代理空间访问。
- 聊天数据隔离-聊天对话历史记录也隔离在每个代理空间中。一个座席空间中的对话和查询不可见，也无法从另一个座席空间访问。

特工空间 Web 应用程序

每个 Agent Space 都有一个专用 Web 应用程序，可在 AWS 管理控制台之外进行访问。the section called “什么是 DevOps 代理 Web 应用程序？”要了解有关 Web 应用程序的更多信息，请参阅。

何时使用多个代理空间

考虑创建多个座席空间以支持不同的组织需求：

- 团队分离 — 为不同的应用程序团队或业务部门创建专用的代理空间，以保持代理空间中明确的所有权界限。
- 环境隔离 — 将生产环境和非生产环境分成不同的代理空间，以防止意外跨环境访问。
- 服务边界 — 使代理空间与特定的服务或应用程序边界保持一致，以保持调查的重点和相关性。
- 合规性要求 — 使用不同的访问控制或数据驻留设置配置单独的代理空间，以满足监管要求。

> **Note**
>
> 创建多个代理空间时，您可以使用专用 AWS 帐户作为代理空间的主帐户，并将不同的应用程序帐户作为辅助帐户进行连接。这种方法允许您保持精细的访问控制，同时确保即使使用自动角色创建功能，每个代理空间也只能访问特定于其预期范围的资源。

什么是 DevOps 代理拓扑？

AWS DevOps Agent's 会自动发现和可视化应用程序中的资源和关系，并在事件调查和提出预防性建议时使用生成的拓扑来了解您的基础架构。

拓扑图是如何创建的

AWS DevOps Agent 通过几个自动化流程生成拓扑图：

- 资源发现 — 代理会自动扫描您的 AWS 帐户，以识别作为应用程序一部分的资源，例如计算实例、存储服务、网络组件和数据库。
- 关系检测-代理分析配置数据、 CloudFormation 堆栈和资源标签，以确定资源如何相互关联。
- 代码和部署映射-连接到 CI/CD 管道时，代理会将基础架构资源链接回其部署流程以及更改的应用程序和基础架构代码。

- 可观察性行为映射 — 来自可观测性系统（例如 Amazon CloudWatch 应用程序信号和 Dynatrace）的数据用于识别观察到的表明资源之间关系的行为。

关键功能

资源映射提供了多种增强事件调查和预防的功能：

- 交互式可视化 — 通过操作员 Web 应用程序中的交互式图表探索您的应用程序拓扑。您可以缩放和浏览拓扑以了解资源之间的复杂关系。您也可以使用 Chat 使用自然语言查询拓扑信息，例如 “向我显示连接到此 DynamoDB 表的所有 Lambda 函数” 或 “此警报会影响哪些资源？”。
- 情境调查 — 在事故调查期间，资源拓扑结构可协助 AWS DevOps 代理识别受影响的组件，了解爆炸半径，并跟踪系统中的撞击路径。
- 根本原因分析 — 对资源关系的详细了解有助于查明问题的根源，即使在具有许多相互依赖关系的复杂分布式系统中也是如此。
- 影响评估 — 在分析事件时，代理可以通过识别拓扑中的依赖链来更好地确定哪些下游服务可能受到影响。
- 预防性建议 — 代理利用拓扑洞察力为弹性改进提出有针对性的建议，提出对系统稳定性影响最大的变更建议。

拓扑视图

Operator Web App 的 “拓扑” 页面中的拓扑可视化提供了多个详细级别：

- L earden — 默认视图，由特工空间理解技能生成。显示按逻辑服务和请求路径组织的基础架构的结构化摘要。
- 系统-显示高级账户和区域边界。
- 容器-显示部署堆栈，例如包含相关资源的 CloudFormation 堆栈。
- 组件-显示容器内的各个组件及其关系。
- 所有资源-显示包含所有已发现资源及其关系的完整视图。

资源发现

通过两种方法发现资源：

- CloudFormation 堆栈 — 代理列出主 AWS 账户和所有关联的辅助账户中的所有 CloudFormation 堆栈及其资源。任何 CloudFormation 用于部署的 infrastructure-as-code工具（包括 C AWS loud Development Kit (AWS CDK)）都支持此功能。
- 资源浏览器-对于未从中部署的资源 CloudFormation，将从资源浏览器中发现带标签的 AWS 资源。目标 AWS 账户必须启用资源浏览器。这对于识别通过 AWS 管理控制台 APIs、 AWS 服务或其他 infrastructure-as-code框架部署的资源的应用程序边界非常有用。

拓扑以外的调查范围

虽然应用程序拓扑在调查期间提供了重要的上下文，但 A AWS DevOps gent 不仅限于调查拓扑中显示的资源。代理可以使用其他数据源（例如 AWS 服务 APIs 或连接的可观测性工具）来调查不在应用程序拓扑中的资源。

要限制代理可以访问的资源，请将分配给代理的角色的策略限制为访问跨账户资源。有关更多信息，请参阅 the section called “限制 AWS 账户中的代理访问权限”。

拓扑和座席空间理解技能

拓扑图将输入到Agent Space理解所学技能中，该技能对基础架构的结构化摘要进行编码，以供调查期间使用。完成新代理空间的拓扑发现后，系统会自动生成座席空间理解技能。有关所学技能的更多信息，请参阅the section called “学到的技能”。

DevOps 特工技能

AWS DevOps Agent Skills 是模块化指令集，可通过针对您的基础设施和运营工作流程量身定制的专业领域知识和调查方法来扩展代理的能力。

什么是技能

技能是包含为代理提供专门功能的 Markdown 指令的独立目录。 AWS DevOps AWS DevOps 代理支持 Agent Skills 规范（打包代理指令和资源的开放标准）的子集，仅支持不可执行的文档：Markdown 指令 PDFs、图像和数据文件。

每项技能都需要一个 Skill.md 文件，其中包含你要为特工提供的说明。 AWS DevOps 除了必需的 Skill.md 文件外，技能还可能包括：

- 针对特定场景或基础设施类型的@@ 调查工作流程。
- 参考资料，包括架构模式和操作程序。

- 座席类型定位 — 可以将技能定位到特定的代理类型（通用、按需、事件分类、事件 RCA、事件缓解、评估），以减少情境消耗并提高代理的注意力。

为什么要使用技能

技能将 AWS DevOps Agent 从通用助理转变为基础架构和操作工作流程的专家。与聊天消息中提供的一次性说明不同，技能是可重复使用的功能，在与 AWS DevOps 代理执行的任务相关时会自动加载。

主要好处：

- 专业化您的代理 — Tailor AWS DevOps Agent 提供针对您的基础架构和运营模式的调查程序、最佳实践和组织知识。
- 减少重复 — 只需创建一次调查工作流程， AWS DevOps Agent 即可在所有相关调查中自动使用这些工作流程，无需重复提供相同的指导。
- 撰写能力-结合多种技能来构建 end-to-end调查工作流程。 AWS DevOps 代理在执行期间读取多种技能，例如用于从自定义 CI/CD 管道检索部署的技能和搜索代码存储库的技能。
- Amplify 自定义工具 — 创建指导 AWS DevOps 代理有效使用自定义 MCP 服务器工具的技能。技能可以记录何时调用特定工具、在不同场景中使用哪些参数，以及如何解释结果以完成特定于您的基础架构的工作流程。

技能是如何运作的

当 AWS DevOps Agent 遇到相关任务时，它会加载相应的技能并按照说明指导其调查。例如，“数据库性能调查” 技能可能包括分析 RDS 限制问题的 step-by-step程序，使代理能够系统地检查警报状态、分析连接指标和识别慢速查询。

技能结构

技能按目录进行组织，其中包含：

```
my-skill/
### SKILL.md              # Main skill instructions
### references/          # Optional: additional reference documentation
### assets/              # Optional: images, diagrams, data files
```

skill.md

`SKILL.md`是唯一的必填文件。它包含以 Markdown 格式编写的核心指令。这个文件应该：

- 描述何时以及如何使用该技能。
- 提供 step-by-step调查程序。
- 包括不同场景的决策树。
- 记录预期产出和成功标准。

前置问题

Frontmatter 是SKILL.md文件顶部的元数据块，封闭在---分隔符之间。它包含name和description字段， AWS DevOps 特工在调查或任务期间使用这些字段来确定何时激活技能。

```
---
name: rds-performance-investigation
description: Investigation procedures for RDS performance issues including
  connection exhaustion, slow queries, replication lag, and storage capacity.
  Use this skill when investigating database latency, connection errors, or
  read/write performance degradation.
---
```

名称-技能的唯一标识符。仅使用小写字母、数字和连字符（最多 64 个字符）。不得以连字符开头或结尾。

描述-详细说明 AWS DevOps 代理何时以及为何应使用此技能。 AWS DevOps 代理评估此字段以确定该技能是否与当前任务相关。即使说明写得很好，模糊或缺失的描述也可能导致特工完全跳过技能。

重要-从代理的角度写下描述。包括应触发技能的特定场景、服务、错误类型或症状。例如，“在调查 Amazon RDS 实例的数据库延迟、连接错误或查询超时时时使用此技能” 比 “RDS 技能” 更有效。

当你在用户界面中创建技能时，系统会根据你提供的名称和描述自动生成 frontmatter。以 zip 文件形式上传的技能必须在文件中包含 frontmatter SKILL.md。

示例：完成技能

以下示例显示了用于调查 RDS 性能问题的完整、成熟的技能。它演示了目录结构、Skill.md frontmatter、可行的调查程序和补充参考文件。

目录结构：

```
rds-performance-investigation/
```

```
### SKILL.md
### references/
#   ### rds-metrics-reference.md
### assets/
    ### rds-investigation-flowchart.png
```

skill.md：

```
---
name: rds-performance-investigation
description: Investigation procedures for RDS performance issues including
  connection exhaustion, slow queries, replication lag, and storage capacity.
  Use this skill when investigating database latency, connection errors, or
  read/write performance degradation.
---

# RDS Performance Investigation

Use this skill when customers report database latency, connection errors,
query timeouts, or read/write performance degradation.


## Step 1: Check alarm status

Query CloudWatch for active alarms on the affected RDS instance. Look for:
- `DatabaseConnections` exceeding 80% of max_connections
- `ReadLatency` or `WriteLatency` above 20ms
- `FreeStorageSpace` below 20% of total storage
- `ReplicaLag` above 30 seconds (read replicas only)


## Step 2: Analyze connection metrics

Retrieve `DatabaseConnections` over the past hour. If connections are near
the max_connections limit, check for connection pool misconfiguration or
long-running idle connections.


## Step 3: Identify slow queries

Use Performance Insights (`pi:GetResourceMetrics`) to retrieve the top SQL
statements by average active sessions. Focus on queries with high `db.load`
contribution or frequent I/O waits.
```

```
## Step 4: Summarize findings

Provide a summary with:
1. Current performance status (healthy / degraded / critical)
2. Root cause hypothesis with supporting metrics
3. Recommended remediation steps ranked by priority
```

references/ .mdrds-metrics-reference:

```
# RDS CloudWatch Metrics Reference


| Metric | Normal Range | Investigation Threshold |
|---|---|---|
| DatabaseConnections | < 70% max_connections | > 80% max_connections |
| ReadLatency | < 5ms | > 20ms |
| WriteLatency | < 5ms | > 20ms |
| FreeStorageSpace | > 30% total storage | < 20% total storage |
| ReplicaLag | < 5 seconds | > 30 seconds |
| CPUUtilization | < 70% | > 85% |
```

创造技能

在创建技能之前，你必须有一个特工空间。有关更多信息，请参阅 the section called “创建代理空间”。

您可以通过两种方式创建技能，具体取决于您的工作流程偏好和技能复杂性：

在 UI 中创建技能

在 A AWS DevOps gent Operator Web 应用程序中创建的技能在一个 Skill.md 文件中包含名称、描述和说明。

要在用户界面中创建技能，请执行以下操作：

- 在 Agent Space Operator Web 应用程序中导航到 “技能” 页面。
- 单击 “添加技能”。
- 从模式中选择 “创建技能”。
- 填写技能表：

  - 名称-仅限小写字母、数字和连字符（最多 64 个字符）。不得以连字符开头或结尾。示例：`rds-throttling-investigation`
  - 描述-简要说明何时使用此技能（建议最少 100 个字符，最多 1,024 个字符）。这可以帮助代理确定何时激活技能。
  - 状态-设置为 “活动”（默认）或 “非活动”。特工不使用非活动技能。
  - 代理类型-选择一个或多个可以使用此技能的代理类型。默认情况下，“通用” 处于选中状态，该技能可供所有代理类型使用。要定位特定的代理，请取消选择 “通用”，然后选择：按需、事件分类、事件 RCA、事件缓解或评估。
  - 说明 — Markdown 格式的 Step-by-step程序。要具体且具有可操作性。
- 单击 “创建” 保存技能。

系统会自动生成一个具有适当前题结构的 Skill.md 文件。

要编辑在 UI 中创建的技能，请执行以下操作：

- 导航到技能列表中的技能，然后单击该技能将其打开。
- 单击编辑。
- 修改名称、描述或说明。
- 单击 “保存” 更新技能。

上传技能

以 zip 文件形式上传的技能包含 Skill.md 文件以及其他资源，例如参考资料或资产。

技能结构：

```
my-skill.zip
### SKILL.md              # Required: main skill instructions
### references/          # Optional: reference documentation
#   ### architecture.md
#   ### troubleshooting.md
### assets/              # Optional: images, diagrams, data files
    ### topology.png
    ### metrics.csv
```

Skill.md 前题要求：

作为 zip 文件上传的技能必须包含 skill.md 中的前题，以及name和字段。description AWS DevOps 代理使用这些字段来确定何时激活技能。有关撰写有效 frontmatter 的详细信息，请参阅本主题前面的 Frontmatter 部分。

```
---
name: rds-performance-analysis
description: Comprehensive RDS performance investigation procedures
  for connection exhaustion, slow queries, and storage capacity issues.
  Use when investigating database latency or read/write degradation.
---


# RDS Performance Analysis


[Your skill instructions here...]
```

要通过 zip 上传创建技能，请执行以下操作：

- 按照上述结构创建一个包含技能文件的目录。
- 确保 Skill.md 包含正确的前言（名称和描述）。
- 将该目录压缩成.zip 文件。
- 在 Agent Space Operator Web 应用程序中导航到 “技能” 页面。
- 单击 “添加技能”。
- 从模式中选择 “上传技能”。
- 拖放您的.zip 文件或单击进行浏览（仅限 ZIP 文件，最大 6 MB）。
- 选择一个或多个可以使用此技能的代理类型（默认情况下选择通用并适用于所有代理类型；取消选择则专门针对按需、事件分类、事件 RCA、事件缓解或评估）。
- 查看 zip 文件要求和验证结果。
- 单击 “上传” 将技能添加到您的特工空间。

以 zip 文件形式上传技能的重要限制：

- 目前不支持脚本 — scripts/ 目录中包含脚本的技能将在上传过程中被拒绝。一旦代理可以访问安全的编码环境，脚本执行功能将在未来的版本中启用。
- 大小限制-压缩文件总大小不得超过 6 MB（包括所有文件）。
- 需要 skill.md — zip 文件必须包含一个带有有效 frontmatter 的 Skill.md 文件。

命名技巧的最佳实践：

使用清晰的描述性名称，比如 “rds-throttling-investigation”，而不是通用名称。一个好的技能名称反映了它所涉及的特定场景或服务，因此可以更轻松地一目了然地识别出正确的技能。

管理技能

AWS DevOps Agent 通过操作员 Web 应用程序提供全面的技能管理功能：

列出技能 — 查看特工空间中的所有技能。技能页面显示技能名称、活动或非活动状态、创建日期、上次更新日期和可用操作。

查看技能 — 单击任意技能可查看其详细视图。在 UI 中创建的技能会显示可编辑的内容，您可以直接在 UI 中修改名称、描述或说明，然后单击 “保存” 进行更新。作为 zip 文件上传的技能会显示一个文件树，其中显示 Skill.md 以及任何其他目录，例如引用/ 和 assets/。单击树中的文件可在只读模式下查看其内容。

为技能选择代理-配置在创建或编辑技能时可以使用每种技能的代理类型。在座席类型下拉列表中，使用复选框选择一个或多个代理类型：通用（默认-适用于所有座席类型）、按需（对话查询）、事件分类（初始事件评估）、事件 RCA（根本原因分析）、事件缓解（自动事件响应）或评估（主动建议）。默认情况下，“通用” 处于选中状态，该技能可供所有代理类型使用。针对特定代理的技能可以减少情境消耗并提高代理的注意力。

激活和停用技能-暂时禁用技能而不使用 Active/Inactive 切换按钮将其删除。打开技能详细信息视图并将开关切换到 “非活动”，以防止代理在保留所有内容和配置的同时加载技能详细信息以进行新的调查。正在进行的调查仍在继续使用该技能。切换回 “激活” 以使该技能立即再次可用。

更新技能-根据现有技能的创建方式对其进行修改。对于在用户界面中创建的技能，请在技能详细信息视图中单击 “编辑”，修改名称、描述或说明，然后单击 “保存” 进行更新。对于以 zip 文件形式上传的技能，请在本地修改文件，创建新的 zip 文件，然后上传新版本。

删除技能-永久移除特工空间中的技能。打开技能列表视图，点击更多选项菜单 () 并选择 “删除”，查看有关永久删除的警告，键入要确认的技能名称，然后单击 “删除技能”。删除操作无法撤消。如果正在进行的调查尝试加载已删除的技能，则可能会受到影响。对于以 zip 文件形式上传的技能，请先下载 zip 文件，然后再将其作为备份删除。如果您再次需要该技能，可以考虑停用该技能，而不是将其删除。

从 Runbook 迁移

现有 Runbook 会自动迁移到技能，无需客户采取任何行动。当你的 Agent Space 过渡到技能模型时，所有 Runbook 都将转换为技能并出现在你的技能用户界面中。迁移后，您可以：

- 查看迁移的技能-检查自动迁移是否正确转换了您的 Runbook。
- 根据需要更新-直接在用户界面中编辑技能以完善说明、更新描述或配置代理类型定位。
- 使用参考文献进行扩展 — 对于可以从其他参考资料或架构图中受益的技能，请将它们重新创建为带有参考文献或资产/目录的 zip 上传技能。
- 创建新技能-为 Runbook 以前未涵盖的调查工作流程添加新技能。

如果您在自动迁移的技能方面遇到任何问题，或者需要有关迁移后更新的帮助，请联系 AWS Support。

学到的技能

什么是学到的技能？

所学技能是特工根据您的 DevOps 代理空间数据生成的结构化知识文件。每项学习的技能都对特 AWS DevOps 工在执行任务时使用的特定知识类型进行编码。在发布时，有两种学习的技能可供选择：Agent Space 理解和工具使用最佳实践。

代理空间理解

Agent Space 理解技能 (`understanding-agent-space`) 分析您连接的云帐户、代码存储库和遥测集成，以绘制代理空间中的资源和关系地图。

该技能会生成一个主`SKILL.md`文件和一组参考文件。主文件包含包含关键域概念的简单语言系统概述、部署环境（AWS 账户和区域对、Azure 订阅和区域等）、显示逻辑服务如何连接的容器级架构图、对应用程序至关重要的请求路径及其遍历的组件，以及代码存储库到容器的映射。

每个逻辑容器都会收到一个专门的参考文件，该文件描述了其内部组件（计算、数据、消息、网络等）以及资源类型和物理标识符（例如 ARNs表名和队列） URLs。参考文件还捕获了可观测性覆盖范围，包括与每个组件关联的警报、仪表板和监视器。它还将每个组件映射到其关联的代码存储库、包和infrastructure-as-code定义，从而提供从源代码到已部署资源的完整可追溯链。

每个关键请求路径都会收到一个专门的参考文件，该文件描述了从入口点到每个中间服务、数据存储和外部依赖关系的组件粒度的完整 end-to-end请求流。该文件包括一个按顺序排列的流程图，显示了组件之间的操作顺序和交互机制，以及每个参与者的责任。它还对与路径相关的可观察性信号进行分类：每个跳的日志组模式、关键指标（延迟、错误率、限制、令牌配额）及其警报名称和维度，以及可以跨服务和账户关联的分布式跟踪跨度。

工具使用最佳实践

Tool Use Best Practices 技能分析了过去使用的调查工具，以提取有效的使用模式、常见的故障模式和参数指导。这有助于 DevOps 代理避免已知的陷阱，减少浪费的步骤进行调查。该技能会生成一个主文件和一组每个工具的参考文件。主文件用作路由索引，其中列出了每个工具及其支持的调查方案，并链接到相应的参考文件。

每个工具的参考文件最多可以包含三个部分：

- 最佳实践 — 从成功使用工具中提取的以调查为导向的技术，例如 CloudWatch Logs Insights 查询模板、特定于环境的指标命名空间和维度以及事件源过滤器。 CloudTrail 每个条目都是围绕调查情景组织的，包括具体的参数值和在过去的调查中观察到的示例。
- 常见错误-反复出现的故障模式及其修复方法。每个条目都描述了特定的错误情况，例如查询无法访问的帐户或构造格式错误的聚合查询，并提供了纠正措施，以便代理可以在不浪费调查步骤的情况下避免错误或从错误中恢复。
- 输出管理 — 针对往往会返回大量响应的工具调用指南。每个条目都描述了一种参数更改或处理策略，该策略可在保持诊断值的同时减小输出大小。

当可以访问实时基础架构时，该技能会先根据您的环境验证模式，然后再将其包括在内。已确认的模式是自信地陈述的，未经证实的模式使用谨慎的语言，不包括被驳回的模式。这样可以使技能与基础架构的当前状态保持一致。

管理所学技能

更新 — DevOps 代理会根据你在特工空间中的活动自动生成和更新学到的技能。以下内容描述了每项技能的更新时间。

DevOps 代理每 30 次调查生成一个更新的 “工具使用最佳实践” 技能。

Agent Space 理解技能由学习代理生成，每当您添加、更新或移除 Agent Space 功能或集成时，该技能都会运行。

要手动重新生成已学技能，请在操作员应用程序的拓扑页面上选择重新生成按钮，或者与代理聊天并要求其更新所学技能。

停用-默认情况下，学到的技能处于激活状态。处于活动状态时， DevOps 代理会在每个 DevOps 代理任务开始时加载它们。要阻止应用已学到的技能，请在操作员应用程序的技能查看器中将其停用。停用技能不会将其删除。该技能会被保留，可以随时重新激活。当技能被停用时， DevOps 特工会在该技能不知情的情况下进行操作。

拓扑视图 — Agent Space 的 Web 应用程序中的拓扑页面使用座席空间理解技能将你的座席空间环境直观地显示为逻辑容器和组件。单击任何容器即可查看其组件、资源标识符和遥测数据。

支持的区域：

本主题描述了您可以使用 AWS DevOps 代理的 AWS 区域。有关 AWS 区域的更多信息，请参阅《账户管理参考指南》中的 “指定您的AWS 账户可以使用的 AWS 区域”。

跨区域资源监控

AWS DevOps 无论您在哪个支持 AWS 区域创建代理空间，代理都可以监控和调查位于任何区域的 AWS 账户中的资源。当您将 AWS 账户与代理空间关联时，代理会发现该账户内所有区域的资源并将其映射。这意味着您不需要在工作负载运行的每个区域都有一个代理空间。

根据您的首选数据驻留地、与运营团队的距离或组织要求选择支持的区域。

支持的区域：

AWS DevOps 代理可在以下 AWS 地区使用。

区域名称	区域代码	控制台链接
美国东部（弗吉尼亚州北部）	us-east-1	打开控制台
美国西部（俄勒冈州）	us-west-2	打开控制台
亚太地区（悉尼）	ap-southeast-2	打开控制台
亚太地区（东京）	ap-northeast-1	打开控制台
欧洲地区（法兰克福）	eu-central-1	打开控制台
欧洲地区（爱尔兰）	eu-west-1	打开控制台

服务端点

区域名称	区域代码	端点	协议
美国东部 (弗吉尼亚北部)	us-east-1	aidevops.us-east-1.amazonaws.com	HTTPS
美国西部（俄勒冈州）	us-west-2	aidevops.us-west-2.amazonaws.com	HTTPS
亚太地区（悉尼）	ap-southeast-2	aidevops.ap-southeast-2.amazonaws.com	HTTPS
亚太地区（东京）	ap-northeast-1	aidevops.ap-northeast-1.amazonaws.com	HTTPS
欧洲地区（法兰克福）	eu-central-1	aidevops.eu-central-1.amazonaws.com	HTTPS
欧洲地区（爱尔兰）	eu-west-1	aidevops.eu-west-1.amazonaws.com	HTTPS

注意事项

- Agent Space 区域选择 — 代理空间及其数据（调查、

拓扑、推荐）存储在您创建拓扑的区域中。选择符合您的数据驻留要求的区域。

- 跨区域监控-与代理关联的 AWS 账户中的资源

无论这些资源部署在哪个区域，都会对空间进行监控。您无需在运行工作负载的每个区域中创建单独的代理空间。

- 第三方集成 — 与 CI/CD 提供商的连接 (GitHub、 GitLab)、

可观察性工具（Dynatrace、Datadog、New Relic、Splunk）和 MCP 服务器是按代理空间配置的，不依赖于区域。

开始使用 AWS DevOps 代理

在本入门指南中，您将创建一个基本的代理空间，配置最低权限，并进行首次基于人工智能的调查。

主题：

创建代理空间

代理空间定义了 AWS DevOps 代理可以访问的工具和基础架构。本指南将引导您创建代理空间、配置主账户访问权限和启用 DevOps 代理 Web 应用程序。要了解有关代理空间概念的更多信息，请参阅“什么是代理空间”。

创建代理空间

访问 AWS DevOps 代理控制台

1. 登录到 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台

命名代理空间

1. 单击“创建代理空间”

在“代理空间详细信息”部分中，提供：

1. 在“名称”字段中，输入代理空间的名称
2. （可选）在描述字段中，添加有关代理空间用途的详细信息

3. （可选）从代理响应语言下拉列表中，选择代理生成响应、调查结果和调查结果时使用的语言。选项包括：印尼语、中文（Simplified/PRC), Chinese (Traditional/Taiwan）、英语（英国）、法语（法国）、德语（德国）、意大利语（意大利）、日语（日本）、韩语（韩国）、葡萄牙语（巴西）、西班牙语（拉丁美洲）、土耳其语（土耳其）、阿拉伯语（沙特阿拉伯）、泰语（泰国）和越南语（越南）。如果未选择任何语言，则代理将使用输入的语言进行响应。

配置主账户访问权限

在 “授予此代理空间 AWS 资源访问权限” 部分，您将设置一个 IAM 角色以向主 AWS 账户授予代理空间访问权限。主 AWS 账户是您创建代理空间的账户。 AWS DevOps 代理需要一个 IAM 角色才能在调查期间发现和访问该账户中的 AWS 资源。

选择角色配置方法。 选择下列选项之一：

选项 1：自动创建新的 AWS DevOps 代理角色（推荐）

此选项会自动创建一个具有相应权限的角色，让 AWS DevOps 代理可以调查您账户中的资源。

> **Note**
>
> 要使用此选项，您必须拥有 IAM 权限才能创建新角色。

1. 选择 “自动创建新的 AWS DevOps 代理角色”
2. （可选）更新要创建的代理空间角色名称

选项 2：分配现有角色

如果其他管理员之前专门为 AWS DevOps 代理创建了角色，则使用此选项。

1. 选择 “分配现有角色”
2. 从下拉菜单中，选择具有相应权限的现有角色

选项 3：使用策略模板创建新的 AWS DevOps 代理角色

当您需要限制代理在主账户中可以访问的服务和资源时，请使用此选项。

1. 选择 “使用策略模板创建新的 AWS DevOps 代理角色”
2. 按照说明创建新角色的信任策略和内联策略。

启用 Agent Space Web 应用程序

Web 应用程序是工作人员与 AWS DevOps 代理进行互动以进行事件调查和审查建议的地方。要了解更多信息，请参阅 AWS DevOps 代理控制台架构 [链接]。启用后，用户可以通过 AWS 管理控制台中的 IAM 身份验证链接访问 Agent Space Web 应用程序。

选择下列选项之一：

选项 1：自动创建新的 AWS DevOps 代理角色（推荐）

此选项会自动创建一个具有访问 DevOps 代理 Web 应用程序的相应权限的角色。

> **Note**
>
> 要使用此选项，您必须拥有 IAM 权限才能创建新角色。

1. 选择 “自动创建新的 AWS DevOps 代理角色”
2. 查看将授予该角色的权限

选项 2：分配现有角色

如果其他管理员先前创建了操作员角色，则使用此选项。

1. 选择 “分配现有角色”
2. 从下拉菜单中，选择具有相应权限的现有角色

选项 3：使用策略模板创建新的 AWS DevOps 代理角色

当您需要自定义 Web 应用程序访问权限时，请使用此选项。

1. 选择 “使用策略模板创建新的 AWS DevOps 代理角色”
2. 按照说明创建新角色的信任策略和内联策略。

添加标签（可选）

在创建过程中，您可以在代理空间中添加 AWS 标签。标签是键值对，可帮助您整理和识别资源。每个代理空间最多可以添加 50 个标签。要添加标签，请展开 “创建代理空间” 页面上的 “标签” 部分，然后单击 “添加新标签”。

完成代理空间创建

填写完所有部分后，单击 “创建”

验证您的代理空间设置

配置完成后，操作员访问按钮将出现在代理空间详细信息页面上。单击它将在新选项卡中打开 Web 应用程序并成功进行身份验证。

后续步骤

设置代理空间后，请考虑以下步骤：

- 如果您的应用程序跨多个账户，请添加辅助 AWS 账户
- 配置第三方集成，例如可观测性工具或票务系统
- 为生产环境设置 AWS 身份中心身份验证
- 浏览您的应用程序资源映射以帮助 AWS DevOps Agent 了解您的基础架构

AWS DevOps 代理 CLI 入门指南

概述

使用 AWS DevOps Agent，您可以监控和管理您的 AWS 基础架构。本指南将引导您使用 AWS 命令行界面 (AWS CLI) 设置 AWS DevOps 代理。您可以创建 IAM 角色、设置代理空间并关联您的 AWS 账户。您还可以启用操作员应用程序，并可以选择连接第三方集成。本指南大约需要 20 分钟才能完成。

AWS DevOps 代理可在六个 AWS 地区使用：美国东部（弗吉尼亚北部）、美国西部（俄勒冈）、亚太地区（悉尼）、亚太地区（东京）、欧洲（法兰克福）和欧洲（爱尔兰）。有关支持的区域的更多信息，请参阅the section called “支持的区域：”。

先决条件

开始之前，请确保您拥有：

- AWS 已安装并配置 CLI 版本 2
- 对您的 AWS 监控账户进行身份验证

- 创建 AWS 身份和访问管理 (IAM) Access Management 角色和附加策略的权限
- 用作监控 AWS 账户的账户
- 熟悉 AWS CLI 和 JSON 语法

在本指南中，请使用您自己的占位符值替换以下占位符值：

- <MONITORING_ACCOUNT_ID>— 您的监控（主要）AWS 账户的 12 位数账户 ID
- <EXTERNAL_ACCOUNT_ID>— 要监控的辅助 AWS 账户的 12 位数账户 ID（在步骤 4 中使用）
- <REGION>— 您的代理空间的 AWS 区域代码（例如，us-east-1或eu-central-1）
- <AGENT_SPACE_ID>— create-agent-space 命令返回的代理空间标识符

IAM 角色设置

1. 创建 DevOps 代理空间角色

通过运行以下命令创建 IAM 信任策略：

```
cat > devops-agentspace-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>"
        },
        "ArnLike": {
          "aws:SourceArn":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/*"
        }
      }
    }
  ]
}
```

```
EOF
```

创建 IAM 角色：

```
aws iam create-role \
  --region <REGION> \
  --role-name DevOpsAgentRole-AgentSpace \
  --assume-role-policy-document file://devops-agentspace-trust-policy.json
```

运行以下命令保存角色 ARN：

```
aws iam get-role --role-name DevOpsAgentRole-AgentSpace --query 'Role.Arn' --output
 text
```

附加 AWS 托管策略：

```
aws iam attach-role-policy \
  --role-name DevOpsAgentRole-AgentSpace \
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
```

创建并附加内联策略以允许创建资源管理器服务相关角色：

```
cat > devops-agentspace-additional-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateServiceLinkedRoles",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF
```

```
aws iam put-role-policy \
  --role-name DevOpsAgentRole-AgentSpace \
  --policy-name AllowCreateServiceLinkedRoles \
  --policy-document file://devops-agentspace-additional-policy.json
```

2. 创建操作员应用程序 IAM 角色

通过运行以下命令创建 IAM 信任策略：

```
cat > devops-operator-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>"
        },
        "ArnLike": {
          "aws:SourceArn":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/*"
        }
      }
    }
  ]
}
EOF
```

创建 IAM 角色：

```
aws iam create-role \
  --role-name DevOpsAgentRole-WebappAdmin \
  --assume-role-policy-document file://devops-operator-trust-policy.json \
  --region <REGION>
```

运行以下命令保存角色 ARN：

```
aws iam get-role --role-name DevOpsAgentRole-WebappAdmin --query 'Role.Arn' --output
 text
```

附上 AWS 托管运营商应用程序政策：

```
aws iam attach-role-policy \
  --role-name DevOpsAgentRole-WebappAdmin \
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsOperatorAppAccessPolicy
```

此托管策略向操作员应用程序授予访问代理空间功能的权限。这些功能包括调查、推荐、知识管理、聊天和 Su AWS pport 集成。该策略使用`aws:PrincipalTag/AgentSpaceId`条件限制对特定代理空间的访问权限。有关完整操作列表的更多信息，请参阅the section called “DevOps 代理 IAM 权限”。

入职步骤

1. 创建代理空间

运行以下命令创建代理空间：

```
aws devops-agent create-agent-space \
  --name "MyAgentSpace" \
  --description "AgentSpace for monitoring my application" \
  --region <REGION>
```

（可选）指定`--kms-key-arn`使用客户托管的 AWS KMS 密钥进行加密。您还可以使用`--tags`添加资源标签和`--locale`设置代理响应的语言。

保存`agentSpaceId`来自响应的（位于`agentSpace.agentSpaceId`）。

要稍后列出您的代理空间，请运行以下命令：

```
aws devops-agent list-agent-spaces \
  --region <REGION>
```

2. 关联您的 AWS 账户

关联您的 AWS 账户以开启拓扑发现。将设置`accountType`为以下值之一：

- `monitor`— 存在代理空间的主账户。此帐户托管代理并用于拓扑发现。
- `source`— 代理监控的额外账户。在步骤 4 中关联外部帐户时，请使用此类型。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id aws \
  --configuration '{
    "aws": {
      "assumableRoleArn": "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/DevOpsAgentRole-
AgentSpace",
      "accountId": "<MONITORING_ACCOUNT_ID>",
      "accountType": "monitor"
    }
  }' \
  --region <REGION>
```

3. 启用操作员应用程序

身份验证流程可以使用 IAM、IAM 身份中心 (IDC) 或外部身份提供商 (IdP)。运行以下命令为您的代理空间启用操作员应用程序：

```
aws devops-agent enable-operator-app \
  --agent-space-id <AGENT_SPACE_ID> \
  --auth-flow iam \
  --operator-app-role-arn "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/DevOpsAgentRole-
WebappAdmin" \
  --region <REGION>
```

要进行 IAM 身份中心身份验证，请使用`--auth-flow idc`并提供`--idc-instance-arn`。对于外部身份提供商，请使用`--auth-flow idp`并提供`--issuer-url`、`--idp-client-id`、和`--idp-client-secret`。有关更多信息，请参阅the section called “设置 IAM 身份中心身份验证”和the section called “设置外部身份提供商 (IdP) 身份验证”。

> 注意：如果您之前为账户中的其他座席空间创建了操作员应用程序角色，则可以重复使用该角色 ARN。

4.（可选）关联其他源账户

要使用 AWS DevOps 代理监控其他账户，请创建 IAM 跨账户角色。

在外部账户中创建跨账户角色

切换到外部账户并创建信任策略。MONITORING_ACCOUNT_ID是托管您在步骤 2 中设置的代理空间的主账户。此配置允许 AWS DevOps 代理服务代表监控帐户在辅助源帐户中扮演角色。

运行以下命令来创建信任策略：

```
cat > devops-cross-account-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<MONITORING_ACCOUNT_ID>",
          "sts:ExternalId":
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/<AGENT_SPACE_ID>"
        }
      }
    }
  ]
}
EOF
```

创建跨账户 IAM 角色：

```
aws iam create-role \
  --role-name DevOpsAgentCrossAccountRole \
  --assume-role-policy-document file://devops-cross-account-trust-policy.json
```

运行以下命令保存角色 ARN：

```
aws iam get-role --role-name DevOpsAgentCrossAccountRole --query 'Role.Arn' --output
 text
```

附加 AWS 托管策略：

```
aws iam attach-role-policy \
  --role-name DevOpsAgentCrossAccountRole \
  --policy-arn arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
```

附加内联策略以允许在外部账户中创建资源管理器服务相关角色：

```
cat > devops-cross-account-additional-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateServiceLinkedRoles",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::<EXTERNAL_ACCOUNT_ID>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

aws iam put-role-policy \
  --role-name DevOpsAgentCrossAccountRole \
  --policy-name AllowCreateServiceLinkedRoles \
  --policy-document file://devops-cross-account-additional-policy.json
```

关联外部账户

切换回您的监控账户，然后运行以下命令关联外部账户：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id aws \
  --configuration '{
    "sourceAws": {
      "accountId": "<EXTERNAL_ACCOUNT_ID>",
      "accountType": "source",
      "assumableRoleArn": "arn:aws:iam::<EXTERNAL_ACCOUNT_ID>:role/
DevOpsAgentCrossAccountRole"
```

```
    }
  }' \
  --region <REGION>
```

5.（可选）助理 GitHub

> 注意：必须先使用 OAuth 流程 GitHub 通过 AWS DevOps 代理控制台进行注册，然后才能通过 CLI 进行关联。

有关 GitHub 通过控制台进行注册的说明，请参阅the section called “连接到 CI/CD 管道”。

列出注册的服务：

```
aws devops-agent list-services \
  --region <REGION>
```

保存<SERVICE_ID>服务类型：。github

在控制台 GitHub 中注册后，通过运行以下命令关联 GitHub 存储库：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "github": {
      "repoName": "<GITHUB_REPO_NAME>",
      "repoId": "<GITHUB_REPO_ID>",
      "owner": "<GITHUB_OWNER>",
      "ownerType": "organization"
    }
  }' \
  --region <REGION>
```

6.（可选）注册并关联 ServiceNow

首先，使用 OAuth 凭据注册 ServiceNow 服务：

```
aws devops-agent register-service \
  --service servicenow \
  --service-details  '{
    "servicenow": {
      "instanceUrl": "<SERVICENOW_INSTANCE_URL>",
```

```
      "authorizationConfig": {
        "oAuthClientCredentials": {
            "clientName": "<SERVICENOW_CLIENT_NAME>",
            "clientId": "<SERVICENOW_CLIENT_ID>",
            "clientSecret": "<SERVICENOW_CLIENT_SECRET>"
        }
      }
    }
  }' \
  --region <REGION>
```

保存返回的<SERVICE_ID>，然后关联 ServiceNow：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "servicenow": {
      "instanceUrl": "<SERVICENOW_INSTANCE_URL>"
    }
  }' \
  --region <REGION>
```

7.（可选）注册并关联 Dynatrace

首先，使用凭据注册 Dynatrace OAuth 服务：

```
aws devops-agent register-service \
  --service dynatrace \
  --service-details '{
  "dynatrace": {
    "accountUrn": "<DYNATRACE_ACCOUNT_URN>",
    "authorizationConfig": {
        "oAuthClientCredentials": {
            "clientName": "<DYNATRACE_CLIENT_NAME>",
            "clientId": "<DYNATRACE_CLIENT_ID>",
            "clientSecret": "<DYNATRACE_CLIENT_SECRET>"
        }
      }
    }
  }' \
  --region <REGION>
```

保存返回的<SERVICE_ID>，然后关联 Dynatrace。资源是可选的。该环境指定要与哪个 Dynatrace 环境相关联。

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "dynatrace": {
      "envId": "<DYNATRACE_ENVIRONMENT_ID>",
      "resources": [
        "<DYNATRACE_RESOURCE_1>",
        "<DYNATRACE_RESOURCE_2>"
      ]
    }
  }' \
  --region <REGION>
```

响应中包含用于集成的 webhook 信息。你可以使用这个 webhook 来触发 Dynatrace 的调查。有关更多信息，请参阅 the section called “连接 Dynatrace”。

8.（可选）注册并关联 Splunk

首先，使用 BearerToken 凭据注册 Splunk 服务。

端点使用以下格式：`https://<XXX>.api.scs.splunk.com/<XXX>/mcp/v1/`

```
aws devops-agent register-service \
  --service mcpserversplunk \
  --service-details '{
  "mcpserversplunk": {
    "name": "<SPLUNK_NAME>",
    "endpoint": "<SPLUNK_ENDPOINT>",
    "authorizationConfig": {
        "bearerToken": {
            "tokenName": "<SPLUNK_TOKEN_NAME>",
            "tokenValue": "<SPLUNK_TOKEN_VALUE>"
        }
      }
    }
  }' \
  --region <REGION>
```

保存返回的内容<SERVICE_ID>，然后关联 Splunk：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpserversplunk":  {
      "name": "<SPLUNK_NAME>",
      "endpoint": "<SPLUNK_ENDPOINT>"
    }
  }' \
  --region <REGION>
```

响应中包含用于集成的 webhook 信息。你可以使用这个 webhook 来触发 Splunk 的调查。有关更多信息，请参阅 the section called “连接 Splunk”。

9.（可选）注册并关联新遗物

首先，使用 API 密钥凭据注册 New Relic 服务。

地区：要US么是EU。

可选字段：applicationIds、entityGuids、alertPolicyIds

```
aws devops-agent register-service \
  --service mcpservernewrelic \
  --service-details '{
    "mcpservernewrelic": {
      "authorizationConfig": {
        "apiKey": {
          "apiKey": "<YOUR_NEW_RELIC_API_KEY>",
          "accountId": "<YOUR_ACCOUNT_ID>",
          "region": "US",
          "applicationIds": ["<APP_ID_1>", "<APP_ID_2>"],
          "entityGuids": ["<ENTITY_GUID_1>"],
          "alertPolicyIds": ["<POLICY_ID_1>"]
        }
      }
    }
  }' \
  --region <REGION>
```

保存返回的物品<SERVICE_ID>，然后关联新遗物：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpservernewrelic":  {
      "accountId": "<YOUR_ACCOUNT_ID>",
      "endpoint": "https://mcp.newrelic.com/mcp/"
    }
  }' \
  --region <REGION>
```

响应中包含用于集成的 webhook 信息。你可以使用这个 webhook 来触发来自 New Relic 的调查。有关更多信息，请参阅 the section called “连接新遗物”。

10.（可选）注册并关联 Datadog

必须先通过 AWS DevOps 代理控制台使用 OAuth 流程注册 Datadog，然后才能通过 CLI 将其关联。有关更多信息，请参阅 the section called “正在连接 DataDog”。

列出注册的服务：

```
aws devops-agent list-services \
  --region <REGION>
```

保存<SERVICE_ID>服务类型：。mcpserverdatadog

然后关联 Datadog：

```
aws devops-agent associate-service \
  --agent-space-id <AGENT_SPACE_ID> \
  --service-id <SERVICE_ID> \
  --configuration '{
    "mcpserverdatadog": {
      "name": "Datadog-MCP-Server",
      "endpoint": "<DATADOG_MCP_ENDPOINT>"
    }
  }' \
  --region <REGION>
```

响应中包含用于集成的 webhook 信息。你可以使用这个 webhook 来触发 Datadog 的调查。有关更多信息，请参阅 the section called “正在连接 DataDog”。

11.（可选）删除代理空间

删除代理空间会删除该代理空间的所有关联、配置和调查数据。此操作无法撤消。

要删除代理空间，请运行以下命令：

```
aws devops-agent delete-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

验证

要验证您的设置，请运行以下命令：

```
# List your agent spaces
aws devops-agent list-agent-spaces \
  --region <REGION>

# Get details of a specific agent space
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>

# List associations for an agent space
aws devops-agent list-associations \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

后续步骤

- 要连接其他集成，请参阅为 AWS DevOps 代理配置功能。
- 要了解代理技能和能力，请参阅the section called “DevOps 特工技能”。
- 要了解操作员 Web 应用程序，请参阅the section called “什么是 DevOps 代理 Web 应用程序？”。

注意

- 将<AGENT_SPACE_ID>、<MONITORING_ACCOUNT_ID><EXTERNAL_ACCOUNT_ID>、<REGION>、等替换为实际值。
- 有关受支持区域的列表，请参阅the section called “支持的区域：”。

创建测试环境

本指南提供动手测试，以使用示例架构验证 AWS DevOps 代理的事件响应功能。如果您想在连接生产系统之前测试 DevOps Agent，请使用此补充内容。

先决条件

- AWS 具有管理权限的账户
- AWS DevOps 使用自动创建代理角色流程创建和配置的 DevOps 代理空间

成本和安全概述

成本保护

- EC2 测试：免费（AWS 免费套餐）或 2 小时约 0.02 美元
- Lambda 测试：免费（100 万 requests/month 免费套餐）
- CloudWatch: 免费（10 个警报，包括基本指标）
- 预计总成本：完整测试费用为 0.00-0.05 美元

这些测试中的安全功能

- 自动终止：内置自动关机
- 符合免费套餐资格：使用最小的实例类型
- 范围有限：极少的隔离测试资源
- 易于清理：只需简单的控制台步骤即可删除所有内容
- 不影响生产：完全独立的测试环境

设置您的 AWS 账户进行测试

> **Important**
>
> 基础设施资源需要部署在您创建 DevOps Agent Space 主云帐户的账户中。 AWS 具体区域无关紧要。

1. 登录 AWS 控制台：https://console.aws.amazon.com
2. 确保您使用的是 DevOps 代理空间所在的同一个 AWS 账户
3. 您可以将任何区域用作测试资源

> **Note**
>
> DevOps 代理的主帐户与您正在创建的测试环境资源之间的 1:1 映射简化了测试设置。您可以轻松地将 DevOps 代理空间扩展到包括次要账户，并启用跨账户调查。

选择你的测试

你可以单独运行任一测试，也可以同时运行两者兼而有之：

测试选项 A：EC2 CPU 容量测试

目的：验证 AWS DevOps 代理检测和调查 EC2 性能问题的能力

预计时间：5 分钟设置 + 10 分钟自动执行

难度：全自动（无需手动步骤）

测试选项 B：Lambda 错误率测试

目的：验证 AWS DevOps 代理检测和调查 Lambda 函数错误的能力

预计时间：10 分钟设置 + 2 分钟触发

难度：非常简单

测试选项 A：EC2 CPU 容量测试

步骤 1：部署用于 EC2 测试的 CloudFormation 堆栈

我们将使用它 CloudFormation 来创建我们的测试资源，这样 AWS DevOps Agent 就可以正确地跟踪和调查它们。

1. 导航至 CloudFormation：

a. 在 AWS 控制台中，搜索 “CloudFormation”，然后单击 CloudFormation

b. 点击创建堆栈 > 使用新资源（标准）

2. 上传模板：

a. 新建一个名为的本地文件 AWS-DevOpsAgent-ec2-test.yaml

b. 将此 CloudFormation 模板复制并粘贴到文件中：

i.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'AWS DevOps Agent EC2 CPU Test Stack'
Parameters:
  MyIP:
    Type: String
    Description: Your current IP address for SSH access (find at https://
whatismyipaddress.com)
    Default: '0.0.0.0/0'
Resources:
  # Security Group for SSH access
  TestSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: AWS-DevOpsAgent-test-sg
      GroupDescription: AWS DevOps Agent beta testing security group
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: !Ref MyIP
          Description: SSH access from your IP
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-SG
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # Key Pair for SSH access
  TestKeyPair:
    Type: AWS::EC2::KeyPair
    Properties:
      KeyName: AWS-DevOpsAgent-test-key
      KeyType: rsa
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Key
        - Key: Purpose
```

```
          Value: AWS-DevOpsAgent-Testing
  # EC2 Instance for CPU testing
  TestInstance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t3.micro
      ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-
kernel-6.1-x86_64}}'
      KeyName: !Ref TestKeyPair
      SecurityGroupIds:
        - !Ref TestSecurityGroup
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          yum update -y
          yum install -y htop

          # Create the CPU stress test script
          cat > /home/ec2-user/cpu-stress-test.sh << 'EOF'
          #!/bin/bash
          echo "Starting AWS DevOpsAgent CPU Stress Test"
          echo "Time: $(date)"
          echo "Instance: $(curl -s http://169.254.169.254/latest/meta-data/
instance-id)"
          echo ""

          # Get number of CPU cores
          CORES=$(nproc)
          echo "CPU Cores: $CORES"
          echo ""

          echo "Starting stress test (5 minutes)..."
          echo "This will generate >70% CPU usage to trigger CloudWatch alarm"
          echo ""

          # Create CPU load using yes command
          echo "Starting CPU load processes..."
          for i in $(seq 1 $CORES); do
              (yes > /dev/null) &
              CPU_PID=$!
              echo "Started CPU load process $i (PID: $CPU_PID)"
              echo $CPU_PID >> /tmp/cpu_test_pids
          done
```

```
          # Auto-cleanup after 5 minutes
          (sleep 300 && echo "Stopping CPU load processes..." && kill $(cat /
tmp/cpu_test_pids 2>/dev/null) 2>/dev/null && rm -f /tmp/cpu_test_pids) &

          echo ""
          echo "CPU load processes started for 5 minutes"
          echo "Check CloudWatch for alarm trigger in 3-5 minutes"
          EOF

          chmod +x /home/ec2-user/cpu-stress-test.sh
          chown ec2-user:ec2-user /home/ec2-user/cpu-stress-test.sh

          # Create auto-shutdown script (safety mechanism)
          cat > /home/ec2-user/auto-shutdown.sh << 'SHUTDOWN_EOF'
          #!/bin/bash
          echo "Auto-shutdown scheduled for 2 hours from now: $(date)"
          sleep 7200
          echo "Auto-shutdown executing at: $(date)"
          sudo shutdown -h now
          SHUTDOWN_EOF

          chmod +x /home/ec2-user/auto-shutdown.sh
          nohup /home/ec2-user/auto-shutdown.sh > /home/ec2-user/auto-
shutdown.log 2>&1 &

          echo "AWS DevOpsAgent test setup completed at $(date)" > /home/ec2-
user/setup-complete.txt
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Instance
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # CloudWatch Alarm for CPU utilization
  CPUAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName: AWS-DevOpsAgent-EC2-CPU-Test
      AlarmDescription: AWS-DevOpsAgent beta test - EC2 CPU utilization alarm
      MetricName: CPUUtilization
      Namespace: AWS/EC2
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
      Threshold: 70
```

```
      ComparisonOperator: GreaterThanThreshold
      Dimensions:
        - Name: InstanceId
          Value: !Ref TestInstance
      TreatMissingData: notBreaching
Outputs:
  InstanceId:
    Description: EC2 Instance ID for testing
    Value: !Ref TestInstance

  SecurityGroupId:
    Description: Security Group ID
    Value: !Ref TestSecurityGroup

  AlarmName:
    Description: CloudWatch Alarm Name
    Value: !Ref CPUAlarm

  SSHCommand:
    Description: SSH command to connect to instance
    Value: !Sub 'ssh -i "AWS-DevOpsAgent-test-key.pem" ec2-user@
${TestInstance.PublicDnsName}'
```

c. 在 CloudFormation 控制台中，选择上传模板文件

d. 单击 “选择文件”

e. 选择AWS-DevOpsAgent-ec2-test.yaml文件

f. 单击下一步

3. 配置堆栈：

a. 堆栈名称：AWS-DevOpsAgent-EC2-Test

b. 参数：

i. MyIP：保留为默认值0.0.0.0/0（如果需要，你可以稍后保护它）

c. 单击下一步

4. 配置堆栈选项：

a. 保留默认值，单击 “下一步”

5. 审核和创建：

a. 勾选我确认 AWS CloudFormation 可能会创建 IAM 资源

b. 点击提交

6. 等待完成：

a. 创建堆栈需要 3-5 分钟

b. 状态将从变CREATE_IN_PROGRESS为 CREATE_COMPLETE

c. 重要：您的 EC2 实例现在是 AWS DevOpsAgent 可以跟踪的 CloudFormation 堆栈的一部分！

可选：安全 SSH 访问（仅当您计划连接到实例时）

如果您只想运行自动测试，请跳过此步骤

1. 导航到 EC2 安全组：
   a. 在 AWS 控制台中，前往 EC2 → 安全组
   b. 查找 AWS-DevOpsAgent-test-sg
2. 更新 SSH 规则：
   a. 选择安全组 → 入站规则选项卡 → 编辑入站规则
   b. 找到 SSH 规则（端口 22）
   c. 将来源从更改0.0.0.0/0为你的 IP：[YOUR_IP]/32
   d. 从中获取您的 IP https://whatismyipaddress.com
   e. 点击保存规则

步骤 2：等待自动执行测试

1. 自动执行测试：
   - CPU 压力测试将在实例启动 5 分钟后自动开始
   - 无需手动干预-只需等待，测试完全在后台运行
2. 监控测试：
   - 实例会自动启动并准备测试
   - 该脚本将运行 5 分钟并生成 > 70% 的 CPU 使用率
   - CloudWatch 警报应在总计 8-10 分钟内触发（延迟 5 分钟 + 警报 3-5 分钟）
3. 可选：手动重新运行（用于其他测试）：
   - 连接到您的实例：EC2 控制台 → Connect **AWS-DevOpsAgent-Test-Instance** → 会话管理器
   - 再次运行 stress 测试：./cpu-stress-test.sh
   - 非常适合多次测试 AWS DevOpsAgent响应

测试选项 B：Lambda 错误率测试

步骤 1：为 Lambda 测试部署 CloudFormation 堆栈

1. 导航至 CloudFormation：
   a. 在 AWS 控制台中，前往 CloudFormation
   b. 点击创建堆栈 → 使用新资源（标准）
2. 上传模板：
   a. 新建一个名为的本地文件 `AWS-DevOpsAgent-lambda-test.yaml`
   b. 将此 CloudFormation 模板复制并粘贴到文件中：
      i.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'AWS DevOpsAgent Lambda Error Test Stack'
Resources:
  # IAM Role for Lambda function
  LambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: AWS-DevOpsAgentLambdaTestRole
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Lambda-Test-Role
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # Lambda function that generates errors
  TestLambdaFunction:
    Type: AWS::Lambda::Function
    Properties:
      FunctionName: AWS-DevOpsAgent-test-lambda
      Runtime: python3.12
      Handler: index.lambda_handler
      Role: !GetAtt LambdaExecutionRole.Arn
```

```
      Code:
        ZipFile: |
          import json
          import random
          import time
          from datetime import datetime
          def lambda_handler(event, context):
              print(f"AWS DevOpsAgent Test Lambda - {datetime.now()}")
              print(f"Event: {json.dumps(event)}")

              # Intentionally generate errors for testing
              error_scenarios = [
                  "Simulated database connection timeout",
                  "Test API rate limit exceeded",
                  "Intentional validation error for AWS DevOpsAgent testing"
              ]

              # Always throw an error for testing purposes
              error_message = random.choice(error_scenarios)
              print(f"Generating test error: {error_message}")

              # This will create a Lambda error that CloudWatch will detect
              raise Exception(f"AWS DevOpsAgent Test Error: {error_message}")
      Description: AWS DevOpsAgent beta test function - intentionally generates
errors
      Timeout: 30
      Tags:
        - Key: Name
          Value: AWS-DevOpsAgent-Test-Lambda
        - Key: Purpose
          Value: AWS-DevOpsAgent-Testing
  # CloudWatch Alarm for Lambda errors
  LambdaErrorAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName: AWS-DevOpsAgent-Lambda-Error-Test
      AlarmDescription: AWS-DevOpsAgent beta test - Lambda error rate alarm
      MetricName: Errors
      Namespace: AWS/Lambda
      Statistic: Sum
      Period: 60
      EvaluationPeriods: 1
      Threshold: 0
      ComparisonOperator: GreaterThanThreshold
```

```
      Dimensions:
        - Name: FunctionName
          Value: !Ref TestLambdaFunction
      TreatMissingData: notBreaching
Outputs:
  LambdaFunctionName:
    Description: Lambda Function Name for testing
    Value: !Ref TestLambdaFunction

  LambdaFunctionArn:
    Description: Lambda Function ARN
    Value: !GetAtt TestLambdaFunction.Arn

  AlarmName:
    Description: CloudWatch Alarm Name
    Value: !Ref LambdaErrorAlarm

  TestCommand:
    Description: AWS CLI command to test the function
    Value: !Sub 'aws lambda invoke --function-name ${TestLambdaFunction} --
payload "{\"test\":\"AWS DevOpsAgent validation\"}" response.json'
```

c. 在 CloudFormation 控制台中，选择上传模板文件

d. 单击 “选择文件”

e. 选择AWS-DevOpsAgent-lambda-test.yaml文件

f. 单击下一步

3. 配置堆栈：

a. 堆栈名称：AWS-DevOpsAgent-Lambda-Test

b. 单击下一步

4. 配置堆栈选项：

a. 保留默认值，单击 “下一步”

5. 审核和创建：

a. 勾选我确认 AWS CloudFormation 可能会创建 IAM 资源

b. 点击提交

6. 等待完成：

a. 创建堆栈需要 2-3 分钟

b. 状态将更改为 CREATE_COMPLETE

步骤 2：触发 Lambda 错误

1. 导航到 Lambda 控制台：
   a. 前往 AWS Lambda 控制台
   b. 找到你的职能 `AWS-DevOpsAgent-test-lambda`
2. 测试函数：
   a. 单击 “测试” 选项卡
   b. 点击创建新活动
   c. 活动名称：`AWS-DevOpsAgent-test-event`
   d. 使用这个 JSON 有效负载：
      i.
```
{
"test": "AWS DevOpsAgent validation",
"timestamp": "2024-01-01T00:00:00Z"
}
```
   e. 单击 “保存”
3. 生成错误：
   a. 单击 “测试” 按钮 3 次（每次之间等待 10 秒）
   b. 每次测试都会产生一个故意的错误
   c. CloudWatch 警报应在 2-3 分钟内触发
   d. AWS DevOpsAgent现在应该能够通过操作员应用程序中的调查来检测警报，接下来你将设置该应用程序。

验证 AWS DevOps 代理检测

步骤 1：Sanity 检查 CloudWatch 警报（可选）

此步骤用于确保上述测试现在处于警报状态。

对于 EC2 测试：

- 在 CloudWatch 控制台中，前往 “警报”
- 开始压力测试后@@ 等待 3-5 分钟
- 您的闹钟应显示为处于警报状态
- 如果仍然 “正常”：再等待 2-3 分钟（CloudWatch 指标可能会延迟）

对于 Lambda 测试：

- 查看AWS-DevOpsAgent-Lambda-Error-Test警报
- 应在运行测试后的 2-3 分钟内显示警报

步骤 2：开始 AWS DevOps 代理调查

1. 打开你的AWS DevOps 代理 AgentSpace
2. 单击 “管理员访问权限”。这将在新窗口中打开 A DevOps gent Space Web 应用程序
3. 点击屏幕右侧的 “开始调查” 按钮
4. 填写以下表格：
   a. 调查详情：描述你想开展的调查。尽可能提供有关调查目标、要探索的领域或相关信息的所有细节。
   b. 调查起点：描述您想要开始调查的信息。您可以提及警报、指标、日志片段或其他任何内容，为 DevOps 代理提供工作起点。在这种情况下，请提供您刚刚创建的警报的摘要。
   c. 事件发生日期和时间（首选 ISO 8601）:: MMZ YYYY-MM-DDTHH
   d. 命名您的调查：示例：Oncall_investigation_1:2025-10-27
   e. AWS 事件的@@ 账户 ID
   f. 事件发生的地@@ 区
   g. 优先级- AWS DevOpsAgent 允许同时进行两次调查。优先级允许您定义调查的执行顺序。
5. 单击 “调查” 启动调查。
6. 点击控制面板中列出的您的调查。您将被带到 “调查详情” 屏幕，您可以在其中查看 DevOps 特工正在采取的详细步骤。

预期结果

EC2 测试结果：

- 检测 EC2 CPU 警报
- 确定根本原因：“CPU 压力测试工作负载”
- 显示时间轴：压力测试 → CPU 峰值 → 警报
- 提供监控和扩展建议

Lambda 测试结果：

- 检测 Lambda 错误率峰值
- 确定根本原因："故意测试异常"
- 显示时间轴：函数调用 → 错误 → 警报
- 为错误处理和监控提供建议

清理说明

清理测试 A（EC2 测试）

自动清理

- 实例将在 2 小时后自动终止（内置于 CloudFormation 模板中）

手动清理（立即）

1. 删除 CloudFormation 堆栈：
   a. 前往 CloudFormation 控制台
   b. 选择`AWS-DevOpsAgent-EC2-Test`堆栈
   c. 单击 "删除"
   d. 确认删除
   e. 这将自动删除所有资源：EC2 实例、安全组、key pair 和 CloudWatch 警报

清理测试 B（Lambda 测试）

1. 删除 CloudFormation 堆栈：
   a. 前往 CloudFormation 控制台
   b. 选择`AWS-DevOpsAgent-Lambda-Test`堆栈
   c. 单击 "删除"
   d. 确认删除
   e. 这将自动删除所有资源：Lambda 函数、IAM 角色和警报 CloudWatch

问题排查

常见问题

“无法连接到 EC2 实例”

- 检查安全组：确保您的 IP 已打开 SSH（端口 22）
- 检查密钥权限：运行 `chmod 400 AWS-DevOpsAgent-test-key.pem`
- 验证公有 IP：必须为实例分配公有 IP
- 等待实例：确保实例处于 “正在运行” 状态

“警报未触发”

- 等待指标： CloudWatch 指标可能需要 2-5 分钟才能显示
- 检查 CPU 负载：通过 SSH 连接到实例并运行`top`以验证 CPU 大于 70%
- 验证压力测试：运行`ps aux | grep yes`以查看加载进程是否正在运行
- 延长等待：有时首次触发警报最多需要 7-8 分钟

测试验证

在以下情况下，您的 AWS DevOp 代理测试成功：

技术验证

- 调查精度：EC2 测试的结果应正确表明警报是由于 CPU 负载而触发的。Lambda 测试的结果应表明这是故意失败。
- 时间轴精度：显示的事件顺序正确
- 推荐质量：提供可行的建议

使用 AWS CDK 开始使用 AWS DevOps 代理

概述

本指南向您展示如何使用 C AWS loud Development Kit (AWS CDK) 创建和部署 AWS DevOps 代理资源。 AWS CDK 应用程序通过自动创建代理空间、 AWS 身份和访问管理 (IAM) 角色、操作员应用程序和账户关联。 AWS AWS CloudFormation

AWS CDK 方法通过将所有必需的资源定义为基础架构即代码，从而自动执行 CLI 入门指南中描述的手动步骤。

AWS DevOps 代理可在以下 6 个 AWS 区域使用：美国东部（弗吉尼亚北部）、美国西部（俄勒冈）、亚太地区（悉尼）、亚太地区（东京）、欧洲（法兰克福）和欧洲（爱尔兰）。有关支持的区域的更多信息，请参阅the section called “支持的区域：”。

先决条件

开始之前，请确保您拥有：

- AWS 已安装命令行接口 (AWS CLI)，并使用相应的凭据进行配置
- Node.js 版本 18 或更高版本
- AWS 全局安装了 CDK 命令行界面 (CLI)。要安装 AWS CDK CLI，请运行以下命令：

```
npm install -g aws-cdk
```

- 一个 AWS 用于监控（主要）账户的账户
- （可选）如果您想设置跨 AWS 账户监控，则需要第二个账户

本指南涵盖的内容

本指南分为两部分：

- 第 1 部分 — 在您的监控账户中部署带有操作员应用程序和 AWS 关联的代理空间。完成本部分后，代理可以监控该账户中的问题。
- 第 2 部分（可选）— 为服务账户添加源 AWS 关联，并将跨账户 IAM 角色部署到该账户。此配置使代理空间能够跨账户监控资源。

创建的资源

第 1 部分： DevOpsAgentStack （监控账户）

- IAM 角色 (`DevOpsAgentRole-AgentSpace`)-由 DevOps 代理服务担任以监控账户。包括`AIDevOpsAgentAccessPolicy`托管策略和允许创建资源管理器服务相关角色的内联策略。

- IAM 角色 (DevOpsAgentRole-WebappAdmin)-操作员应用程序角色，具有代理操作的AIDevOpsOperatorAppAccessPolicy托管策略。
- 代理空间 (MyCDKAgentSpace)-使用AWS::DevOpsAgent::AgentSpace CloudFormation 资源创建的中央代理空间。包括操作员应用程序配置。
- 关联（AWS 监视器）-使用AWS::DevOpsAgent::Association CloudFormation 资源将监控账户链接到代理空间。
- 关联（AWS 来源）-（可选）将服务帐户链接到代理空间以进行跨账户监控。

第 2 部分： ServiceStack （服务帐号，可选）

- IAM 角色 (DevOpsAgentRole-SecondaryAccount) — 具有固定名称的跨账户角色。受监控账户中代理空间的信任。包括AIDevOpsAgentAccessPolicy托管策略和允许创建资源管理器服务相关角色的内联策略。
- Lambda 函数 (echo-service) — 一个回显输入事件的简单示例服务。

设置

步骤 1：克隆示例存储库

运行以下命令来克隆存储库并切换到项目目录：

```
git clone https://github.com/aws-samples/sample-aws-devops-agent-cdk.git
cd sample-aws-devops-agent-cdk
```

第 2 步：安装依赖关系

运行以下命令来安装项目依赖项：

```
npm install
```

第 1 部分：部署代理空间

在本节中，您将在您的监控账户中创建代理空间、IAM 角色、操作员应用程序和 AWS 关联。

步骤 1：配置监控账户 ID

打开lib/constants.ts并设置您的监控账户 ID：

以下示例显示了要更新的常量：

```
export const MONITORING_ACCOUNT_ID = "<YOUR_MONITORING_ACCOUNT_ID>";
```

第 2 步：引导 AWS CDK 环境

如果您尚未在监控账户中引导 AWS CDK，请运行以下命令：

```
cdk bootstrap aws://<MONITORING_ACCOUNT_ID>/<REGION> --profile monitoring
```

步骤 3：构建和部署

运行以下命令来构建 TypeScript 代码并部署堆栈：

```
npm run build
cdk deploy DevOpsAgentStack --profile monitoring
```

第 4 步：记录堆栈输出

部署完成后， AWS CDK 会打印堆栈输出。记录这些值以备后用。

以下示例显示了预期的输出：

```
Outputs:
DevOpsAgentStack.AgentSpaceArn = arn:aws:aidevops:<REGION>:123456789012:agentspace/
abc123
DevOpsAgentStack.AgentSpaceRoleArn = arn:aws:iam::123456789012:role/DevOpsAgentRole-
AgentSpace
DevOpsAgentStack.OperatorRoleArn = arn:aws:iam::123456789012:role/DevOpsAgentRole-
WebappAdmin
DevOpsAgentStack.AssociationId = assoc-xyz
```

如果您计划完成第 2 部分，请保存该AgentSpaceArn值。您需要它来配置服务帐号堆栈。

步骤 5：验证部署

要验证代理空间是否已成功创建，请运行以下 AWS CLI 命令：

```
aws devopsagent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

此时，您的代理空间已在启用操作员应用程序并关联您的监控账户的情况下进行部署。代理可以监控此账户中的问题。

第 2 部分（可选）：添加跨账户监控

在本节中，您将扩展设置，以便您的代理空间可以监控第二个 AWS 帐户（服务帐户）中的资源。这涉及两个操作：

1. 在中添加指向服务帐号的源 AWS 关联。 DevOpsAgentStack
2. 使用信任代理空间的 IAM 角色将部署 ServiceStack 到服务账户。

> **Important**
>
> 必须先完成第 1 部分，然后才能继续。 ServiceStack 需要 DevOpsAgentStack 部署输出`AgentSpaceArn`中的。

步骤 1：配置服务帐号 ID

打开`lib/constants.ts`并设置您的服务帐号 ID：

以下示例显示了要更新的常量：

```
export const SERVICE_ACCOUNT_ID = "<YOUR_SERVICE_ACCOUNT_ID>";
```

使用此账户 ID DevOpsAgentStack 创建源 AWS 关联。如果您 DevOpsAgentStack 在设置此值之前部署了，请重新部署以创建关联：

运行以下命令进行重新部署：

```
npm run build
cdk deploy DevOpsAgentStack --profile monitoring
```

步骤 2：设置代理空间 ARN

复制 DevOpsAgentStack 输出中的`AgentSpaceArn`值（第 1 部分，步骤 4）并将其设置为`lib/constants.ts`：

以下示例显示了要更新的常量：

```
export const AGENT_SPACE_ARN =
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/<SPACE_ID>";
```

ServiceStack 使用此值来确定次要账户角色的信任策略范围。 ServiceStack 只有在设置此值时才会合成。

第 3 步：引导服务账号

如果您尚未在服务帐号中引导 AWS CDK，请运行以下命令：

```
cdk bootstrap aws://<SERVICE_ACCOUNT_ID>/<REGION> --profile service
```

步骤 4：部署 ServiceStack

运行以下命令，使用服务帐号 ServiceStack 的凭据构建和部署：

```
npm run build
cdk deploy ServiceStack --profile service
```

这将在服务帐号中创建以下资源：

- 信任监控账户中代理空间的 IAM 角色 (`DevOpsAgentRole-SecondaryAccount`)
- 将 echo Lambda 函数 (`echo-service`) 作为示例服务

步骤 5：验证部署

要确认 Lambda 函数已成功部署，请运行以下命令来测试 echo 服务：

```
aws lambda invoke \
  --function-name echo-service \
  --payload '{"test": "hello world"}' \
  --profile service \
  response.json
cat response.json
```

问题排查

本节介绍常见问题以及如何解决这些问题。

CloudFormation 未找到资源类型

- 确认您正在中进行部署the section called “支持的区域：”。
- 确认您的 AWS CLI 配置了相应的权限。

IAM 角色创建失败

- 验证您的部署角色是否有权创建 IAM 角色。
- 检查信任政策条件是否与您的账户 ID 相符。

跨账户部署失败，并显示 “无法在目标账户中扮演角色”

- 每个堆栈都必须使用目标账户的凭据进行部署。使用该`--profile`标志指定正确的 AWS CLI 配置文件。
- 验证 AWS CDK 是否已在目标账户中被引导。

IAM 传播延迟

- IAM 角色更改可能需要几分钟才能传播。如果创建角色后立即创建代理空间失败，请等待几分钟并重新部署。

清理

要移除所有资源，请按相反的顺序销毁堆栈。

运行以下命令来销毁堆栈：

```
# If you deployed the ServiceStack, destroy it first
cdk destroy ServiceStack --profile service
# Then destroy the DevOpsAgentStack
cdk destroy DevOpsAgentStack --profile monitoring
```

警告：此操作将永久删除您的代理空间和所有相关数据。此操作无法撤消。在继续操作之前，请确保已备份所有重要信息。

安全注意事项

- AWS CDK 应用程序使用信任策略创建 IAM 角色，该策略仅允许`aidevops.amazonaws.com`服务委托人代入这些角色。

- 信任策略包括限制访问您的特定 AWS 账户和代理空间 ARN 的条件。
- 所有策略都遵循最低权限原则。根据贵组织的安全要求查看和自定义 IAM 策略。
- 跨账户角色 (`DevOpsAgentRole-SecondaryAccount`) 使用固定名称，其作用域仅限于特定的代理空间 ARN。

后续步骤

使用 AWS CDK 部署 AWS DevOps 代理后：

1. 在《 DevOps 代理用户指南》中了解AWS DevOps 代理的全部功能。
2. 考虑将 AWS CDK 部署集成到您的 CI/CD 管道中，以实现自动化基础架构管理。

其他资源

- AWS DevOps 代理用户指南
- 网站上的 CDK 存储库示例 GitHub
- CLI 入门指南

开始使用 AWS DevOps 代理 AWS CloudFormation

概述

本指南向您介绍如何使用 AWS CloudFormation 模板来创建和部署 AWS DevOps 代理资源。这些模板可以自动创建代理空间、Ident AWS ity and Access Management 角色、操作员应用程序 AWS 和账户关联作为基础架构即代码。

该 CloudFormation 方法通过在声明式 YAML 模板中定义所有必需的资源，自动执行 CLI 入门指南中描述的手动步骤。

AWS DevOps 代理可在以下 6 个 AWS 区域使用：美国东部（弗吉尼亚北部）、美国西部（俄勒冈）、亚太地区（悉尼）、亚太地区（东京）、欧洲（法兰克福）和欧洲（爱尔兰）。有关支持的区域的更多信息，请参阅the section called “支持的区域：”。

先决条件

开始之前，请确保您拥有：

- AWS 已安装命令行接口 (AWS CLI)，并使用相应的凭据进行配置
- 创建 IAM 角色和 CloudFormation 堆栈的权限
- 一个 AWS 用于监控（主要）账户的账户
- （可选）如果您想设置跨 AWS 账户监控，则需要第二个账户

本指南涵盖的内容

本指南分为两部分：

- 第 1 部分 — 在您的监控账户中部署带有操作员应用程序和 AWS 关联的代理空间。完成本部分后，代理可以监控该账户中的问题。
- 第 2 部分（可选）— 将跨账户 IAM 角色部署到辅助账户并添加源 AWS 关联。此配置使代理空间能够跨账户监控资源。

第 1 部分：部署代理空间

在本节中，您将创建一个 CloudFormation 模板，用于在您的监控账户中配置代理空间、IAM 角色、操作员应用程序和 AWS 关联。

步骤 1：创建 CloudFormation 模板

将以下模板另存为`devops-agent-stack.yaml`：

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Agent Space with IAM roles, operator app, and AWS
 association

Parameters:
  AgentSpaceName:
    Type: String
    Default: MyCloudFormationAgentSpace
    Description: Name for the agent space
  AgentSpaceDescription:
    Type: String
    Default: Agent space deployed with CloudFormation
    Description: Description for the agent space

Resources:
  # IAM role assumed by the DevOps Agent service to monitor the account
```

```
  DevOpsAgentSpaceRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-AgentSpace
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action: sts:AssumeRole
            Condition:
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
              ArnLike:
                aws:SourceArn: !Sub arn:aws:aidevops:${AWS::Region}:
${AWS::AccountId}:agentspace/*
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
      Policies:
        - PolicyName: AllowCreateServiceLinkedRoles
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Sid: AllowCreateServiceLinkedRoles
                Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource:
                  - !Sub arn:aws:iam::${AWS::AccountId}:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer

  # IAM role for the operator app interface
  DevOpsOperatorRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-WebappAdmin
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action:
```

```
              - sts:AssumeRole
              - sts:TagSession
            Condition:
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
              ArnLike:
                aws:SourceArn: !Sub arn:aws:aidevops:${AWS::Region}:
${AWS::AccountId}:agentspace/*
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsOperatorAppAccessPolicy

  # The agent space resource
  AgentSpace:
    Type: AWS::DevOpsAgent::AgentSpace
    DependsOn:
      - DevOpsAgentSpaceRole
      - DevOpsOperatorRole
    Properties:
      Name: !Ref AgentSpaceName
      Description: !Ref AgentSpaceDescription
      OperatorApp:
        Iam:
          OperatorAppRoleArn: !GetAtt DevOpsOperatorRole.Arn

  # Association linking the monitoring account to the agent space
  MonitorAssociation:
    Type: AWS::DevOpsAgent::Association
    Properties:
      AgentSpaceId: !GetAtt AgentSpace.AgentSpaceId
      ServiceId: aws
      Configuration:
        Aws:
          AssumableRoleArn: !GetAtt DevOpsAgentSpaceRole.Arn
          AccountId: !Ref AWS::AccountId
          AccountType: monitor

Outputs:
  AgentSpaceId:
    Description: The agent space ID
    Value: !GetAtt AgentSpace.AgentSpaceId
  AgentSpaceArn:
    Description: The agent space ARN
    Value: !GetAtt AgentSpace.Arn
  AgentSpaceRoleArn:
```

```
    Description: The agent space IAM role ARN
    Value: !GetAtt DevOpsAgentSpaceRole.Arn
  OperatorRoleArn:
    Description: The operator app IAM role ARN
    Value: !GetAtt DevOpsOperatorRole.Arn
```

步骤 2：部署堆栈

运行以下命令部署堆栈。<REGION>替换为the section called “支持的区域：”（例如，us-east-1）。

```
aws cloudformation deploy \
  --template-file devops-agent-stack.yaml \
  --stack-name DevOpsAgentStack \
  --capabilities CAPABILITY_NAMED_IAM \
  --region <REGION>
```

步骤 3：记录堆栈输出

部署完成后，运行以下命令以检索堆栈输出。记录这些值以备后用。

```
aws cloudformation describe-stacks \
  --stack-name DevOpsAgentStack \
  --query 'Stacks[0].Outputs' \
  --region <REGION>
```

以下示例显示了预期的输出：

```
[
  {
    "OutputKey": "AgentSpaceId",
    "OutputValue": "abc123def456"
  },
  {
    "OutputKey": "AgentSpaceArn",
    "OutputValue": "arn:aws:aidevops:<REGION>:<ACCOUNT_ID>:agentspace/abc123def456"
  },
  {
    "OutputKey": "AgentSpaceRoleArn",
    "OutputValue": "arn:aws:iam::<ACCOUNT_ID>:role/DevOpsAgentRole-AgentSpace"
  },
  {
    "OutputKey": "OperatorRoleArn",
```

```
    "OutputValue": "arn:aws:iam::<ACCOUNT_ID>:role/DevOpsAgentRole-WebappAdmin"
  }
]
```

如果您计划完成第 2 部分，请保存该AgentSpaceArn值。您需要它来配置跨账户角色。

步骤 4：验证部署

要验证代理空间是否已成功创建，请运行以下 AWS CLI 命令：

```
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

此时，您的代理空间已在启用操作员应用程序并关联您的监控账户的情况下进行部署。代理可以监控此账户中的问题。

第 2 部分（可选）：添加跨账户监控

在本节中，您将扩展设置，以便您的代理空间可以监控第二个 AWS 帐户（服务帐户）中的资源。这涉及两个操作：

1. 在信任代理空间的服务账户中部署 IAM 角色。
2. 在监控账户中添加指向服务帐号的源 AWS 关联。

> 注意：必须先完成第 1 部分，然后才能继续。服务账户模板需要第 1 部分堆栈AgentSpaceArn的输出。

步骤 1：创建服务账号模板

将以下模板另存为devops-agent-service-account.yaml。此模板在辅助账户中创建跨账户 IAM 角色。

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Cross-account IAM role for secondary account monitoring

Parameters:
  MonitoringAccountId:
    Type: String
    Description: The 12-digit AWS account ID of the monitoring account
  AgentSpaceArn:
```

```
    Type: String
    Description: The ARN of the agent space from the monitoring account

Resources:
  # Cross-account IAM role trusted by the agent space
  DevOpsSecondaryAccountRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: DevOpsAgentRole-SecondaryAccount
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: aidevops.amazonaws.com
            Action: sts:AssumeRole
            Condition:
              StringEquals:
                aws:SourceAccount: !Ref MonitoringAccountId
              ArnLike:
                aws:SourceArn: !Ref AgentSpaceArn
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AIDevOpsAgentAccessPolicy
      Policies:
        - PolicyName: AllowCreateServiceLinkedRoles
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Sid: AllowCreateServiceLinkedRoles
                Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource:
                  - !Sub arn:aws:iam::${AWS::AccountId}:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer

Outputs:
  SecondaryAccountRoleArn:
    Description: The cross-account IAM role ARN
    Value: !GetAtt DevOpsSecondaryAccountRole.Arn
```

步骤 2：部署服务账号堆栈

使用服务帐号的凭据，运行以下命令：

```
aws cloudformation deploy \
  --template-file devops-agent-service-account.yaml \
  --stack-name DevOpsAgentServiceAccountStack \
  --capabilities CAPABILITY_NAMED_IAM \
  --parameter-overrides \
    MonitoringAccountId=<MONITORING_ACCOUNT_ID> \
    AgentSpaceArn=<AGENT_SPACE_ARN> \
  --region <REGION>
```

步骤 3：添加源 AWS 关联

切换回监控账户并创建源 AWS 关联。您可以通过创建单独的堆栈或更新原始模板来实现此目的。以下示例使用独立模板。

将以下模板另存为`devops-agent-source-association.yaml`：

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS DevOps Agent - Source AWS association for cross-account monitoring

Parameters:
  AgentSpaceId:
    Type: String
    Description: The agent space ID from the monitoring account stack
  ServiceAccountId:
    Type: String
    Description: The 12-digit AWS account ID of the service account
  ServiceAccountRoleArn:
    Type: String
    Description: The ARN of the DevOpsAgentRole-SecondaryAccount role in the service
 account

Resources:
  SourceAssociation:
    Type: AWS::DevOpsAgent::Association
    Properties:
      AgentSpaceId: !Ref AgentSpaceId
      ServiceId: aws
      Configuration:
        SourceAws:
```

```
          AccountId: !Ref ServiceAccountId
          AccountType: source
          AssumableRoleArn: !Ref ServiceAccountRoleArn

Outputs:
  SourceAssociationId:
    Description: The source association ID
    Value: !Ref SourceAssociation
```

使用监控账户凭证部署关联堆栈：

```
aws cloudformation deploy \
  --template-file devops-agent-source-association.yaml \
  --stack-name DevOpsAgentSourceAssociationStack \
  --parameter-overrides \
    AgentSpaceId=<AGENT_SPACE_ID> \
    ServiceAccountId=<SERVICE_ACCOUNT_ID> \
    ServiceAccountRoleArn=arn:aws:iam::<SERVICE_ACCOUNT_ID>:role/DevOpsAgentRole-
SecondaryAccount \
  --region <REGION>
```

验证

通过运行以下 AWS CLI 命令来验证您的设置：

```
# List your agent spaces
aws devops-agent list-agent-spaces \
  --region <REGION>

# Get details of a specific agent space
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>

# List associations for an agent space
aws devops-agent list-associations \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

问题排查

本节介绍常见问题以及如何解决这些问题。

CloudFormation 未找到资源类型

- 确认您正在中进行部署the section called “支持的区域：”。
- 确认您的 AWS CLI 配置了相应的权限。

IAM 角色创建失败

- 验证您的部署凭证是否有权使用自定义名称创建 IAM 角色 (CAPABILITY_NAMED_IAM)。
- 检查信任政策条件是否与您的账户 ID 相符。

跨账户部署失败

- 每个堆栈都必须使用目标账户的凭据进行部署。使用该`--profile`标志指定正确的 AWS CLI 配置文件。
- 验证`AgentSpaceArn`参数是否与第 1 部分堆栈输出中的精确 ARN 相匹配。

IAM 传播延迟

- IAM 角色更改可能需要几分钟才能传播。如果创建角色后立即创建代理空间失败，请等待几分钟并重新部署。

清理

要移除所有资源，请按相反顺序删除堆栈。

警告：此操作将永久删除您的代理空间和所有相关数据。此操作无法撤消。在继续操作之前，请确保已备份所有重要信息。

运行以下命令删除堆栈：

```
# If you deployed the source association stack, delete it first
aws cloudformation delete-stack \
  --stack-name DevOpsAgentSourceAssociationStack \
  --region <REGION>

aws cloudformation wait stack-delete-complete \
  --stack-name DevOpsAgentSourceAssociationStack \
  --region <REGION>
```

```
# If you deployed the service account stack, delete it next (using service account
 credentials)
aws cloudformation delete-stack \
  --stack-name DevOpsAgentServiceAccountStack \
  --region <REGION>

aws cloudformation wait stack-delete-complete \
  --stack-name DevOpsAgentServiceAccountStack \
  --region <REGION>

# Delete the main stack last
aws cloudformation delete-stack \
  --stack-name DevOpsAgentStack \
  --region <REGION>
```

后续步骤

使用 AWS CloudFormation以下方法部署 AWS DevOps 代理后：

- 要连接其他集成，请参阅为 AWS DevOps 代理配置功能。
- 要了解代理技能和能力，请参阅the section called “DevOps 特工技能”。
- 要了解操作员 Web 应用程序，请参阅the section called “什么是 DevOps 代理 Web 应用程序？”。

使用 Terraform 开始使用 AWS DevOps Agent

概述

本指南向您展示如何使用 Terraform 创建和部署 AWS DevOps 代理资源。Terraform 配置可自动创建代理空间、IAM 角色、操作员应用程序和 AWS 账户关联。

Terraform 方法通过将所有必需的资源定义为基础架构即代码，自动执行 CLI 入门指南中描述的手动步骤。

AWS DevOps 代理可在以下 6 个 AWS 区域使用：美国东部（弗吉尼亚北部）、美国西部（俄勒冈）、亚太地区（悉尼）、亚太地区（东京）、欧洲（法兰克福）和欧洲（爱尔兰）。有关支持的区域的更多信息，请参阅the section called “支持的区域：”。

先决条件

在开始之前，请确保您具有以下各项：

- Terraform >= 1.0 已安装
- AWS 已安装 CLI 并使用相应的凭据进行配置
- 一个 AWS 用于监控（主要）账户的账户
- （可选）如果您想设置跨 AWS 账户监控，则需要第二个账户

本指南涵盖的内容

本指南分为两部分：

- 第 1 部分 — 在您的监控账户中部署带有操作员应用程序和 AWS 关联的代理空间。完成本部分后，代理可以监控该账户中的问题。
- 第 2 部分（可选）— 为服务账户添加源 AWS 关联，并将跨账户 IAM 角色和 echo Lambda 部署到该账户。这允许代理空间监控跨账户的资源。

创建的资源

第 1 部分：监控账户

- IAM 角色 (DevOpsAgentRole-AgentSpace-*)-由 DevOps 代理服务担任以监控账户。包括AIDevOpsAgentAccessPolicy托管策略和允许创建资源管理器服务相关角色的内联策略。
- IAM 角色 (DevOpsAgentRole-WebappAdmin-*)-操作员应用程序角色，具有代理操作的AIDevOpsOperatorAppAccessPolicy托管策略。
- 代理空间（可配置名称）-使用awscc_devopsagent_agent_space资源创建的中央代理空间。包括操作员应用程序配置。
- 关联（AWS 监视器）-使用awscc_devopsagent_association资源将监控账户链接到代理空间。
- 关联（AWS 来源）-（可选）将服务帐户链接到代理空间以进行跨账户监控。

第 2 部分：服务帐号（可选）

- IAM 角色 (DevOpsAgentRole-SecondaryAccount-TF)-具有固定名称的跨账户角色。受监控账户中代理空间的信任。包括AIDevOpsAgentAccessPolicy托管策略和允许创建资源管理器服务相关角色的内联策略。
- Lambda 函数 (echo-service-tf) — 一个回显输入事件的简单示例服务。

设置

步骤 1：克隆示例存储库

```
git clone https://github.com/aws-samples/sample-aws-devops-agent-terraform.git
cd sample-aws-devops-agent-terraform
```

步骤 2：配置变量

复制示例变量文件并根据您的环境对其进行自定义：

```
cp terraform.tfvars.example terraform.tfvars
```

`terraform.tfvars`使用您的代理空间名称和描述进行编辑：

```
agent_space_name        = "MyCompanyAgentSpace"
agent_space_description = "DevOps Agent Space for monitoring production workloads"
```

第 1 部分：部署代理空间

在本节中，您将在您的监控账户中创建代理空间、IAM 角色、操作员应用程序和 AWS 关联。

步骤 1：自动化部署（推荐）

使用提供的部署脚本来简化设置：

```
./deploy.sh
```

此脚本自动：

- 检查先决条件（Terraform、CL AWS I、证书）
- 如果需要`terraform.tfvars`，可以从示例中创建
- 初始化、验证、计划和应用 Terraform

或者，如果您更喜欢手动控制：

```
terraform init
terraform plan
```

```
terraform apply
```

yes当系统提示您确认部署时键入。

第 2 步：记录输出

部署完成后，Terraform 会打印输出。记录以下值以备后用：

```
Outputs:
agent_space_id              = "abc123"
agent_space_arn             =
 "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/abc123"
agent_space_name            = "MyCompanyAgentSpace"
devops_agentspace_role_arn  = "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/
DevOpsAgentRole-AgentSpace-a1b2c3d4"
devops_operator_role_arn    = "arn:aws:iam::<MONITORING_ACCOUNT_ID>:role/
DevOpsAgentRole-WebappAdmin-a1b2c3d4"
primary_account_id          = "<MONITORING_ACCOUNT_ID>"
primary_account_association_id = "assoc-xyz"
```

如果您计划完成第 2 部分，请保存该agent_space_arn值。您将需要它来配置服务帐号资源。

步骤 3：验证部署

运行部署后验证脚本：

```
./post-deploy.sh
```

或者使用 C AWS LI 验证代理空间是否已成功创建：

```
aws devops-agent get-agent-space \
  --agent-space-id <AGENT_SPACE_ID> \
  --region <REGION>
```

此时，您的代理空间已在启用操作员应用程序并关联您的监控账户的情况下进行部署。代理可以监控此账户中的问题。

第 2 部分（可选）：添加跨账户监控

在本节中，您将扩展设置，以便代理空间可以监控第二个 AWS 帐户（服务帐户）中的资源。这涉及两个操作：

1. 添加指向服务帐号的源 AWS 关联。
2. 将跨账户 IAM 角色和 echo Lambda 函数部署到服务账户。

> **Important**
>
> 在继续操作之前，必须完成第 1 部分。服务账户资源需要第 agent_space_arn 1 部分部署输出中的。

步骤 1：配置服务帐号 ID

在中terraform.tfvars，设置您的服务帐号 ID：

```
service_account_id = "<YOUR_SERVICE_ACCOUNT_ID>"
```

步骤 2：设置代理空间 ARN

复制第 1 部分输出（步骤 2）中的agent_space_arn值并将其设置为terraform.tfvars：

```
agent_space_arn = "arn:aws:aidevops:<REGION>:<MONITORING_ACCOUNT_ID>:agentspace/
<SPACE_ID>"
```

服务帐号资源使用此值来确定次要账户角色的信任策略范围。这些资源只有在设置此值时才会创建。

第 3 步：配置 “aws.service” 提供商

在中main.tf，使用服务帐号的凭据配置aws.service提供商别名。您可以使用命名配置文件或代入角色：

使用个人资料：

```
provider "aws" {
  alias   = "service"
  region  = var.aws_region
  profile = "your-service-account-profile"
}
```

或者使用代入角色：

```
provider "aws" {
  alias  = "service"
  region = var.aws_region
  assume_role {
    role_arn = "arn:aws:iam::<SERVICE_ACCOUNT_ID>:role/OrganizationAccountAccessRole"
  }
}
```

步骤 4：部署

应用更新的配置：

```
terraform apply
```

这将在服务帐号中创建以下资源：

- 信任监控账户中代理空间的 IAM 角色 (DevOpsAgentRole-SecondaryAccount-TF)
- 将 echo Lambda 函数 (echo-service-tf) 作为示例服务

它还会在监控账户中创建 AWS 关联服务帐号的源关联。

步骤 5：验证部署

测试回显服务以确认 Lambda 函数已成功部署：

```
aws lambda invoke \
  --function-name echo-service-tf \
  --payload '{"test": "hello world"}' \
  --profile <your-service-account-profile> \
  --region <REGION> \
  response.json
cat response.json
```

问题排查

IAM 传播延迟

- 该配置包括在 IAM 角色创建和代理空间创建time_sleep之间有 30 秒的间隔。 DevOps 代理服务会在创建代理空间期间验证操作员角色的信任策略，如果 IAM 尚未完全传播，则可能会失败。如果

您仍然看到信任策略错误，请稍等片刻然后`terraform apply`再次运行 — IAM 角色将已经存在，应用程序将从上次停下来的地方继续运行。

权限错误

- 验证您的 AWS 证书是否具有创建角色和策略所必需的 IAM 权限。
- 检查信任政策条件是否与您的账户 ID 相符。

跨账户部署失败

- 必须为`aws.service`提供商配置服务帐户的凭据。使用命名配置文件或代入角色块。
- 验证该`agent_space_arn`值是否与第 1 部分输出中的 ARN 相匹配。

未找到 Terraform 资源类型

- 确认您的`awscc`提供程序版本`~> 1.0`或更高版本。`awscc_devopsagent_agent_space`和`awscc_devopsagent_association`资源需要 AWS 云控制提供商。

清理

如果您部署了第 2 部分，则要移除所有资源，请按相反的顺序销毁：

```
./cleanup.sh
```

或者手动：

```
terraform destroy
```

警告：这将永久删除您的代理空间和所有相关数据。在继续操作之前，请确保已备份所有重要信息。

安全注意事项

- Terraform 配置使用信任策略创建 IAM 角色，该策略仅允许`aidevops.amazonaws.com`服务委托人代入这些角色。
- 信任策略包括限制访问您的特定 AWS 账户和代理空间 ARN 的条件。

- 所有策略都遵循最低权限原则。根据贵组织的安全要求查看和自定义 IAM 策略。
- 跨账户角色 (`DevOpsAgentRole-SecondaryAccount-TF`) 使用固定名称，其作用域仅限于特定的代理空间 ARN。

后续步骤

使用 Terraform 部署 AWS DevOps 代理之后：

1. 在《 DevOps 代理用户指南》中了解AWS DevOps 代理的全部功能。
2. 考虑将 Terraform 部署集成到您的 CI/CD 管道中，以实现自动化基础架构管理。

其他资源

- AWS DevOps 代理用户指南
- 示例 Terraform 存储库
- CLI 入门指南

与 DevOps 代理合作

与 DevOps 代理合作

AWS DevOps 从检测到调查、恢复和预防，代理在整个事件生命周期中与您的运营团队合作。以下主题介绍如何使用 DevOps 代理管理此生命周期的每个阶段。

自主事件响应

当检测到事件时，无论是通过与票务系统的内置集成、监控工具中的 webhook 还是手动触发器，DevOps 代理都会自动开始调查。代理会分析指标、日志、跟踪、代码更改和部署历史记录，以确定根本原因并提出缓解计划。如果您需要其他帮助，可以直接通过 A DevOps gent Space 网络应用程序升级到 AWS Support，该应用程序会自动与支持工程师共享调查背景，因此您不必重复代理已经发现的内容。有关更多信息，请参阅 the section called “自主事件响应”。

按需 DevOps 任务

在事件生命周期中的任何时候，您都可以通过对话聊天界面与 DevOps 代理互动。使用自然语言询问有关您的 AWS 资源、系统运行状况、警报状态和部署历史记录的问题。聊天具有情境感知功能 — 当你查看特定调查时，你可以引导代理探索特定的假设，关注特定的日志，或者更新其根本原因分析。您还可以查询整个环境中的资源配置、错误趋势和调查见解，而无需在控制台之间切换。有关更多信息，请参阅 the section called “按需 DevOps 任务”。

主动预防事故

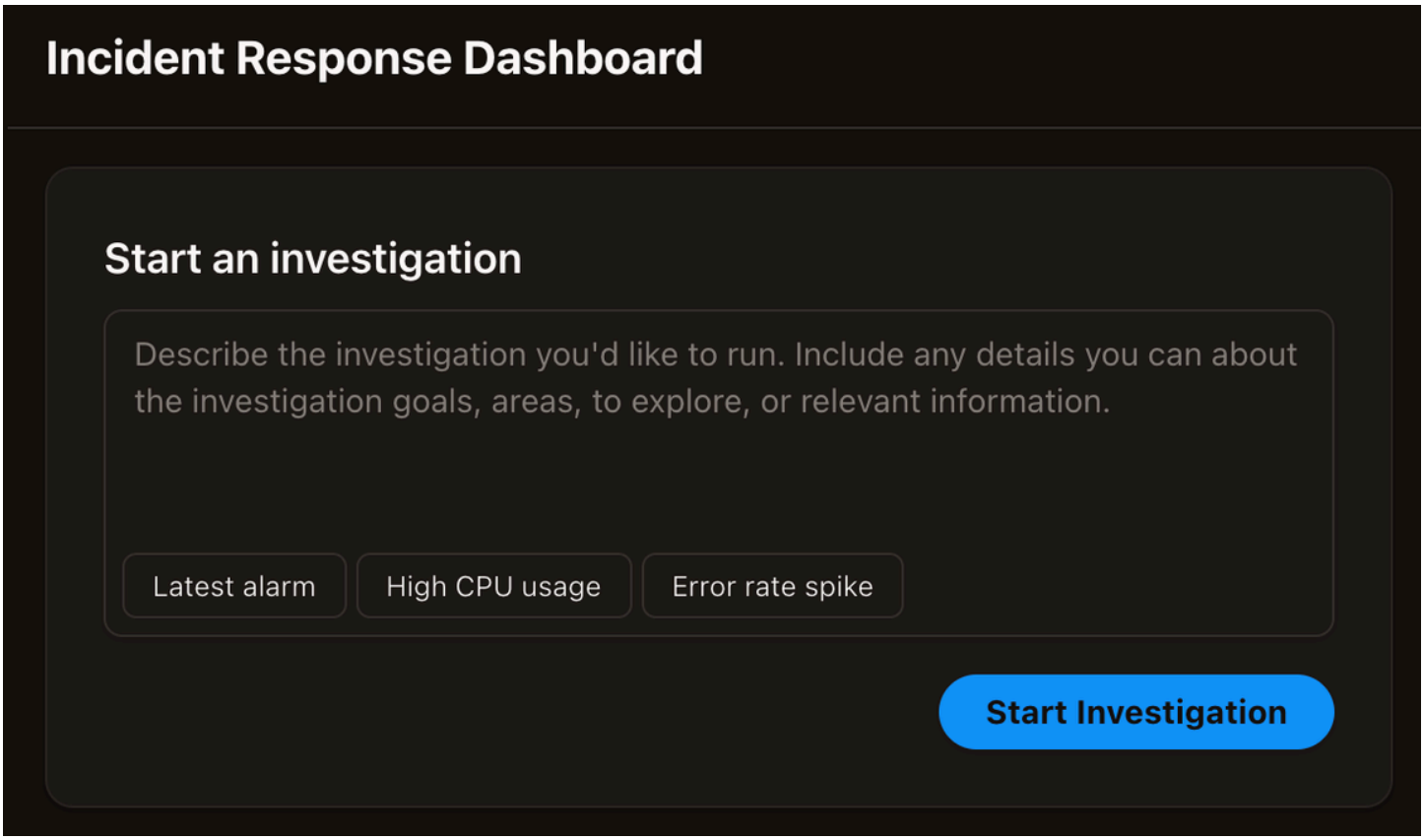
解决事件后， DevOps Agent 会分析您的调查历史记录中的模式，以生成建议，防止将来发生事件并缩短平均检测时间。建议涵盖四个方面：可观察性态势、测试差距、代码变更和基础架构架构。代理每周进行一次评估，并在发生新事件时更新建议。您可以接受、拒绝或跟踪推荐，客服人员会从您的反馈中学习，以完善未来的建议。有关更多信息，请参阅 the section called “主动预防事故”。

自主事件响应

开始调查

事件响应调查可以通过以下三种方式之一启动。

- 内置集成 ——你可以像 ServiceNow 使用内置集成一样将 A DevOps gent Space 连接到票务系统。连接后，将从支持票证中自动触发 DevOps 代理事件响应调查，您的 DevOps 代理将向原始工单提供其主要发现、根本原因分析和缓解计划的更新。
- Webhooks-你可以使用 webhook 向代理发送事件。 AWS DevOps 例如，您可以使用 webhook 通过工 PagerDuty 单或 Grafana 警报触发事件响应调查。
- 手动-您可以从任何 A DevOps gent Space 网络应用程序的 “事件响应” 选项卡手动启动事件响应调查。您可以输入描述您希望 DevOps 代理调查的事件的自由格式文本，它将制定调查计划，收集调查结果，确定根本原因，并主动提出制定缓解计划。您还可以从几个预先配置的起点中进行选择以快速开始调查：最新警报用于调查您最近触发的警报并分析底层指标和日志以确定根本原因；CPU 使用率高，用于调查计算资源中的高 CPU 利用率指标并确定哪些进程或服务消耗了过多的资源；或者错误率峰值通过分析指标、应用程序日志和确定故障来源来调查应用程序错误率最近增加的情况。



单击 “开始调查” 后，系统会要求您提供一些其他详细信息，以帮助代理集中精力开展工作。调查对话框包括以下字段：

- 调查详情-预先填写您的描述。您可以对其进行编辑以缩小调查范围。
- 调查起点-（可选）描述代理的特定警报、指标、日志片段或其他起点。

- 事件发生日期和时间 — 以 UTC 格式自动填充当前时间。如果事件发生得更早，请进行调整。
- 命名您的调查-自动生成，带有时间戳。您可以对其进行自定义（最多 400 个字符）。
- 优先级-从下拉列表中选择调查优先级（默认为 “中”）。

根据需要查看并调整这些字段，然后单击 “开始调查...” 开始。然后，您将被带到调查详细信息页面，在那里您可以看到您的 DevOps 特工在行动！

事件分类

分类阶段是 AWS DevOps Agent 事件响应系统的第一阶段。当外部事件触发时，例如来自 Datadog 的警报、来自 Dynatrace 的事件单或 Dynatrace 的问题， AWS DevOps Agent 会在几秒钟内自动处理该事件，以确定是应独立调查还是将其与现有调查相关联。 ServiceNow

分诊阶段的主要功能是事件关联——识别相关事件并将其整合到单一调查中，以避免重复工作和资源浪费。当新事件到来时， AWS DevOps Agent 会在回顾窗口（通常为 20 分钟）内对事件进行分析，同时进行主动调查。它使用人工智能驱动的分析，检查组件相似性、地理区域和时间模式等因素，以确定事件之间的关系。

AWS DevOps 代理做出以下两个决定之一：

- 关联 — 将事件与现有调查关联起来，并向该调查发送指导信息，其中包含有关新事件的背景信息。
- 继续 — 安排对事件进行新的独立调查。

查看分诊决定

当事件关联时，主要调查会收到一条指导信息，其中包含关联事件的详细信息和关联推理。在你的 A AWS DevOps gent Space Web 应用程序上，你会看到 “已关联” 的状态以及解释事件关联原因的关联推理。主要调查显示所有关联事件的列表，使您可以一起查看正在调查的相关问题的全部范围。您的外部票务系统（ServiceNow PagerDuty、等）和沟通渠道（Slack）将收到一条通知，告知该事件与关联推理有关。

取消关联事件和自定义关联规则

如果 AWS DevOps 代理错误地关联了事件，则可以通过 Agen AWS DevOps t Space Web 应用程序手动取消与事件的关联。这将把这起无关的事件重新安排为独立调查。您还可以通过创建包含您的关联逻辑的 AWS DevOps AWS DevOps 代理技能并将其与会审阶段关联来提供自定义关联规则来指导代理。

寻求人类支持

AWS DevOps 客服人员可以直接与 Su AWS pport 联系，以简化您的事件响应流程。当您需要 AWS 支持部门的其他帮助时，您可以通过 A DevOps gent Space 网络应用程序创建支持案例，自动与 AWS 支持工程师共享调查背景，从而缩短解释问题所需的时间。

工作原理

在调查事件时， AWS DevOps 代理会生成其分析的全面日志，包括：

- 根本原因调查结果
- 分析的指标、日志和跟踪
- 审查了代码更改和部署历史记录
- 建议的补救措施
- 事件和系统行为的时间表

您可以直接通过 A AWS DevOps gent Space 网络应用程序将 AWS 调查上报给 Support。当您这样做时， AWS DevOps Agent 会自动将其调查日志传递给 Support，从而为支持工程师提供有关您的调查的完整背景信息，而无需您手动收集和解释细节。 AWS

与 Su AWS pport 聊天

创建支持案例后，您可以在 A AWS DevOps gent Space 网络应用程序的单独聊天窗口中与 AWS 支持部门沟通。从而让您能够实现以下目的：

- 与 Su AWS pport 工程师讨论您的问题以及您的 AWS DevOps 客服人员的调查时间表
- 在同 AWS DevOps 一个界面中查看客 AWS 服的自动分析和支持部门的专家指导
- 根据需要无缝共享其他信息或澄清

聊天体验可让您随时访问 AWS DevOps 代理调查和 S AWS upport 对话，从而加快协作和解决问题的速度。

Support 计划要求

您能否通过 AWS DevOps 代理创建支持案例并与之交互取决于您的 AWS 支持计划。请参阅 Supp ort Plans 用户指南，详细了解您的权利。

注意 Basic Support 客户无法创建技术支持案例，因此无法将 AWS DevOps 代理调查上报给 AWS 支持开发者支持客户可以通过 AWS DevOps 代理创建案例，但必须访问AWS 支持中心与支持工程师通信，因为开发者支持不包括基于聊天的支持。所有其他计划都可以使用代理中的集成聊天体验。 AWS DevOps 有关支持计划权利的完整详细信息，包括响应时间和可用案例严重程度，请参阅 Support Pl AWS ans 用户指南。

与 Support 共享了 AWS 哪些信息

当您通过 A AWS DevOps gent Space 网络应用程序创建支持案例时，系统会自动与 AWS 支持部门共享以下信息：

- 调查时间表：按时间顺序排列的特 AWS DevOps 工分析记录
- 资源信息：受影响的 AWS 资源
- 可观察性数据：来自集成监控工具的相关指标、日志和跟踪
- 最新更改：代码部署、基础架构更改和配置更新
- 修复尝试：建议使用操作 AWS DevOps 代理
- 影响评估：事件的范围和严重程度

与 Su AWS pport 共享的所有数据都遵循您现有 AWS 的数据驻留和安全配置。 AWS DevOps Agent 仅共享与您的具体调查相关的信息，并尊重贵组织的数据治理政策。

开始使用

要使用 AWS DevOps 代理的 Su AWS pport 集成，请执行以下操作：

1. 确保您有一个有效的Su AWS pport计划。
2. 验证您的 AWS DevOps 代理的 IAM 权限包括创建支持案例（支持：CreateCase，支持：DescribeCases）。
3. 当 AWS DevOps 代理正在调查问题并且您需要 AWS 支持帮助时，请从您的 A DevOps gent Space 网络应用程序中选择 “寻求人工支持”。
4. 查看将与 Support 共享的 AWS 调查摘要。
5. 根据您的支持计划权利选择适当的案例严重性。
6. 提交案例- AWS DevOps 代理会自动包含您的调查日志。

聊天窗口会自动打开，允许您立即开始与 Supp AWS ort 合作。

主动预防事故

AWS DevOps 代理分析您的事件调查模式，以提供有针对性的建议，从而持续改善您的运营状况并防止将来发生事件。通过 Operator Web 应用程序中的 Ops Backlog 页面访问主动事件预防。

主动式事件预防的工作原理

AWS DevOps 代理评估最近的事件调查，以确定持久的改进措施，以防止将来发生事故并缩短平均检测时间 (MTTD)。该代理分析多起事件，以确定可以防止将来发生整类事件的建议，重点是最有影响力的建议，以确保这些建议具有可操作性。

默认情况下，代理每周自动运行一次评估。如果您希望仅按需进行评估，则可以暂停日程安排。手动评估随时可用，当最近的调查要求对建议的改进进行快速周转时，这很有用。

代理发现了四个类别的改进，显示在 Ops Backlog 页面的建议分类图表中：

- 可观察性-增强监控、警报、日志记录和系统可见性的建议，以更快、更准确地检测问题。
- 基础架构 — 优化资源配置、容量调整和架构弹性的建议。
- 治理 — 关于加强部署流程、管道改进、测试实践和操作控制的建议。
- 代码优化-改善应用程序代码质量、错误处理和代码弹性的建议。

这种分类可帮助您了解最需要改进运营的地方，并允许您根据团队的重点领域确定建议的优先顺序。

优势

- 防止事件反复出现 — 系统地解决根本原因，而不是反复应对相同类型的问题
- 减少运营疲劳 — 让您的团队摆脱重复的消防工作，专注于创新和战略改进
- 提高系统弹性 — 根据真实事件数据加强您的基础架构、可观察性和部署流程
- 从历史模式中学习 — 利用过去事件的见解，进行有针对性的改进，从而产生最大的影响

代理摘要

Web App 的 Ops Backlog 页面中的 “代理摘要” 描述了最近一次事件评估的结果。摘要说明了分析的事件调查数量，哪些事件与过去的事件相似，以及哪些建议是根据新信息创建或更新的。

该摘要可帮助您快速了解代理在最近的评估中发现了什么，并重点介绍了可能对您的运营状况产生最大影响的最值得注意的建议。

控制评估

您可以控制 AWS DevOps 代理何时评估事件并生成建议：

- 手动运行评估-单击 Ops Backlog（操作待办事项）页面中的立即运行按钮可立即开始评估。当最近的调查要求对建议的改进进行快速周转时，这很有用。
- 停止正在进行的评估-单击 Ops Bac k log（操作待办事项）页面中的 “停止评估” 按钮可暂停当前正在进行的评估。

管理推荐

AWS DevOps 代理在 Ops Backlog 页面中提供建议，您可以在其中查看和管理这些建议：

- 查看建议详情-单击建议可打开建议详细信息页面，您可以在其中查看有关建议改进的更多信息，包括为该建议提供依据的事件、预期影响和后续步骤。有关代码更改的建议，您还可以查看代理就绪规范，该规范可以交给编码代理实施。
- 保留 — 单击 “保留” 可在待办事项列表中保留建议以供跟踪。这使您可以监控计划实施哪些改进并跟踪其进度。
- 丢弃 — 单击 “放弃” 可从待办事项列表中删除推荐。当你放弃推荐时，你可以用自然语言解释为什么它不能满足你的需求。工程师从这些反馈中吸取教训，并利用这些反馈为未来的建议提供信息，确保这些建议随着时间的推移与您的运营优先事项和要求更加一致。
- 已实施-单击 “已实施” 将建议标记为已完成。这可以帮助您跟踪应用了哪些改进，并允许代理衡量其建议在一段时间内的有效性。
- 自动删除-未标记为 “保留” 或 “已实施” 的建议可以在大约 6 周后删除，前提是实施该建议无法防止出现新的事件。这样可以确保 Ops Backlog 页面专注于最相关的改进，以应对您的运营挑战。
- 建议更新 — 当发现建议本来可以防止的新事件时，会更新现有建议。更新可能会更改建议的优先级或根据新的见解完善建议。

代理就绪规格

对于涉及代码或配置更改的建议， AWS DevOps 代理可以生成代理就绪规范。该规范提供了一个结构化文档，可以直接交给编码代理进行实施。

该规格包括：

- 问题陈述-问题及其根本原因的摘要

- 解决方案摘要-对推荐方法的高级描述
- 目标存储库-需要进行更改的特定存储库
- 代码更改 — 详细描述需要更改的内容和原因，以及特定的文件路径和实现注意事项
- 测试要求-需要测试哪些场景
- 实施计划 — 实施变更的分阶段方法

Agent 就绪规范通过为编码代理提供进行生产就绪更改所需的上下文，而无需与工程师进行大量合作，从而加快实施速度。 back-and-forth

实施建议

为了最大限度地发挥主动事件预防建议的价值，请考虑采取以下措施来执行这些建议：

- 使用代理就绪规范 — 有关代码变更的建议，请使用生成的规范将其交给编码代理或将其用作手动实施的详细指南，从而加快实施。
- 向工单待办事项中添加建议 — 将建议复制到团队的工单系统或项目管理工具，以确保这些建议与其他工程工作一起被优先考虑。
- 根据影响对建议进行优先排序 — 首先关注针对最常见或最严重的事件类型或影响关键系统的建议。
- 跟踪实施进度 — 通过观察类似事件是否随着时间的推移而减少，监控哪些建议已得到实施，并衡量其有效性。
- 与开发团队协调-与拥有受影响系统的相应团队共享建议，确保他们拥有实施改进所需的背景和资源。

按需 DevOps 任务

AWS DevOps Agent On Demand Tasks 是一款生成式人工智能 (AI) 驱动的对话助手，使运营团队能够使用自然语言查询其应用程序架构、分析系统运行状况并访问调查见解。您可以询问有关您的 AWS 资源、系统指标、警报状态、部署历史和事件模式的问题。聊天可根据您的实际基础设施和运营数据提供即时答案，无需在多个 AWS 控制台或监控工具之间切换。

聊天集成在 A DevOps gent Space Web 应用程序中，可根据您正在查看的页面提供上下文感知响应。该界面保留对话历史记录，使您能够继续之前的讨论，并在之前的查询基础上再接再厉。

任务能力

AWS DevOps Agent On Demand Tasks 提供全面的功能来帮助您管理和了解您的基础架构：

资源查询 — 询问代理空间中的 AWS 资源，包括 Lambda 函数、DynamoDB 表、EKS 部署、证书和基础设施配置。聊天可以根据运行时版本、容量设置或部署状态等属性筛选和分析资源。例如，问 “有多少 Lambda 在使用 Python 3.8？” 或 “我有即将过期的证书吗？”

系统运行状况分析-查询当前和历史系统运行状况指标，包括警报状态、错误率、CPU 利用率和服务可用性。聊天可以生成涵盖特定时间段的健康摘要，并识别系统行为的趋势。问诸如 “过去 24 小时内触发了哪些警报？” 之类的问题 或 “过去一小时内有 5xx 错误吗？”

调查见解-访问已完成和正在进行的调查的信息，包括根本原因分析、探讨的假设、审查的日志和解决方案模式。聊天可以识别常见的事件原因，并根据历史数据提供建议。查询 “上个月最常见的事件原因是什么？” 或 “完成调查的平均解决时间是多少？”

调查指导-在查看调查详情页面时，通过指示代理关注特定日志、探索特定的假设或更新根本原因分析来指导调查。提供指导输入，例如 “关注支付服务的日志并更新您的 RCA” 或 “探索 DynamoDB 限制导致问题的假设”。

聊天对象-生成结构化报告和文档，例如运行状况摘要、错误报告和事件分析。构件显示在专用面板中，并支持在对话中进行版本化编辑。

建议筛选-使用特定标准（例如与特定服务或运营问题相关的建议）查询事件预防建议。Chat 解释了每项建议的影响和实施注意事项。例如，“向我展示可以防止涉及 DynamoDB 的事件的建议” 或 “哪些建议可以帮助我更快地检测请求延迟问题？”

访问聊天

聊天作为永久面板位于 A DevOps gent Space 网络应用程序的左侧。左侧边栏包括一个 + New 聊天按钮、一个用于导航到 “事件”、“操作待办事项列表” 和 “拓扑” 的 “页面” 部分，以及一个显示你最近对话的 “聊天” 部分。选择 “查看全部” 以查看您的完整对话历史记录。

Chat 会根据您的访问位置提供情境感知响应：

拓扑 — 询问有关您的 Agent Space 资源、架构和运行状况的一般问题。聊天功能可以全面了解所有关联的账户和服务。在此上下文中，您可以查询资源配置、部署历史记录、拓扑信息和可观测性工具集成。

事件响应-查看事件响应页面时，询问有关代理空间中的调查趋势、解决时间和事件模式的问题。Chat 可以分析历史调查数据，找出常见原因和改进机会。

调查详情-在查看特定调查时，Chat 会提供有关该调查的上下文感知回复。询问已审查的日志、探讨的假设、根本原因结论和缓解计划。您还可以提供指导性输入来指导调查重点。

预防-在预防页面上，使用过滤器查询建议，了解提出建议的原因，并探索实施方法。聊天可帮助您确定事件预防建议的优先顺序并了解其影响。

当您在页面之间切换时，聊天界面仍然可用，但是上下文会发生变化，以便为当前视图提供相关信息。当你开始新的对话时，它是在没有先前背景的情况下开始的。当您继续进行现有对话时，Chat 会保留完整的对话历史记录，供后续问题使用。

情境感知响应

Chat 会根据你在 A DevOps gent Space 网络应用程序中查看的页面来调整其响应。这种情境感知可确保您无需指定要询问的调查或资源范围即可获得相关信息。

查看调查详情页面时，Chat 会自动了解到您询问的是该特定调查。诸如 “你看了什么日志？” 之类的问题 或者 “你探索了哪些假设？” 请参阅当前显示的调查。当您提供指导输入时，Chat 会将其应用于正在进行的调查，并在适当的情况下创建新的根本原因版本。

在 “预防” 页面上，Chat 了解到您对事件预防建议感兴趣。查询会自动筛选和分析您的 Agent Space 上下文中的推荐。系统会识别您询问的是一般性建议还是具体的建议细节。

从 “拓扑” 页面访问 “聊天” 时，“聊天” 可让您广泛了解座席空间中的所有资源、指标和历史数据。您可以询问任何资源、服务或运营问题，而无需具体说明调查或建议背景。

这种情境感知使您无需反复指定您所引用的调查、建议或资源范围，从而创建更自然的对话流程。

管理对话

聊天保留对话历史记录，使您可以继续之前的讨论并参考之前的查询。

创建新对话-单击聊天面板中的 “新会话” 按钮，无需事先提供上下文即可开始新的对话。新的对话不会延续以前聊天中的信息，因此您可以毫不费力地提出不相关的问题。

访问对话历史记录-单击 “历史记录” 可查看代理空间中之前的所有对话。对话按时间顺序排列，并带有时间戳和预览文本。对话历史记录会保留 90 天，并且对您在代理空间中的用户帐户不公开。

继续对话-从历史记录中选择任何对话以从上次中断的地方继续。聊天保留了之前消息的完整上下文，使您能够提出引用对话早期部分的后续问题。当您在查看对话时切换页面时，对话上下文会保留，但特定页面的上下文会根据您的当前位置进行更新。

请注意，对话历史记录在每个代理空间中是隔离的。一个座席空间中的对话不可见，也无法从其他座席空间访问。这种隔离可确保根据您的组织界限对敏感信息进行隔离。

生成工件

AWS DevOps 代理支持聊天构件，即代理在对话期间生成的结构化版本化文档。Artifacts 在聊天界面中提供了一个专用的交互式面板，用于查看和编辑 AI 生成的内容，例如运营报告、错误摘要和运行状况评估。

您可以从 A DevOps gent Space Web 应用程序中的任何页面请求构件。Chat 使用当前页面上下文来确定构件内容的范围。

工件的工作原理

当你让 Chat 创建或更新内容时，Chat 会生成一个构件（通常是格式化的文档），并将其显示在对话旁边的构件面板中。

生成-发送自然语言请求以创建报告或文档。例如，询问 “为我的 Agent Space 生成每周运行状况报告” 或 “向我出示上周 4xx 错误的报告”。

查看 — 文物显示在对话旁边的专用面板中。您可以在继续与 Chat 互动的同时查看完整内容。

编辑-通过 “聊天” 请求对工件进行更改。例如，询问 “添加有关 Lambda 冷启动的部分” 或 “更新报告以包含上个月的数据”。Chat 会根据您请求的更改创建工件的新版本。

查询示例

以下示例演示了您可以向 Chat 提问的问题类型。这些示例按用例和上下文进行组织。

Artifact 生成查询

在 DevOps Agent Space 网络应用程序的任何页面上：

- 为我的 Agent Space 生成每周运行状况摘要
- 创建上周所有 4xx 错误的报告
- 生成过去 30 天的事件摘要报告
- 创建本周支付服务的警报活动摘要
- 生成最近 7 天的部署历史报告
- 将所有未解决的建议汇总到一份报告中

资源信息查询

在 DevOps Agent Space 网络应用程序的任何页面上：

- 有多少 Lambda 函数在使用 Python 3.8？
- 我有即将过期的证书吗？
- 列出所有包含按需计费的 DynamoDB 表
- 向我展示生产中的 EKS 集群
- 在过去 90 天内未部署哪些 Lambda 函数？
- 列出未启用版本控制的 S3 存储桶
- 哪些 RDS 实例正在运行数据库版本 X？

系统运行状况查询

从 “拓扑” 或 “事件响应” 页面：

- 过去 24 小时内触发了哪些警报？
- 过去一小时内有 5xx 错误吗？
- 向我展示支付服务的 Lambda 错误趋势
- 我的 ECS 集群的 CPU 使用率是多少？
- 我的负载均衡器中是否有不健康的目标？
- 给我看昨天的 API Gateway 限制事件
- 上周哪些服务的错误率最高？
- 给我一份涵盖过去 24 小时的总体健康报告

可观测性工具查询

来自拓扑：

- 列出 Splunk 日志组
- 向我展示 Prometheus 指标及其警报阈值
- 为该服务配置了哪些 Datadog 监视器？
- 列出新的 Relic 警报策略
- 给我看 Dynatrace 仪表板配置

调查见解查询

从 “事件响应” 页面：

- 上个月最常见的事件原因是什么？
- 完成调查的平均解决时间是多少？
- 总结上周的调查及其风险评估
- 有多少事件是由于 DynamoDB 限制引起的？
- 向我展示过去一个季度的调查趋势
- 哪些服务发生的事件最频繁？

调查详情查询

从调查详情页面：

- 你看了什么日志？
- 你探讨了哪些假设？
- 你提议的缓解措施有多危险？
- 这起事件中发生的事件的时间表是什么？
- 你为什么得出结论，这是根本原因？
- 哪些证据支持您的根本原因分析？
- 在你的调查期间，谁提供了指导？
- 给我一份事件调查的摘要

调查指导查询

从调查详情页面：

- 关注世界标准时间 14:00-15:00 之间的支付服务日志，并更新您的 RCA
- 探索 DynamoDB 限流导致问题的假设
- 检查 ECS 集群配置以查看警报是否由此引起
- 只查看最近 2 小时的日志，而不是整天的日志
- 调查下午 3 点的错误激增情况
- 查看 API Gateway 日志而不是 Lambda 日志

预防建议查询

来自 “预防” 页面：

- 我的三大事件预防建议是什么？
- 向我展示可以防止涉及 DynamoDB 的事件的建议
- 哪些建议可以帮助我更快地检测请求延迟问题？
- 列出可以防止类似事件的可观测性改进
- 向我展示支付服务的基础设施建议
- 哪些建议对系统弹性的影响最大？

在特工空间中启用 “聊天”

聊天功能可在所有 DevOps Agent Space 网络应用程序中使用。设置过程取决于您的座席空间是新的还是现有的座席空间。

新特工空间

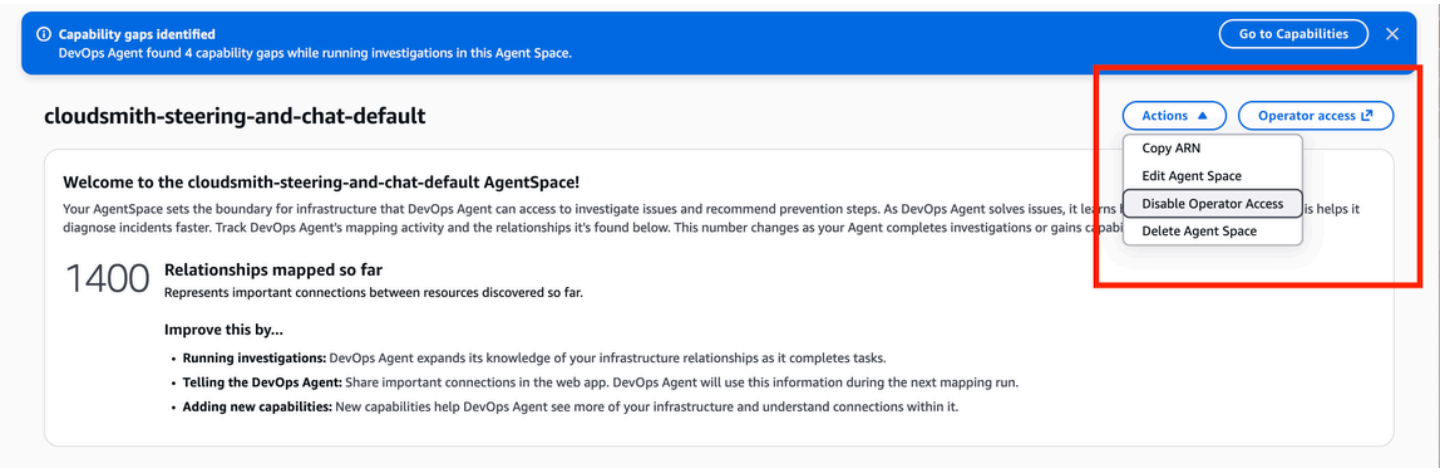
当您创建新的代理空间时，聊天功能会自动启用。无需进行其他配置或 IAM 权限设置。配置 A DevOps gent Space Web 应用程序后，聊天将立即作为永久面板显示在任何页面的左侧。

现有代理空间

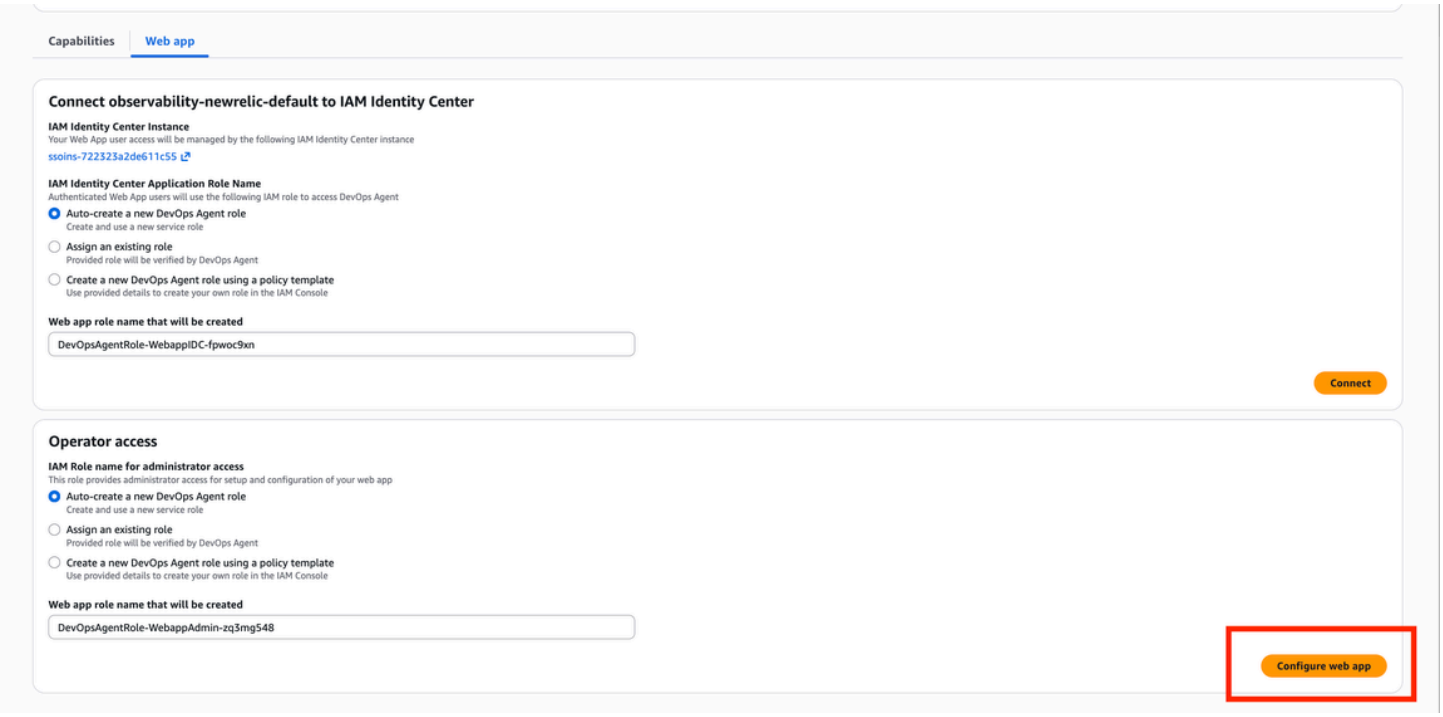
如果您在 Chat 发布之前创建了代理空间，则必须启用所需的 IAM 权限。你有两个选择：

选项 1：撤销并重新启用操作员应用程序访问权限

导航到 AWS DevOps 代理管理员控制台，找到右上角的操作下拉列表，然后禁用当前的操作员访问配置。



然后启用自动创建选项以供操作员访问。



这会自动应用聊天所需的 IAM 权限以及所有其他当前操作员权限。

选项 2：手动添加 IAM 权限

将以下 IAM 权限添加到您现有的操作员访问角色中：

- `aidevops:ListChats`— 查看聊天对话记录
- `aidevops:CreateChat`— 创建新的聊天对话
- `aidevops:SendMessage`— 发送消息并接收回复

导航到 AWS IAM 控制台，找到您的 DevOps 代理操作员角色，然后将这些权限添加到角色策略中。添加权限后，聊天将立即可用。

完成任一选项后，刷新您的 A DevOps gent Space Web 应用程序，聊天面板将出现在任何页面的左侧。

为 AWS DevOps 代理配置功能

AWS DevOps 代理功能通过将代理连接到您现有的工具和基础架构来扩展代理的功能。配置这些功能以实现全面的事件调查、自动响应工作流程以及与您的 DevOps 生态系统的无缝集成。

以下功能可帮助您最大限度地提高 DevOps 代理的效率：

- AWS EKS 访问设置-为公共和私有 EKS 环境启用对 Kubernetes 集群、容器日志和集群事件的自省
- Azure 集成 ——连接 Azure 订阅和 Azure DevOps 组织，调查 Azure 资源并将 Azure DevOps 部署与事件关联起来
- CI/CD Pipeline In tegration-Connect GitHub 和 GitLab 管道可将部署与事件关联起来，并在调查期间跟踪代码更改
- MCP 服务器连接-通过模型上下文协议连接外部可观测性工具和自定义监控系统，从而扩展调查功能
- 多账户 AWS 访问权限-配置辅助 AWS 账户，以便在事件响应期间调查整个组织的资源
- 遥测源集成 ——连接 Datadog、Dynatrace、Grafana、New Relic 和 Splunk 等监控平台，实现全面的可观察性数据访问
- 票务和聊天集成 ——Connect ServiceNow PagerDuty、和 Slack 可自动执行事件响应工作流程并实现团队协作
- Webhook 配置-允许外部系统通过 HTTP 请求自动触发 DevOps 代理调查
- Amazon EventBridge 集成-通过将调查和缓解生命周期事件路由到亚马逊目标，将 AWS DevOps 代理整合到事件驱动的应用程序中 EventBridge

您可以根据团队的特定需求和现有工具堆栈独立配置每项功能。从对您的事件响应工作流程最重要的集成开始，然后根据需要扩展到其他功能。

从公开预览版迁移到正式发布

如果您在公开预览版期间使用了 AWS DevOps 代理，则必须在 GA 发布之前更新您的 IAM 角色。本指南介绍如何更新账户中的监控角色和操作员角色。

发生了什么变化

1. 无法再访问预览期间的按需聊天记录
2. 新的托管策略取代了预览期间可用的策略

3. 代理空间的 IAM Identity Center 应用程序访问范围可能已过时

公共预览版中的按需聊天记录

GA 版本引入了额外的安全措施，以加强对聊天记录的访问控制。由于这些变化，无法再访问公开预览期（2026年3月30日之前）的按需聊天记录。在公开预览期间创建的调查期刊和调查结果不受影响。此更改仅适用于按需聊天对话。

新的托管策略

对于 GA， AWS 提供了新的托管策略来取代预览时代的策略：

角色类型	删除	添加
监控	`AIOpsAssistantPolicy` 托管策略	`AIDevOpsAgentAccessPolicy` 托管策略
运营商（IAM 和 IDC）	内联策略	`AIDevOpsOperatorAppAccessPolicy` 托管策略

此外，运营商角色需要更新的信任策略，IDC 运营商角色需要新的内联策略。

先决条件

- 访问配置了您的 DevOps 代理角色的 AWS 账户（主账户和所有次要账户）
- 修改角色、策略和信任关系的 IAM 权限
- 您的座席空间 ID、 AWS 账户 ID 和区域（在 DevOps 代理控制台中可见）

步骤 1：更新监控角色

更新您的主账户和每个辅助账户中的监控角色。这些是在您的代理空间的 “权能” 选项卡下配置的 Primary/Secondary 源角色（示例 primary/secondary 角色:`DevOpsAgentRole-AgentSpace-3xj2396z`）。

1. 在 DevOps 代理控制台中，前往您的代理空间，然后选择功能选项卡。
2. 找到 Primary/Secondary 源代码的监视角色（例如`DevOpsAgentRole-AgentSpace-3xj2396z`），然后选择编辑。

3. 在 “权限策略” 下，移除AIOpsAssistantPolicy AWS 托管策略。

4. 选择添加权限、附加策略并附加AIDevOpsAgentAccessPolicy托管策略。

5. 编辑内联政策并将其内容替换为以下内容，替换您的账户 ID：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateServiceLinkedRoles",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": [
                "arn:aws:iam::<account-id>:role/aws-service-role/resource-
explorer-2.amazonaws.com/AWSServiceRoleForResourceExplorer"
            ]
        }
    ]
}
```

1. 无需更改监控角色的信任策略。验证它是否与以下内容匹配：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/*"
                }
```

```
            }
        }
    ]
}
```

- 对每个辅助账户中的监控角色重复步骤 2—6。

步骤 2：更新操作员角色 (IAM)

1. 在 DevOps 代理控制台中，选择访问选项卡并找到操作员角色。
2. 在 IAM 控制台中，从操作员角色中移除现有的内联策略。
3. 选择添加权限、附加策略并附加AIDevOpsOperatorAppAccessPolicy托管策略。
4. 选择 “信任关系” 选项卡，然后选择 “编辑信任策略”。将信任策略替换为以下内容，替换您的账户 ID、地区和座席空间 ID：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": ["sts:AssumeRole", "sts:TagSession"],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                }
            }
        }
    ]
}
```

步骤 3：更新操作员角色 (IDC)

如果您将 IAM 身份中心与 DevOps 代理一起使用，请更新每个 IDC 操作员角色。

1. 在 IAM 控制台中，转至角色并搜索`WebappIDC`以查找您的 DevOps 代理 IDC 角色（例如`DevOpsAgentRole-WebappIDC-<id>`）。
2. 对于每个 IDC 角色：

a. 移除现有的内联策略。

b. 选择添加权限、附加策略并附加`AIDevOpsOperatorAppAccessPolicy`托管策略。

c. 选择 “信任关系” 选项卡，然后选择 “编辑信任策略”。将信任策略替换为以下内容，替换您的账户 ID、地区和座席空间 ID：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": ["sts:AssumeRole", "sts:TagSession"],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                }
            }
        },
        {
            "Sid": "TrustedIdentityPropagation",
            "Effect": "Allow",
            "Principal": {
                "Service": "aidevops.amazonaws.com"
            },
            "Action": "sts:SetContext",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:SourceAccount": "<account-id>"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:aidevops:<region>:<account-
id>:agentspace/<agentspace-id>"
                },
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "Null": {
                    "sts:RequestContextProviders": "false"
                }
            }
        }
    ]
}
```

d. 使用以下权限创建新的内联策略，替换您的账户 ID：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDevOpsAgentSSOAccess",
            "Effect": "Allow",
            "Action": [
                "sso:ListInstances",
                "sso:DescribeInstance"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowDevOpsAgentIDCUserAccess",
            "Effect": "Allow",
            "Action": "identitystore:DescribeUser",
            "Resource": [
                "arn:aws:identitystore::<account-id>:identitystore/*",
                "arn:aws:identitystore:::user/*"
            ]
        }
```

```
    ]
}
```

重新连接 IAM 身份中心（如果适用）

在公开预览版期间创建的代理空间可能将 IAM Identity Center 应用程序配置为过时的访问范围。对于 GA，正确的范围是**aidevops:read_write**。如果您的 IAM Identity Center 应用程序具有之前的作用域 (**awsaidevops:read_write**)，则必须断开连接并重新连接 IAM 身份中心。

如何检查您的 IAM 身份中心应用程序范围

运行以下 AWS CLI 命令以检查您的 IAM 身份中心应用程序的范围。您可以在 IAM 身份中心控制台的“应用程序” 下找到应用程序 ARN。

```
aws sso-admin list-application-access-scopes \
  --application-arn arn:aws:sso::<account-id>:application/<instance-id>/<application-
id>
```

输出应显示正确的范围 **aidevops:read_write**：

```
{
    "Scopes": [
        {
            "Scope": "aidevops:read_write"
        }
    ]
}
```

如果示波器显示 **awsaidevops:read_write**，则表示它已过时。请按照以下步骤进行更新。

如何重新连接 IAM 身份中心

AWS 托管 IAM 身份中心应用程序的访问范围无法直接更新。您必须断开连接并重新连接：

1. 在 AWS DevOps 代理控制台中，前往您的代理空间并选择访问选项卡。
2. 选择 IAM 身份中心配置旁边的断开连接。
3. 确认断开连接。
4. 选择 Connect 再次设置 IAM 身份中心。该服务创建了一个具有正确范围的新 IAM 身份中心应用程序。

5. 在 IAM Identity Center 控制台中将用户和群组重新分配给新应用程序。

> **Important**
>
> 断开连接会删除与 IAM Identity Center 用户账户关联的个人用户聊天和项目历史记录。用户在重新连接后需要重新登录。

验证

完成所有步骤后：

1. 返回 DevOps 代理控制台，确认代理空间访问选项卡上没有出现任何权限错误。
2. 测试操作员 Web 应用程序，确认其加载和运行正常。
3. 如果您使用 IDC，请验证用户是否可以进行身份验证并访问操作员体验。

问题排查

迁移后出现权限被拒绝错误

- 确认它已`AIOpsAssistantPolicy`被移除并`AIDevOpsAgentAccessPolicy`已附加到监视角色。
- 确认旧的内联策略已被移除并`AIDevOpsOperatorAppAccessPolicy`已附加到操作员角色。
- 检查运营商信任策略是否包括`sts:TagSession`。
- 确认已将所有占位符值（`<account-id>`、`<region>`、`<agentspace-id>`）替换为实际值。

二级账户不起作用

- 每个辅助账户的监控角色必须独立更新。登录每个账户并重复步骤 1。

IDC 身份验证失败

- 验证 IDC 信任策略同时包含`sts:AssumeRole`/`sts:TagSession`语句和语`TrustedIdentityPropagation`句。
- 使用、确认内联策略 `sso:ListInstancessso:DescribeInstance`，并`identitystore:DescribeUser`已创建。

迁移后缺少按需聊天记录

- 在 GA 发布后，无法访问公开预览期的按需聊天记录。这是由于 GA 中引入的增强安全措施而出现的预期行为。调查期刊和公开预览版的结果不受影响。

AWS EKS 访问权限设置

您可以通过对公有集群和私有集群运行只读`kubectl`命令来让 AWS DevOps 代理调查您的 Amazon EKS 集群中的问题。您可以将任意数量的 EKS 集群连接到同一个代理空间。

连接后，代理可以帮助诊断集群中的操作问题——描述资源、检索 Pod 日志、检查集群事件、检查节点运行状况等。代理无法在您的集群中创建、修改或删除任何资源。

先决条件

在设置 EKS 访问权限之前，请确保您的 EKS 集群的身份验证模式包含 EKS API。您可以在 Amazon EKS 控制台的 “访问” 选项卡上进行检查。如果该模式不包含 EKS API，请在继续操作之前选择包含该模式的模式。

设置

对于要为其创建访问条目的每个集群，都需要从 Amazon EKS 控制台完成这些步骤。您可以在 “能力” > “云” > “主要来源” > “编辑” 下的 “代理空间”（参见the section called “创建代理空间”）中找到您的 IAM 角色 ARN。

1. 转到 “访问权限” 选项卡。如果身份验证模式已经显示 EKS API，则可以添加访问条目。否则，请选择包含 EKS API 的模式。
2. 在 “访问权限” 选项卡中，创建一个新的 IAM 访问权限条目。复制您的主云源 IAM 角色 ARN，并将其作为访问条目的 IAM 委托人输入。单击下一步。
3. 选择 Amazon AWS 托管AIOpsAssistantPolicy访问策略，然后选择集群作为访问范围。（或者，如果您希望代理仅访问某些命名空间，请选择所需的 Kuber netes 命名空间）。单击 “添加策略”，然后单击 “下一步”。
4. 查看更改并确认选择了正确的访问权限入口策略和 IAM 角色，然后单击 “创建” 创建您的访问条目。

要验证 EKS 访问权限配置是否正确，请导航到 Operator 应用程序并开始新的调查，向代理询问有关您的集群的问题，例如 “列出默认命名空间中的所有 pod” 或 “向我显示我的集群中的最近事件”。

问题排查

如果代理无法访问您的集群，请验证访问条目是否使用了设置对话框中显示的正确 IAM 角色 ARN 以及是否已附加 A mazon AIOps AssistantPolicy 访问策略。

连接 Azure

Azure 集成使 AWS DevOps 代理能够调查 Azure 环境中的资源，并将 Azure DevOps 管道部署与操作事件关联起来。通过连接 Azure，代理可以查看你的 Azure 基础架构，并可以对 Azure 资源 AWS 和 Azure 资源执行根本原因分析。

Azure 集成由两个独立的功能组成：

- Azure 资源 — 使代理能够发现和调查 Azure 云资源，例如虚拟机、Azure Kubernetes 服务 (AKS) 集群、数据库和网络组件。在事件调查期间，代理使用 Azure 资源图来查询你的资源。
- Azure DevOps-允许代理访问 Azure DevOps 存储库和管道执行历史记录。代理可以将代码更改和部署与事件关联起来，以帮助确定潜在的根本原因。

每项功能都是在 AWS 账户级别注册的，然后可以与各个代理空间相关联。

注册方法

AWS DevOps 代理支持两种连接到 Azure 的方法：

- 管理员同意 — 基于同意的简化流程，您可以在 Azure 租户中授权 A AWS DevOps gent Entra 应用程序。在控制台中，这显示为 “管理员同意” 选项。此方法需要使用有权在 Microsoft Entra ID 中执行管理员同意的帐户登录。
- 应用程序注册 — 一种自我管理的方法，您可以使用出站联合身份验证创建自己的带有联合身份凭证的 Entra 应用程序。在控制台中，这显示为 “应用程序注册” 选项。当您需要对应用程序配置进行更多控制或没有管理员同意权限时，此方法非常适合。

这两种方法都提供相同的功能。您可以在同一个 AWS 账户中使用一种或两种方法。

已知限制条件

- 管理员同意：每个 Azure 租户一个 AWS 帐户 — 每个 Azure 租户一次只能将其 A AWS DevOps gent Entra 应用程序与一个 AWS 帐户相关联。要将同一租户与其他 AWS 账户关联，必须先取消注册现有注册。

- 应用程序注册：每次注册的应用程序都是唯一的 — 每个应用程序注册必须使用不同的应用程序（客户端 ID）。不能使用相同的客户端 ID 注册多个配置。
- Azure DevOps：源代码访问权限 — 无论源代码托管在哪里，Azure DevOps 集成都提供对管道执行历史记录的访问权限。但是，要访问实际的源代码，必须通过支持的源提供程序（例如the section called “正在连接 GitHub”）单独连接存储库。托管在 Bitbucket 中的源代码无法通过 Azure DevOps 集成直接访问。

主题

连接 Azure 资源

Azure 资源集成使 AWS DevOps 代理能够在事件调查期间发现和调查 Azure 订阅中的资源。代理使用 Azure 资源图进行资源发现，并且可以访问 Azure 环境中的指标、日志和配置数据。

此集成遵循两个步骤的过程：在 AWS 帐户级别注册 Azure，然后将特定的 Azure 订阅与各个代理空间相关联。

先决条件

在连接 Azure 资源之前，请确保你有：

- 访问 AWS DevOps 代理控制台
- 有权访问目标订阅的 Azure 帐户
- 对于管理员同意方法：有权在 Microsoft Entra ID 中进行管理员同意的帐户
- 对于应用程序注册方法：具有配置联合身份凭证权限的 Entra 应用程序，并在您的 AWS 账户中启用了出站联合身份验证

> 注意：您也可以从代理空间内开始注册。导航到辅助源，单击 “添加”，然后选择 Azure。如果 Azure Cloud 尚未注册，则控制台会引导你先完成注册。

通过管理员同意注册 Azure 资源

管理员同意方法在 AWS DevOps 代理托管的应用程序中使用基于同意的流程。

第 1 步：开始注册

1. 登录 AWS 管理控制台并导航到 AWS DevOps 代理控制台
2. 转到能力提供者页面
3. 找到 “A zure 云” 部分，然后单击 “注册”
4. 选择管理员同意注册方法

第 2 步：完成管理员同意

1. 查看正在申请的权限
2. 点击继续 — 您将被重定向到 Microsoft Entra 管理员同意页面
3. 使用有权执行管理员同意的用户主账号登录
4. 审查 AWS DevOps 代理申请并授予同意

步骤 3：完成用户授权

1. 管理员同意后，系统会提示您进行用户授权，以验证您作为授权租户成员的身份
2. 使用属于同一 Azure 租户的帐户登录
3. 授权后，您将重定向回 AWS DevOps 代理控制台，状态为成功

步骤 4：分配角色

请参阅下面的分配 Azure 角色。选择成员时搜索AWS DevOps 代理。

通过应用程序注册注册 Azure 资源

应用程序注册方法使用您自己的 Entra 应用程序和联合身份凭证。

第 1 步：开始注册

1. 在 AWS DevOps 代理控制台中，转到功能提供者页面
2. 找到 “A zure 云” 部分，然后单击 “注册”
3. 选择应用程序注册方法

第 2 步：创建和配置您的 Entra 应用程序

按照控制台中显示的说明执行以下操作：

1. 在您的 AWS 账户中启用出站身份联合（在 IAM 控制台中，前往账户设置 → 出站联合身份验证）
2. 在你的 Microsoft Entra ID 中创建 Entra 应用程序，或者使用现有的 Entra 应用程序
3. 在应用程序上配置联合身份凭证

第 3 步：提供注册详情

在注册表中填写以下内容：

- 租户 ID — 你的 Azure 租户标识符
- 租户名称-租户的显示名称
- 客户端 ID — 您创建的 Entra 应用程序的应用程序（客户端）ID
- 受众-联邦凭证的受众标识符

步骤 4：创建 IAM 角色

当您通过控制台提交注册时，将自动创建 IAM 角色。它允许 AWS DevOps 代理使用凭据并调用`sts:GetWebIdentityToken`。

步骤 5：分配角色

请参阅下面的分配 Azure 角色。搜索您在选择成员时创建的 Entra 应用程序。

第 6 步：完成注册

1. 在 AWS DevOps 代理控制台中确认配置
2. 单击 “提交” 完成注册

分配 Azure 角色

注册后，授予应用程序对你的 Azure 订阅的读取权限。管理员同意和应用程序注册方法的此步骤相同。

1. 在 Azure 门户中，导航到你的目标订阅
2. 前往访问控制 (IAM)
3. 单击 “添加” > “添加角色分配”
4. 选择读者角色并单击 “下一步”

5. 单击 “选择成员”，搜索应用程序（要么是获得管理员同意的AWS DevOps 代理，要么是您自己的 Entra 应用程序进行应用程序注册）
6. 选择应用程序，然后单击 “查看 + 分配”
7. （可选）要使代理能够访问 Azure Kubernetes 服务 (AKS) 集群，请完成以下 AKS 访问权限设置。

> 安全要求：只能为服务主体分配读者角色（以及下面列出的 AKS 只读角色）。Reader 角色充当安全边界，将代理限制为只读操作，并限制间接提示注入攻击的影响。为角色分配写入或操作权限会大大增加提示注入的爆炸半径，并可能导致 Azure 资源受到损害。AWS DevOps 代理仅执行读取操作。代理不会修改、创建或删除 Azure 资源。

AKS 访问设置（可选）

步骤 1：Azure 资源管理器 (ARM) 级别访问权限

为应用程序分配 Azure Kubernetes 服务集群用户角色。

在 Azure 门户中，前往 “订阅” → “选择订阅” → “访问控制 (IAM)” → “添加角色分配” → 选择 Azure Kubernetes 服务集群用户角色 → 分配给应用程序（要么是获得管理员同意的AWS DevOps 代理，要么是你自己的 Entra 应用程序进行应用程序注册）。

这涵盖了订阅中的所有 AKS 集群。要将范围限定到特定群集，请改为在资源组或单个群集级别进行分配。

第 2 步：访问 Kubernetes API

根据集群的身份验证配置选择一个选项：

选项 A：适用于 Kubernetes 的 Azure 基于角色的访问控制 (RBAC)（推荐）

1. 如果尚未启用，请在集群上启用 Azure RBAC：Azure 门户 → AKS 集群 → 设置 → 安全配置 → 身份验证和授权 → 选择 Azure RBAC
2. 分配只读角色：Azure 门户 → 订阅 → 选择订阅 → 访问控制 (IAM) → 添加角色分配 → 选择 Azure Kubernetes 服务 RBAC 阅读器 → 分配给应用程序

这涵盖了订阅中的所有 AKS 集群。

选项 B：Azure 活动目录 (Azure AD) + Kubernetes RBAC

如果你的集群已经使用默认的 Azure AD 身份验证配置，并且你不想启用 Azure RBAC，则使用此选项。这需要按集群进行`kubectl`设置。

1. 将以下清单另存为devops-agent-reader.yaml：

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: devops-agent-reader
rules:
  - apiGroups: [""]
    resources: ["namespaces", "pods", "pods/log", "services", "events", "nodes"]
    verbs: ["get", "list"]
  - apiGroups: ["apps"]
    resources: ["deployments", "replicasets", "statefulsets", "daemonsets"]
    verbs: ["get", "list"]
  - apiGroups: ["metrics.k8s.io"]
    resources: ["pods", "nodes"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: devops-agent-reader-binding
subjects:
  - kind: User
    name: "<SERVICE_PRINCIPAL_OBJECT_ID>"
    apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: devops-agent-reader
  apiGroup: rbac.authorization.k8s.io
```

1. <SERVICE_PRINCIPAL_OBJECT_ID>替换为服务主体的对象 ID。要找到它，请执行以下操作：Azure Portal → Entra ID → 企业应用程序 → 搜索应用程序名称（要么是获得管理员同意的AWS DevOps 代理，要么是你自己的 Entra 应用程序进行应用程序注册）。
2. 适用于每个集群：

```
az aks get-credentials --resource-group <rg> --name <cluster-name>
kubectl apply -f devops-agent-reader.yaml
```

注意：不支持仅使用本地帐户（不使用 Azure AD）的集群。我们建议在集群上启用 Azure AD 集成以使用此功能。

权限最低的自定义角色（可选）

为了实现更严格的访问控制，你可以创建一个自定义 Azure 角色，该角色的范围仅限于 A AWS DevOps gent 使用的资源提供者，而不是广泛的读者角色：

```
{
  "Name": "AWS DevOps Agent - Azure Reader",
  "Description": "Least-privilege read-only access for AWS DevOps Agent incident
 investigations.",
  "Actions": [
    "Microsoft.AlertsManagement/*/read",
    "Microsoft.Compute/*/read",
    "Microsoft.ContainerRegistry/*/read",
    "Microsoft.ContainerService/*/read",
    "Microsoft.ContainerService/managedClusters/commandResults/read",
    "Microsoft.DocumentDB/*/read",
    "Microsoft.Insights/*/read",
    "Microsoft.KeyVault/vaults/read",
    "Microsoft.ManagedIdentity/*/read",
    "Microsoft.Monitor/*/read",
    "Microsoft.Network/*/read",
    "Microsoft.OperationalInsights/*/read",
    "Microsoft.ResourceGraph/resources/read",
    "Microsoft.ResourceHealth/*/read",
    "Microsoft.Resources/*/read",
    "Microsoft.Sql/*/read",
    "Microsoft.Storage/*/read",
    "Microsoft.Web/*/read"
  ],
  "NotActions": [],
  "DataActions": [],
  "NotDataActions": [],
  "AssignableScopes": [
    "/subscriptions/{your-subscription-id}"
  ]
}
```

将订阅与代理空间关联

在帐户级别注册 Azure 后，将特定订阅与您的代理空间相关联：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡
3. 在 “次要来源” 部分中，单击 “添加”
4. 选择 Azure
5. 为要关联的 Azure 订阅提供订阅 ID
6. 单击 “添加” 完成关联

你可以将多个订阅与同一个代理空间相关联，让代理在你的 Azure 环境中可见。

管理 Azure 资源连接

- 查看已连接的订阅-在 “功能” 选项卡中，“次要来源” 部分列出了所有连接的 Azure 订阅。
- 删除订阅-要断开订阅与代理空间的连接，请在 “辅助来源” 列表中选择该订阅并单击 “删除”。这不会影响账户级别的注册。
- 移除注册-要完全删除 Azure Cloud 注册，请转到功能提供者页面并删除注册。必须先移除所有代理空间关联。

连接 Azure DevOps

Azure DevOps 集成使 AWS DevOps 代理能够访问 Azure DevOps 组织中的存储库和管道执行历史记录。代理可以将代码更改和部署与操作事件关联起来，以帮助确定潜在的根本原因。

> 注意：Azure DevOps 管道可以使用 Azure 存储库或 Bitbucket 中的源代码。 GitHub无论源提供者如何，Azure DevOps 集成都提供对管道执行历史记录的访问权限。但是，要在调查期间访问实际源代码，必须通过支持的集成（例如）单独连接存储库the section called “正在连接 GitHub”。无法通过此集成直接访问 Bitbucket 中的源代码。

此集成遵循两个步骤的过程： DevOps 在 AWS 帐户级别注册 Azure，然后将特定项目与各个代理空间相关联。

先决条件

在连接 Azure 之前 DevOps，请确保你有：

- 访问 AWS DevOps 代理控制台
- 至少有一个包含存储库和管道历史记录的项目的 Azure DevOps 组织

- 向你的 Azure DevOps 组织添加用户的权限
- 对于管理员同意方法：有权在 Microsoft Entra ID 中进行管理员同意的帐户
- 对于应用程序注册方法：具有配置联合身份凭证权限的 Entra 应用程序，并在您的 AWS 账户中启用了出站联合身份验证

> 注意：您也可以从代理空间内开始注册。导航到 “管道” 部分，单击 “添加”，然后选择 Azure DevOps。如果 Azure DevOps 尚未注册，则控制台会引导你先完成注册。

DevOps 通过管理员同意注册 Azure

管理员同意方法在 AWS DevOps 代理托管的应用程序中使用基于同意的流程。

第 1 步：开始注册

1. 登录 AWS 管理控制台并导航到 AWS DevOps 代理控制台
2. 前往 “能力提供者” 页面
3. 找到 Azure DevOps 部分，然后单击 “注册”
4. 出现提示时输入你的 Azure DevOps 组织名称

第 2 步：完成管理员同意

1. 点击继续-您将被重定向到 Microsoft Entra 管理员同意页面
2. 使用有权执行管理员同意的用户主账号登录
3. 审查 AWS DevOps 代理申请并授予同意

步骤 3：完成用户授权

1. 管理员同意后，系统会提示您进行用户授权，以验证您作为授权租户成员的身份
2. 使用属于同一 Azure 租户的帐户登录
3. 授权后，您将重定向回 AWS DevOps 代理控制台，状态为成功

步骤 4：在 Azure 中授予访问权限 DevOps

请参阅 DevOps下面的在 Azure 中授予访问权限。添加用户时搜索AWS DevOps 代理。

DevOps 通过应用程序注册注册 Azure

应用程序注册在 Azure 资源和 Azure 之间共享 DevOps。如果你已经完成了 Azure 资源的应用程序注册，可以跳到在 Az ure 中授予访问权限 DevOps。

第 1 步：开始 ADO 应用程序注册

1. 在 AWS DevOps 代理控制台中，转到功能提供者页面
2. 找到 “A zure 云” 部分，然后单击 “注册”
3. 选择应用程序注册方法

第 2 步：创建和配置您的 Entra 应用程序

按照控制台中显示的说明执行以下操作：

1. 在您的 AWS 账户中启用出站身份联合（在 IAM 控制台中，前往账户设置 → 出站联合身份验证）
2. 在你的 Microsoft Entra ID 中创建 Entra 应用程序，或者使用现有的 Entra 应用程序
3. 在应用程序上配置联合身份凭证

第 3 步：提供注册详情

在注册表中填写以下内容：

- 租户 ID — 你的 Azure 租户标识符
- 租户名称-租户的显示名称
- 客户端 ID — Entra 应用程序的应用程序（客户端）ID
- 受众-联邦凭证的受众标识符

步骤 4：创建 IAM 角色

当您通过控制台提交注册时，将自动创建 IAM 角色。它允许 AWS DevOps 代理使用凭据并调用`sts:GetWebIdentityToken`。

第 5 步：完成注册

1. 在 AWS DevOps 代理控制台中确认配置
2. 单击 “提交” 完成注册

步骤 6：在 Azure 中授予访问权限 DevOps

请参阅 DevOps下面的在 Azure 中授予访问权限。添加用户时，搜索您在应用程序注册期间创建的 Entra 应用程序。

在 Azure 中授权访问权限 DevOps

注册后，向你的 Azure DevOps 组织授予应用程序访问权限。管理员同意和应用程序注册方法的此步骤相同。

1. 在 Azure 中 DevOps，前往 “组织设置” > “用户” > “添加用户”
2. 搜索应用程序（管理员同意AWS DevOps 代理或您自己的 Entra 应用程序注册应用程序）
3. 将访问级别设置为 “基本”
4. 在 “添加到项目” 下，选择您希望代理访问的项目
5. 在 Azure DevOps 群组下，选择项目读取器
6. 单击 “添加” 完成

> 安全要求：仅分配项目读者组。只读访问权限充当安全边界，将代理限制为只读操作，并限制间接提示注入攻击的影响。为群组分配写入或操作权限会大大增加提示注入的爆炸半径，并可能导致 Azure DevOps 资源受到损害。

将项目与代理空间关联

DevOps 在帐户级别注册 Azure 后，将特定项目与您的代理空间相关联：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡
3. 在 “管道” 部分中，单击 “添加”
4. DevOps从可用提供程序列表中选择 Azure
5. 从可用项目的下拉列表中选择项目
6. 单击 “添加” 完成关联

管理 Azure DevOps 连接

- 查看连接的项目-在 “功能” 选项卡中，“管道” 部分列出了所有连接的 Azure DevOps 项目。

- 移除项目-要断开项目与代理空间的连接，请在 “管道” 部分将其选中，然后单击 “删除”。
- 移除注册-要完全删除 Azure DevOps 注册，请转到功能提供者页面并删除注册。必须先移除所有代理空间关联。

连接到 CI/CD 管道

CI/CD 管道集成使 AWS DevOps 代理能够监控部署并在调查期间将代码更改与操作事件关联起来。通过连接您的 CI/CD 提供商，代理可以跟踪部署事件并将其与 AWS 资源关联，以帮助在事件响应期间确定潜在的根本原因。

AWS DevOps Agent 支持通过两步流程与流行 CI/CD 平台集成：

1. 账户级注册 — 在账户级别注册您的 CI/CD 提供商一次 AWS
2. Agent Space 连接 — 根据您的组织需求，将特定项目或存储库连接到各个代理空间

这种方法允许您在多个代理空间之间共享 CI/CD 提供商注册，同时保持对每个空间监控哪些项目的精细控制。

支持的 CI/CD 提供商

AWS DevOps 代理支持以下 CI/CD 平台：

- GitHub— 使用 AWS DevOps 代理 GitHub 应用程序连接来自 GitHub.com 的存储库。
- GitLab— 连接来自 GitLab.com的项目、托管 GitLab 实例或可公开访问的自托管 GitLab 部署。

主题

- the section called “正在连接 GitHub”
- the section called “正在连接 GitLab”

正在连接 GitHub

GitHub 集成使 AWS DevOps 代理能够在事件调查期间访问代码存储库并接收部署事件。此集成遵循两个步骤的过程：账户级注册 GitHub，然后将特定的存储库连接到各个代理空间。

AWS DevOps 代理同时支持 GitHub .com (SaaS) 和 GitHub 企业服务器（自托管）实例。

先决条件

在连接之前 GitHub，请确保您具有：

- 访问 AWS DevOps 代理管理员控制台
- 具有管理员权限的 GitHub 用户账户或组织
- 授权在您的账户或组织中安装 GitHub 应用程序

对于 GitHub 企业服务器，您还需要：

- 可通过 HTTPS 访问的 GitHub 企业服务器实例（版本 3.x 或更高版本）
- 您的 GitHub 企业服务器实例的 HTTPS 网址（例如，`https://github.example.com`）
- （可选）私有连接（如果您的 GitHub 企业服务器实例不可公开访问）

注册 GitHub （账户级别）

GitHub 在 AWS 账户级别注册，并在该账户中的所有代理空间之间共享。每个 AWS 账户只需要注册 GitHub 一次。

步骤 1：导航到管道提供商

1. 登录到 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台
3. 前往 “功能” 选项卡
4. 在 “管道” 部分中，单击 “添加”
5. GitHub从可用提供商列表中选择

如果 GitHub 尚未注册，系统将提示您先注册。

步骤 2：选择连接类型

在 “注册 GitHub 账户/组织” 屏幕上，选择您是以用户还是组织身份进行连接：

- 用户-包含用户名和个人资料的个人 GitHub 账户
- 组织 — 一个共享 GitHub 帐户，多个人可以同时在多个项目中进行协作

如果您要连接到 GitHub 企业服务器实例，请选中 “使用 GitHub 企业服务器” 复选框并输入您的实例的 HTTPS URL（例如`https://github.example.com`）。

如果您的 E GitHub nterprise Server 实例不可公开访问，则可以选择配置私有连接，以允许 AWS DevOps 代理安全地访问您的实例。有关更多信息，请参阅 the section called “连接到私人托管的工具”。

> **Note**
>
> 请勿在 URL 中包含`/api/v3`或任何尾随路径 — 仅输入基本 URL。

第 3 步：设置 GitHub 应用程序

单击 “提交” 开始应用程序设置过程。后续步骤会有所不同，具体取决于您连接的是 GitHub .com 服务器还是 GitHub 企业服务器。

对于 GitHub .com

1. 您将被重定向 GitHub 到安装 AWS DevOps 代理 GitHub 应用程序。
2. 选择要在哪个账户或组织中安装该应用程序。
3. 该应用程序允许 AWS DevOps 代理接收来自已连接存储库的事件，包括部署事件。

适用于 GitHub 企业服务器

GitHub Enterprise Server 使用 GitHub 应用程序清单流程，该流程会自动在您的实例上设置新 GitHub 应用程序。这涉及到您的 GitHub 企业服务器实例的两次重定向。

1. 您的浏览器将被重定向到您的 GitHub 企业服务器实例的 “创建 GitHub 应用程序” 页面。
2. 您将看到预先填充的应用程序名称。可以根据需要随时更改名称。单击 “创建 GitHub 应用程序”。
3. 您将被重定向回 AWS DevOps Agent，代理将清单代码交换为应用程序凭据。

步骤 4：选择存储库并完成安装

1. 您将看到该 GitHub 应用程序的 “安装和授权” 页面。
2. 选择允许该应用程序访问的存储库：
   - 所有存储库-授予对所有当前和将来存储库的访问权限

- 仅选择存储库-从您的账户或组织中选择特定的存储库

3. 单击 “安装并授权”。
4. 您将被重定向回 AWS DevOps 代理控制台，该控制台 GitHub 将显示为已注册账号。

将存储库连接到代理空间

在账户 GitHub 级别注册后，您可以将特定的存储库连接到各个代理空间：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡
3. 在 “管道” 部分中，单击 “添加”
4. GitHub从可用提供商列表中选择
5. 选择与该代理空间相关的存储库子集
6. 单击 “添加” 完成连接

您可以根据组织需求将不同的存储库集连接到不同的代理空间。

了解 GitHub 应用程序

AWS DevOps 代理 GitHub 应用程序：

- 请求对仓库的只读访问权限
- 接收部署事件和其他存储库事件
- 允许 AWS DevOps Agent 将代码更改与操作事件关联起来
- 可以通过您的 GitHub 设置随时卸载

对于 GitHub 企业服务器， GitHub 应用程序是在注册期间在您的实例上自动创建的。您可以管理应用程序的存储库访问权限，也可以通过 “设置” > “应用程序” > “已安装的 GitHub 应用程序” 将其卸载。要完全删除应用程序定义，请前往 “设置” > “开发者设置” > “ GitHub 应用程序”。

管理 GitHub 连接

- 更新存储库访问权限-要更改 GitHub 应用程序可以访问的存储库，请转到您的 GitHub 帐户或组织设置（或您的 E GitHub nterprise Server 实例设置），导航到已安装的 GitHub 应用程序，然后修改 AWS DevOps 代理应用程序配置。

- 查看连接的存储库-在 AWS DevOps 代理控制台中，选择您的代理空间，然后转到 “功能” 选项卡，在 “管道” 部分中查看连接的存储库。
- 删除 GitHub 连接-要断开与 GitHub 代理空间的连接，请在 “管道” 部分中选择该连接，然后单击 “删除”。要完全卸载该 GitHub 应用程序，请将其从您的 GitHub 帐户或组织设置中卸载。对于 E GitHub nterprise Server，由于 GitHub 应用程序是在注册期间直接在您的实例上创建的，因此您可以选择通过执行以下两项操作来完全清理应用程序：
  - 卸载应用程序 — 前往 “设置” > “应用程序” > “已安装的 GitHub 应用程序”，在应用程序上单击 “配置”，然后将其卸载。
  - 删除应用程序-前往 “设置” > “开发者设置” > “ GitHub 应用程序”，选择应用程序，转到 “高级” 选项卡，然后选择 “删除 GitHub 应用程序”。警告：删除 GitHub 应用程序是永久性的，无法撤消。如果将其删除，则需要从头开始在 AWS DevOps 代理控制台中重新注册 GitHub 企业服务器才能创建新应用程序。

正在连接 GitLab

GitLab 集成使 AWS DevOps 代理能够监控来自 GitLab 管道的部署，以便在事件响应期间为因果调查提供信息。此集成遵循两个步骤的过程：账户级注册 GitLab，然后将特定项目连接到各个代理空间。

注册 GitLab （账户级别）

GitLab 在 AWS 账户级别注册，并在该账户中的所有代理空间之间共享。然后，各个代理空间可以选择适用于其代理空间的特定项目。

步骤 1：导航到管道提供商

1. 登录到 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台
3. 转到能力提供者页面（可从侧面导航栏访问）
4. GitLab在 Pi pelin e 下的 “可用提供商” 部分中查找，然后点击注册

步骤 2：配置 GitLab 连接

在 GitLab 注册页面上，配置以下内容：

连接类型-选择您是以个人还是群组的身份进行连接：

- 个@@ 人（默认）-包含 GitLab 用户名和个人资料的个人用户帐户

- 群组 — 在群组中 GitLab，您可以使用群组同时管理一个或多个相关项目

GitLab 实例类型-选择要连接的 GitLab 实例类型：

- GitLab.com（默认）-公共 GitLab 服务
- 可公开访问的自托管 GitLab — 选中使用 GitLab 自托管终端节点复选框并提供您的 GitLab 实例的 URL

> **Note**
>
> 目前，仅支持可公开访问的 GitLab 实例。

访问令牌 — 提供 GitLab 个人访问令牌：

1. 在单独的浏览器选项卡中，登录到您的 GitLab 帐户
2. 导航到您的用户设置并选择访问令牌
3. 使用以下权限创建新的个人访问令牌：
   - `read_repository`— 访问存储库内容所必需的
   - `read_virtual_registry`— 访问虚拟注册表信息所必需的
   - `read_registry`— 访问注册表信息所必需的
   - `api`— 读取和写入 API 访问权限所必需的
   - `self_rotate`-轮换代币是必需的。 AWS DevOps 代理目前不支持此功能，但稍后将支持该功能。现在添加可以防止将来需要创建新代币。
4. 将代币到期时间设置为自当前日期起最长 365 天
5. 复制生成的令牌
6. 返回 AWS DevOps 代理控制台
7. 将令牌粘贴到 “访问令牌” 字段

第 3 步：完成注册

（可选）标签-为组织目的向 GitLab 注册添加 AWS 标签。

单击 “下一步” 查看您的配置，然后单击 “提交” 以完成 GitLab 注册过程。系统将验证您的访问令牌并建立连接。

将项目连接到代理空间

在账户 GitLab 级别注册后，您可以将特定项目关联到各个代理空间：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡
3. 在 “管道” 部分中，单击 “添加”
4. GitLab从可用提供商列表中选择
5. 选择与您的代理空间相关的 GitLab 项目
6. 单击 “保存”

AWS DevOps 代理将监控这些项目的部署情况，以便 GitLab 为因果调查提供信息。

管理 GitLab 连接

- 更新访问令牌-如果您的访问令牌过期或需要更新，则可以在 AWS DevOps 代理控制台中通过修改账户级别的 GitLab 注册来对其进行更新。
- 查看关联项目-在 AWS DevOps 代理控制台中，选择您的代理空间，然后转到功能选项卡，在 “管道” 部分中查看连接的项目。
- 移除 GitLab 连接-要断开 GitLab 项目与代理空间的连接，请在 “管道” 部分中选择该连接，然后单击 “删除”。要完全删除 GitLab 注册，请先将其从所有代理空间中删除，然后在账户级别删除注册。

连接 MCP 服务器

模型上下文协议 (MCP) 服务器通过提供对来自外部可观测性工具、自定义监控系统和操作数据源的数据的访问权限来扩展 AWS DevOps 代理的调查能力。本指南介绍如何将 MCP 服务器连接到 AWS DevOps 代理。

要求

在连接 MCP 服务器之前，请确保您的服务器满足以下要求：

- 可流式传输的 HTTP 传输协议 — 仅支持实现可流式传输 HTTP 传输协议的 MCP 服务器。
- 身份验证支持-您的 MCP 服务器必须支持 OAuth 2.0 身份验证流程或基于 API 密钥/令牌的身份验证。

安全注意事项

将 MCP 服务器连接到 AWS DevOps Agent 时，请考虑以下安全方面：

- 工具许可名单 — 您应该只将代理空间所需的特定工具列入许可名单，而不是公开 MCP 服务器上的所有工具。有关如何允许每个代理空间列出工具，请参阅在代理空间中配置 MCP 工具。

请注意，任何 MCP 刀具的最大刀具长度均为 64。

- 提示注入风险 — 自定义 MCP 服务器可能会带来额外的提示注入攻击风险。有关更多信息，请参见提示注入保护： AWS DevOps 客户端安全。
- 只读工具和访问权限- 仅允许将只读 MCP 工具列入许可名单，并确保仅允许身份验证凭据进行只读访问。

有关即时注入和责任共担模型的更多信息，请参阅AWS DevOps 代理安全。

> **Note**
>
> 如果您的 MCP 服务器位于专用网络上，请参阅 the section called “连接到私人托管的工具”

注册 MCP 服务器（账户级）

MCP 服务器在 AWS 账户级别注册，并在该账户中的所有代理空间之间共享。然后，各个代理空间可以从每台 MCP 服务器中选择他们需要的特定工具。

步骤 1：MCP 服务器详细信息

1. 登录 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台
3. 转到能力提供者页面（可从侧面导航栏访问）
4. 在 “可用提供商” 部分中找到 MCP 服务器，然后单击 “注册”
5. 在 MCP 服务器详细信息页面上，输入以下信息：
   - 名称-输入 MCP 服务器的描述性名称
   - 端点网址-输入 MCP 服务器端点的完整 HTTPS 网址
   - 描述（可选）-添加描述以帮助确定服务器的用途

- 启用动态客户端注册-如果要允许 AWS DevOps 代理自动向 MCP 服务器的授权服务器注册，请选中此复选框

6. 单击下一步

> **Note**
>
> MCP 服务器端点 URL 将显示在您账户的 AWS CloudTrail 日志中。

步骤 2：授权流程

为您的 MCP 服务器选择身份验证方法：

OAuth 客户端凭证-如果您的 MCP 服务器使用 OAuth 客户端凭据流：

1. 选择OAuth 客户凭证
2. 单击下一步

OAuth 3LO（三腿 OAuth）— 如果您的 MCP 服务器使用 OAuth 3LO 进行身份验证：

1. 选择 OAuth 3 LO
2. 单击下一步

API 密钥 — 如果您的 MCP 服务器使用 API 密钥身份验证：

1. 选择 API 密钥
2. 单击下一步

步骤 3：授权配置

根据所选的身份验证方法配置其他授权参数：

对于 OAuth 客户凭证：

1. 客户端 ID-输入客户端的 OAuth 客户端 ID
2. 客户机密钥-输入 OAuth 客户端的客户机密钥
3. 交易所 URL — 输入令 OAuth 牌交换端点 URL

4. 交换参数-输入用于通过服务进行身份验证的 OAuth 令牌交换参数
5. 添加作用域-为身份验证添加 OAuth 范围
6. 单击下一步

对于 OAuth 3LO：

1. 客户端 ID-输入客户端的 OAuth 客户端 ID
2. 客户密钥-如果您的客户需要，请输入 OAuth 客户端的 OAuth 客户机密钥
3. 交易所 URL — 输入令 OAuth 牌交换端点 URL
4. 授权 URL-输入 OAuth 授权端点 URL
5. Code Chall enge Support-如果您的 OAuth 客户支持代码挑战，请选中此复选框
6. 添加作用域-为身份验证添加 OAuth 范围
7. 单击下一步

对于 API 密钥：

1. 输入 API 密钥名称
2. 输入请求中将包含 API 密钥的标头的名称
3. 输入你的 API 密钥值
4. 单击下一步

第 4 步：查看并提交

1. 查看所有 MCP 服务器配置详细信息
2. 单击 “提交” 完成注册
3. AWS DevOps 代理将验证与您的 MCP 服务器的连接
4. 成功验证后，您的 MCP 服务器将在账户级别注册

在代理空间中配置 MCP 工具

在帐户级别注册 MCP 服务器后，您可以配置该服务器中的哪些工具可供特定的代理空间使用：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间

2. 前往 “功能” 选项卡
3. 在 “MCP 服务器” 部分中，单击 “添加”
4. 选择要连接到此代理空间的已注册 MCP 服务器
5. 配置此 MCP 服务器上的哪些工具应可供代理空间使用：
   - 允许所有工具-使 MCP 服务器中的所有工具都可用
   - 选择特定工具-允许您选择要列入许可名单的工具
6. 单击 “添加” 将 MCP 服务器连接到您的代理空间

AWS DevOps 现在，在代理空间进行调查期间，代理将能够使用您的 MCP 服务器上的许可名单工具。

管理 MCP 服务器连接

更新身份验证凭证-如果需要更新您的身份验证凭据，则需要重新注册您的 MCP 服务器。导航到 AWS DevOps 代理控制台中的 “功能提供者” 页面，找到您的 MCP 服务器，移除所有活动关联，然后单击 “取消注册”。接下来，使用新的身份验证凭据注册您的 MCP 服务器，并与您的代理空间重新创建所有必要的关联。

查看连接的 MCP 服务器-要查看连接到您的代理空间的所有 MCP 服务器，请选择您的代理空间，转到 “功能” 选项卡，然后查看 “MCP 服务器” 部分。您也可以在此处更新所选工具。

删除 MCP 服务器连接 -要断开 MCP 服务器与代理空间的连接，请在 “MCP 服务器” 部分中选择该服务器，然后单击 “删除”。要完全删除 MCP 服务器注册，请先将其从所有代理空间中删除，然后删除账户级别的注册。

相关主题

- AWS DevOps 代理中的安全
- 设置代理空间
- 即时注射保护

关联多个 AWS 账户

辅助 AWS 账户允许 AWS DevOps 代理调查组织中多个 AWS 账户的资源。当您的应用程序跨多个账户时，添加辅助账户可确保代理在事件调查期间可以看到所有相关资源。对构成应用程序的账户和资源的更大访问权限可确保更高的调查准确性。

先决条件

在添加辅助 AWS 账户之前，请确保您已经：

- 使用主账户访问 AWS DevOps 代理控制台
- 对辅助 AWS 账户的管理权限
- 在辅助账户中创建角色的 IAM 权限

添加辅助 AWS 账户

除了以下步骤外，您还可以使用the section called “AWS DevOps 代理 CLI 入门指南”以编程方式添加辅助帐户。

步骤 1：启动辅助账户配置

1. 登录 AWS 管理控制台并导航到 AWS DevOps 代理控制台
2. 选择您的代理空间
3. 前往 “功能” 选项卡
4. 在 “云” 部分中，找到 “次要来源” 子部分
5. 单击 “添加”

步骤 2：指定角色名称

1. 在 “为你的角色命名” 字段中，输入你将在辅助账户中创建的角色的名称
2. 请注意此名称——在辅助账户中创建角色时，您将再次使用该名称
3. 复制控制台中提供的信任策略并将其保存在暂存空间中

步骤 3：在辅助账户中创建角色

1. 打开新的浏览器选项卡，然后使用辅助 AWS 账户登录 IAM 控制台
2. 导航到 IAM > 角色 > 创建角色
3. 选择 “自定义信任策略”
4. 粘贴您在步骤 2 中复制的信任策略
5. 单击下一步

步骤 4：附加 AWS 托管策略

1. 在 “权限策略” 部分中，搜索 AIOpsAssistantPolicy
2. 选中AIOpsAssistantPolicy托管策略旁边的复选框
3. 单击下一步

步骤 5：命名并创建角色

1. 在 “角色名称” 字段中，输入您在步骤 2 中提供的相同角色名称
2. （可选）添加描述以帮助确定角色的用途
3. 查看信任策略和附加权限
4. 单击 “创建角色”

步骤 6：附加内联策略

1. 在 IAM 控制台中，找到并选择您刚刚创建的角色
2. 前往 “权限” 选项卡
3. 单击 “添加权限” > “创建内联策略”
4. 切换到 JSON 选项卡
5. 粘贴您在步骤 2 中保存的策略
6. 将策略粘贴到 IAM 控制台的 JSON 编辑器中
7. 单击下一步
8. 为内联策略提供名称（例如，DevOpsAgentInlinePolicy“”）
9. 点击创建策略

步骤 7：完成配置

1. 使用主账号返回 AWS DevOps 代理控制台
2. 单击 “下一步” 完成辅助账户配置
3. 验证连接状态是否显示为 “活动”

了解必需的策略

AWS DevOps 代理需要三个策略组件才能访问辅助账户中的资源：

- 信任策略-允许主账户中的 AWS DevOps 代理在辅助账户中扮演该角色。这就建立了账户之间的信任关系。
- AIOpsAssistantPolicy （ AWS 托管策略 ）— 提供 AWS DevOps 代理调查辅助账户资源所需的核心只读权限。此政策由新增功能维护 AWS 和更新。
- 内联策略-提供特定于您的 Agent Space 配置的其他权限。此策略是根据您的 Agent Space 设置生成的，可能包括特定集成或功能的权限。

在主账户中，A AWS DevOps gent IAM 角色必须能够代入在辅助账户中创建的角色。

管理辅助账户

- 查看已连接的帐户-在 “功能” 选项卡中，“次要来源” 子部分列出了所有已连接的次要帐户及其连接状态。
- 更新 IAM 角色-如果您需要修改权限，请更新辅助账户中附加到该角色的内联策略。更改将立即生效。
- 删除次要帐户-要断开次要帐户的连接，请在次要来源列表中将其选中，然后单击 “删除”。这不会删除辅助账户中的 IAM 角色。

连接遥测源

AWS DevOps Agent 提供了三种连接到遥测源的方式。

内置双向集成

目前， AWS DevOps Agent 通过内置的双向集成支持 Dynatrace 用户，可实现以下功能：

- 拓扑资源映射- AWS DevOps 代理将通过代理 AWS DevOps 托管的 Dynatrace MCP DevOps 服务器使用实体和关系来增强您的代理空间拓扑。
- 自动触发调查 ——可以将 Dynatrace 工作流程配置为触发事件解决方案 Dynatrace Problems 中的调查。
- 遥测内省—— AWS DevOps 代理可以在通过代理托管的 Dynatrace MCP 服务器调查问题时反省 Dynatrace 遥测数据。 AWS DevOps

- 状态更新- AWS DevOps 代理将在 Dynatrace 用户界面上发布关键调查结果、根本原因分析和生成的缓解计划。

要了解双向集成，请参阅

- the section called “连接 Dynatrace”

内置单向集成

目前， AWS DevOps Agent 通过 AWS CloudWatch内置的单向集成支持 Datadog、Grafana、New Relic 和 Splunk 用户。

安全最佳实践：为内置单向集成配置凭据时，我们建议将 API 密钥和令牌限定为只读访问权限。 AWS DevOps 代理仅使用这些凭据进行遥测内省，不需要对遥测提供商的写入权限。

AWS CloudWatch 内置的单向集成无需额外设置，可实现以下功能：

- 拓扑资源映射- AWS DevOps 代理将通过您配置的主云和辅助 AWS 云帐户使用实体和关系来增强您的 DevOps 代理空间拓扑。
- 遥测内省- AWS DevOps 代理可以在通过主云账户和 AWS CloudWatch 辅助云账户配置期间提供的 IAM 角色调查问题时对遥测进行内省。 AWS

内置的 Datadog、Grafana、New Relic 和 Splunk 单向集成需要设置并启用以下功能：

- 自动触发调查 ——可以将 Datadog、Grafana、New Relic 和 Splunk 事件配置为通过代理 webhook 触发 AWS DevOps 代理事件解决调查。 AWS DevOps
- 遥测内省—— AWS DevOps 代理可以在通过每个提供商的远程 MCP 服务器调查问题时反省 Datadog、Grafana、New Relic 和 Splunk 遥测。

要了解单向集成，请参阅以下内容：

- the section called “正在连接 DataDog”
- the section called “连接 Grafana”
- the section called “连接新遗物”
- the section called “连接 Splunk”

Bring-your-own 遥测源

对于任何其他遥测来源，包括 Prometheus 指标，您可以 AWS DevOps 利用 Agent 对 webhook 和 MCP 服务器集成的支持。

要了解 bring-your-own集成，请参阅以下内容

- the section called “通过 Webhook 调用 DevOps 代理”
- the section called “连接 MCP 服务器”

连接 Dynatrace

内置、双向集成

目前， AWS DevOps Agent 通过内置的双向集成支持 Dynatrace 用户，可实现以下功能：

- 拓扑资源映射- AWS DevOps 代理将使用您的 Dynatrace 环境中可用的实体和关系来增强您的 DevOps 代理空间拓扑。
- 自动触发调查 ——可以将 Dynatrace 工作流程配置为触发事件解决方案 Dynatrace Problems 中的调查。
- 遥测内省—— AWS DevOps 代理可以在通过代理托管的 Dynatrace MCP 服务器调查问题时反省 Dynatrace 遥测数据。 AWS DevOps
- 状态更新- AWS DevOps 代理将在 Dynatrace 用户界面上发布关键调查结果、根本原因分析和生成的缓解计划。

信息载入

入职流程

启动您的 Dynatrace 可观测性系统包括三个阶段：

1. Connect-通过配置账户访问凭证，与您可能需要的所有环境建立与 Dynatrace 的连接
2. 启用-在具有特定 Dynatrace 环境的特定代理空间中激活 Dynatrace
3. 配置您的 Dynatrace 环境-下载工作流程和仪表板并导入 Dynatrace，记下 webhook 的详细信息以便在指定的代理空间中触发调查

步骤 1：Connect

与你的 Dynatrace 环境建立连接

配置

1. 转到能力提供者页面（可从侧面导航栏访问）
2. 在 “遥测” 下的 “可用提供商” 部分中找到 Dynatrac e，然后单击 “注册”
3. 在 Dynatrace 中创建具有详细权限的 OAuth 客户端。
   a. 请参阅 Dynatrac e 文档
   b. 准备就绪后，按下一步
   c. 您可以将多个 Dynatrace 环境连接起来，然后将范围连接到您可能拥有的每个 DevOps 代理空间的特定环境。
4. 在 OAuth 客户端设置中输入你的 Dynatrace 详细信息：
   - 客户名称
   - 客户端 ID
   - 客户机密钥
   - 账号 URN
5. 单击下一步
6. 查看并添加

步骤 2：启用

在特定的代理空间中激活 Dynatrace 并配置适当的作用域

配置

1. 在座席空间页面上，选择一个座席空间，然后按查看详情
2. 选择 “权能” 选项卡
3. 找到 “遥测” 部分，按添加
4. 你会注意到 Dynatrace 的状态为 “已注册”。单击 “添加” 将其添加到您的代理空间
5. Dynatrace 环境 ID-提供你想与该代理空间关联的 Dynatrace 环境 ID。 DevOps
6. 输入一个或多个 Dynatrace 实体 IDs ——这些帮助 DevOps 代理发现你最重要的资源，例如服务或应用程序。如果您不确定，可以按删除。
7. 查看并按保存

8. 复制 Webhook 网址和 Webhook 密钥。要将这些凭证添加到 Dynatrace，请参阅 Dynatrace 文档。

第 3 步：配置你的 Dynatrace 环境

要完成 Dynatrace 设置，你需要在 Dynatrace 环境中执行某些设置步骤。按照 Dynatrac e 文档中的说明进行操作。

支持的事件架构

AWS DevOps Agent 使用 webhook 支持来自 Dynatrace 的两种类型的事件。下面记录了支持的事件架构：

事件事件

事件事件用于触发调查。事件架构是：

```
{
    "event.id": string;
    "event.status": "ACTIVE" | "CLOSED";
    "event.status_transition": string;
    "event.description": string;
    "event.name": string;
    "event.category": "AVAILABILITY" | "ERROR" | "SLOWDOWN" | "RESOURCE_CONTENTION" |
 "CUSTOM_ALERT" | "MONITORING_UNAVAILABLE" | "INFO";
    "event.start"?: string;
    "affected_entity_ids"?: string[];
}
```

缓解事件

缓解事件用于触发生成缓解报告，以便调查后续步骤。事件架构是：

```
{
    "task_id": string;
    "task_version": number;
    "event.type": "mitigation_request";
}
```

移除

遥测源在两个级别上连接，分别是代理空间级别和账户级别。要将其完全删除，必须先将其从所有使用它的代理空间中删除，然后才能将其取消注册。

步骤 1：从代理空间中移除

1. 在座席空间页面上，选择一个座席空间，然后按查看详情
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 选择 Dynatrace
5. 按移除

第 2 步：注销账号

1. 转到能力提供者页面（可从侧面导航栏访问）
2. 滚动到 “当前注册” 部分。
3. 检查代理空间计数是否为零（如果不是，请在其他代理空间中重复上述步骤 1）
4. 按 Dynatrace 旁边的 “取消注册”

正在连接 DataDog

内置，单向集成

目前， AWS DevOps Agent 通过内置的单向集成支持 Datadog 用户，支持以下功能：

- 自动触发调查-可以将 Datadog 事件配置为通过 AWS DevOps 代理 webhook 触发 AWS DevOps 代理事件解决调查。
- 遥测内省 —— AWS DevOps 代理可以在通过每个提供商的远程 MCP 服务器调查问题时反省 Datadog 遥测数据。

信息载入

步骤 1：Connect

使用账户访问凭证与您的 Datadog 远程 MCP 端点建立连接

配置

1. 转到能力提供者页面（可从侧面导航栏访问）
2. 在 “遥测” 下的 “可用提供商” 部分中找到 Datadog，然后单击 “注册”

3. 输入您的 Datadog MCP 服务器详细信息：
   - 服务器名称-唯一标识符（例如 my-datadog-server）
   - 端点 URL-您的 Datadog MCP 服务器端点。端点网址因您的 Datadog 网站而异。请参阅下面的 Datadog 站点端点表。
   - 描述-可选的服务器描述
4. 单击下一步
5. 审核并提交

Datadog 网站端点

MCP 端点网址因您的 Datadog 网站而异。要识别您的网站，请在登录 Datadog 时在浏览器中查看网址，或者参阅访问 Dat adog 网站。

Datadog 网站	网站域名	MCP 端点网址
US1（默认）	`datadoghq.com`	`https://mcp.datadoghq.com/api/unstable/mcp-server/mcp`
US3	`us3.datadoghq.com`	`https://mcp.us3.datadoghq.com/api/unstable/mcp-server/mcp`
US5	`us5.datadoghq.com`	`https://mcp.us5.datadoghq.com/api/unstable/mcp-server/mcp`
EU1	`datadoghq.eu`	`https://mcp.datadoghq.eu/api/unstable/mcp-server/mcp`
AP1	`ap1.datadoghq.com`	`https://mcp.ap1.datadoghq.com/api/un`

Datadog 网站	网站域名	MCP 端点网址
		`stable/mcp-server/mcp`
AP2	`ap2.datadoghq.com`	`https://mcp.ap2.datadoghq.com/api/unstable/mcp-server/mcp`

Authorization

通过以下方式完成 OAuth 授权：

- 在 Dat OAuth adog 页面上以您的用户身份进行授权
- 如果未登录，请单击 “允许”、“登录”，然后单击 “授权”

配置完成后，Datadog 将在所有代理空间中使用。

步骤 2：启用

DataDog 在特定的代理空间中激活并配置适当的作用域

配置

1. 在座席空间页面上，选择一个座席空间，然后按查看详情（如果您尚未创建座席空间，请参阅the section called “创建代理空间”）
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 按添加
5. 选择 Datadog
6. 下一步
7. 查看并按保存
8. 复制 Webhook 网址和 API 密钥

步骤 3：配置 Webhook

使用 Webhook 网址和 API 密钥，您可以将 Datadog 配置为发送事件以触发调查，例如从警报中触发调查。

为确保 DevOps 代理可以使用发送的事件，请确保传输到 webhook 的数据与下面指定的数据架构相匹配。 DevOps 代理可以忽略与此架构不匹配的事件。

设置方法和标题

```
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

将正文作为 JSON 字符串发送。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

使用 Datadog webh https://docs.datadoghq.com/integrations/ooks/ 发送网络挂钩（注意选择不授权，改用自定义标题选项）。

了解更多：Datadog 远程 MCP 服务器

移除

遥测源在两个级别上连接，分别是代理空间级别和账户级别。要将其完全删除，必须先将其从所有使用它的代理空间中删除，然后才能将其取消注册。

步骤 1：从代理空间中移除

1. 在座席空间页面上，选择一个座席空间，然后按查看详情
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 选择 Datadog
5. 按移除

第 2 步：注销账号

1. 转到能力提供者页面（可从侧面导航栏访问）
2. 滚动到 “当前注册” 部分。
3. 检查代理空间计数是否为零（如果不是，请在其他代理空间中重复上述步骤 1）
4. 按 Datadog 旁边的 “取消注册”

连接 Grafana

Grafana 集成 AWS DevOps 使代理能够在事件调查期间从您的 Grafana 实例中查询指标、仪表板和警报数据。此集成遵循两个步骤的过程：Grafana的账户级注册，然后将其连接到各个代理空间。

为了提高安全性，Grafana 集成仅启用只读工具。写入工具已禁用，无法启用。这意味着代理可以查询和读取您的 Grafana 实例中的数据，但不能创建、修改或删除任何 Grafana 资源，例如仪表板、警报或注释。有关更多信息，请参阅AWS DevOps 客户端中的安全。

Grafana 要求

在连接 Grafana 之前，请确保您已具备以下条件：

- Grafana 版本 9.0 或更高版本。由于缺少 API 端点，某些功能，尤其是与数据源相关的操作，可能无法在早期版本中正常运行。
- 可通过 HTTPS 访问的 Grafana 实例。支持公共网络和私有网络端点。通过私有网络连接，您的 Grafana 实例可以托管在没有公共互联网访问权限的 VPC 内。有关更多信息，请参阅 the section called “连接到私人托管的工具”。
- 一个 Grafana 服务账户，其访问令牌具有适当的读取权限

注册 Grafana（账户级别）

Grafana 在账户级别注册，并在 AWS 该账户中的所有代理空间之间共享。

第 1 步：配置 Grafana

1. 登录到 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台
3. 转到 “能力提供者” 页面（可从侧面导航栏访问）
4. 在 “遥测” 下的 “可用提供商” 部分中找到 Grafana，然后单击 “注册”
5. 在配置 Grafana 页面上，输入以下信息：
   - 服务名称（必填）-仅使用字母数字字符、连字符和下划线输入 Grafana 服务器的描述性名称。例如 `my-grafana-server`。
   - Grafana 网址（必填）— 输入您的 Grafana 实例的完整 HTTPS 网址。例如 `https://myinstance.grafana.net`。
   - 服务账户访问令牌（必填）— 输入 Grafana 服务账户访问令牌。代币通常以开头`glsa_`。要创建服务帐号令牌，请导航至您的 Grafana 实例，转至管理 > 服务帐号，创建一个具有 Viewer 角色的服务帐号，然后生成令牌。
   - 描述（可选）-添加描述以帮助确定服务器的用途。例如 `Production Grafana server for monitoring`。
6. （可选）为组织目的向注册添加 AWS 标签。
7. 单击下一步

第 2 步：查看并提交 Grafana 注册信息

1. 查看所有 Grafana 配置细节
2. 单击 “提交” 完成注册
3. 成功注册后，Grafana 将显示在 “能力提供者” 页面的 “当前已注册” 部分

将 Grafana 添加到代理空间

在账户级别注册 Grafana 后，你可以将其连接到个人代理空间：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡

3. 在 “遥测” 部分，单击 “添加”

4. 从可用提供商列表中选择 Grafana

5. 单击 “保存”

配置 Grafana 警报 webhook

您可以将 Grafana 配置为在警报触发时自动 AWS DevOps 触发代理调查，方法是通过 Grafana 联系点发送 webhook。有关 Webhook 身份验证方法和凭据管理的详细信息，请参阅。the section called “通过 Webhook 调用 DevOps 代理”

步骤 1：创建自定义通知模板

在您的 Grafana 实例中，导航到警报 > 联系方式 > 通知模板，然后使用以下内容创建一个新模板：

```
{{ define "devops-agent-payload" }}
{
  "eventType": "incident",
  "incidentId": "{{ (index .Alerts 0).Labels.alertname }}-{{ (index .Alerts
 0).Fingerprint }}",
  "action": "{{ if eq .Status "resolved" }}resolved{{ else }}created{{ end }}",
  "priority": "{{ if eq .Status "resolved" }}MEDIUM{{ else }}HIGH{{ end }}",
  "title": "{{ (index .Alerts 0).Labels.alertname }}",
  "description": "{{ (index .Alerts 0).Annotations.summary }}",
  "service": "{{ if (index .Alerts 0).Labels.job }}{{ (index .Alerts 0).Labels.job }}
{{ else }}grafana{{ end }}",
  "timestamp": "{{ (index .Alerts 0).StartsAt }}",
  "data": {
    "metadata": {
      {{ range $k, $v := (index .Alerts 0).Labels }}
      "{{ $k }}": "{{ $v }}",
      {{ end }}
      "_source": "grafana"
    }
  }
}
{{ end }}
```

此模板将 Grafana 警报格式化为代理所期望的 webhook 有效载荷结构。 AWS DevOps 它将警报标签、注释和状态映射到相应的字段中，并将所有警报标签作为元数据包括在内。

注意：此模板仅处理群组中的第一个警报。默认情况下，Grafana 将多个触发警报分组为一个通知。要确保每个警报都是单独发送的，请将通知策略配置为按分组`alertname`。此外，此模板不会对标签值或注释中的特殊 JSON 字符进行转义。确保警报标签和`summary`注释中不包含双引号或换行符等字符，这会生成无效的 JSON。

第 2 步：创建 webhook 联系人

1. 在 Grafana 中，导航至 “警报” > “联系人”，然后单击 “添加联系人”
2. 选择 Webhook 作为集成类型
3. 将网址设置为你的 AWS DevOps 代理 webhook 端点
4. 在 “可选 Webhook 设置” 下，根据您的 Webhook 类型配置身份验证标头。有关详细信息，请参阅 Webhook 身份验证方法。
5. 将 “消息” 字段设置为使用您的自定义模板：`{{ template "devops-agent-payload" . }}`
6. 单击 “保存联系人”

步骤 3：将联系人分配给通知策略

1. 导航至 “警报” > “通知策略”
2. 编辑现有策略或创建新策略
3. 将联系点设置为您创建的 webhook 联系点
4. 点击 “保存政策”

当匹配的警报触发时，Grafana 会将格式化的有效负载发送 AWS DevOps 给 Agent，代理将自动开始调查。

限制

- ClickHouse 数据源工具-目前不支持 ClickHouse 数据源工具。
- 主动预防事件 — 目前the section called “主动预防事故”不使用 Grafana 工具。计划在 future 版本中提供支持。

亚马逊 Managed Grafana 注意事项

如果您使用的是亚马逊托管 Grafana (AMG)，请注意以下限制：

- 不支持 Webhook 联系点 — AMG 目前在其警报配置中不支持 webhook 联系点。您不能使用 AMG 直接向代理发送警报 Webhook。 AWS DevOps 有关详情，请参阅 Amazon Managed Grafana 中的提醒联系人。
- 服务账户令牌到期 — AMG 服务账户令牌的有效期最长为 30 天。在代币到期之前，您需要轮换代币并在代理 AWS DevOps 中更新您的 Grafana 注册。有关如何更新凭据的信息，请参阅管理 Grafana 连接。有关 AMG 代币限制的详细信息，请参阅亚马逊托管 Grafana 中的服务账户。

管理 Grafana 连接

- 更新凭证-如果您的服务帐号令牌过期或需要更新，请从 “功能提供商” 页面注销 Grafana，然后使用新令牌重新注册。
- 查看连接的实例-在 AWS DevOps 代理控制台中，选择您的代理空间，然后转到功能选项卡以查看连接的遥测源。
- 移除 Grafana — 要断开 Grafana 与代理空间的连接，请在 “遥测” 部分将其选中，然后单击 “删除”。要完全删除注册，请先将其从所有代理空间中删除，然后从 “能力提供者” 页面取消注册。

连接新遗物

内置，单向集成

目前， AWS DevOps Agent 通过内置的单向集成支持 New Relic 用户，支持以下功能：

- 自动触发调查-可以将新的 Relic 事件配置为通过代理 webhook 触发 AWS DevOps AWS DevOps 代理事件解决调查。
- 遥测内省 —— AWS DevOps 代理可以在通过每个提供商的远程 MCP 服务器调查问题时反省 New Relic 遥测数据。

信息载入

步骤 1：Connect

使用账户访问凭证与你的 New Relic 远程 MCP 端点建立连接

请在 New relic 中使用全平台用户（不是 Basic/Core）来启用 New Relic MCP 工具。

配置

1. 转到能力提供者页面（可从侧面导航栏访问）

2. 在 “遥测” 下的 “可用提供者” 部分中找到新遗物，然后单击 “注册”
3. 按照说明获取您的 New Relic API 密钥
4. 输入你的 New Relic MCP 服务器 API 密钥详细信息：
   - 账户 ID：输入你在上面获得的 New Relic 账户 ID
   - API 密钥：输入上面获得的 API 密钥
   - 根据您的 New Relic 账户所在地@@ 选择美国或欧盟地区。
5. 单击 “添加”

步骤 2：启用

在特定的代理空间中激活 New Relic 并配置适当的作用域

配置

1. 在座席空间页面上，选择一个座席空间，然后按查看详情（如果您尚未创建座席空间，请参阅the section called “创建代理空间”）
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 按添加
5. 选择新遗物
6. 下一步
7. 查看并按保存
8. 复制 Webhook 网址和 API 密钥

步骤 3：配置 Webhook

使用 Webhook 网址和 API 密钥，您可以将 New Relic 配置为发送事件以触发调查，例如从警报中触发调查。有关设置 webhook 的更多详细信息，请参阅更改跟踪 web hook。

为确保 DevOps 代理可以使用发送的事件，请确保传输到 webhook 的数据与下面指定的数据架构相匹配。 DevOps 代理可以忽略与此架构不匹配的事件。

设置方法和标题

```
    method: "POST",
    headers: {
```

```
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

将正文作为 JSON 字符串发送。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

发送带有 New Relic https://newrelic.com/instant-observability/ 网络挂钩通知的网络挂钩。您可以为授权类型选择 Bearer token，也可以选择不授权，然后将其添加`Authorization: Bearer <Token>`为自定义标题。

了解更多：https://docs.newrelic.com/docs/agentic-/ ai/mcp/overview

移除

遥测源在两个级别上连接，分别是代理空间级别和账户级别。要将其完全删除，必须先将其从所有使用它的代理空间中删除，然后才能将其取消注册。

步骤 1：从代理空间中移除

1. 在座席空间页面上，选择一个座席空间，然后按查看详情
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 选择新遗物
5. 按移除

第 2 步：注销账号

1. 转到能力提供者页面（可从侧面导航栏访问）
2. 滚动到 “当前注册” 部分。
3. 检查代理空间计数是否为零（如果不是，请在其他代理空间中重复上述步骤 1）
4. 在 “新遗物” 旁边按 “取消注册”

连接 Splunk

内置，单向集成

目前， AWS DevOps Agent 通过内置的单向集成支持 Splunk 用户，可实现以下功能：

- 自动触发调查-可以将 Splunk 事件配置为通过代理 webhook 触发 AWS DevOps AWS DevOps 代理事件解决调查。
- 遥测内省- AWS DevOps 代理可以在通过每个提供商的远程 MCP 服务器调查问题时反省 Splunk 遥测数据。

先决条件

获取 Splunk API 代币

你需要一个 MCP 网址和令牌才能连接 Splunk。

Splunk 管理员步骤

您的 Splunk 管理员需要执行以下步骤：

- 启用 REST API 访问权限
- 在部署时@@ 启用令牌身份验证。
- 创建新角色 “mcp_user”，新角色不需要具备任何能力。
- 将角色 “mcp_user” 分配给部署中有权使用 MCP 服务器的所有用户。
- 如果用户无权自己创建令牌，则为受众群体为 “mcp” 的授权用户创建令牌，并设置适当的到期时间。

Splunk 用户步骤

Splunk 用户需要执行以下步骤：

- 从 Splunk 管理员那里获取适当的令牌，或者如果他们有权限，则自己创建一个。代币的受众必须是“mcp”。

信息载入

第 1 步：Connect

使用账户访问凭证与您的 Splunk 远程 MCP 端点建立连接

配置

1. 转到“能力提供者”页面（可从侧面导航栏访问）
2. 在“遥测”下的“可用提供商”部分中找到 Splunk，然后单击“注册”
3. 输入您的 Splunk MCP 服务器详细信息：
   - 服务器名称-唯一标识符（例如 my-splunk-server）
   - 端点 URL-您的 Splunk MCP 服务器端点：

```
https://<YOUR_SPLUNK_DEPLOYMENT_NAME>.api.scs.splunk.com/
<YOUR_SPLUNK_DEPLOYMENT_NAME>/mcp/v1/
```

- 描述-可选的服务器描述
- 令牌名称-用于身份验证的不记名令牌的名称：`my-splunk-token`
- 令牌值用于身份验证的持有者令牌值

步骤 2：启用

在特定的代理空间中激活 Splunk 并配置适当的作用域

配置

1. 在座席空间页面上，选择一个座席空间，然后按查看详情（如果您尚未创建座席空间，请参阅the section called “创建代理空间”）
2. 选择“权能”选项卡

3. 向下滚动到 “遥测” 部分
4. 按添加
5. 选择 Splunk
6. 下一步
7. 查看并按保存
8. 复制 Webhook 网址和 API 密钥

步骤 3：配置 Webhook

使用 Webhook 网址和 API 密钥，您可以将 Splunk 配置为发送事件以触发调查，例如从警报中触发调查。

为确保 DevOps 代理可以使用发送的事件，请确保传输到 webhook 的数据与下面指定的数据架构相匹配。 DevOps 代理可以忽略与此架构不匹配的事件。

设置方法和标题

```
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer <Token>",
    },
```

将正文作为 JSON 字符串发送。

```
{
    eventType: 'incident';
    incidentId: string;
    action: 'created' | 'updated' | 'closed' | 'resolved';
    priority: "CRITICAL" | "HIGH" | "MEDIUM" | "LOW" | "MINIMAL";
    title: string;
    description?: string;
    timestamp?: string;
    service?: string;
    // The original event generated by service is attached here.
    data?: object;
}
```

使用 https://help.splunk.com/en/Splunk splunk-enterprise/alert-and-respond/alerting-manual/9.4/configure-alert-actions/use-发送 webhook a-webhook-alert-action（注意选择不授权，改用自定义标题选项）

了解更多：

- Splunk 的 MCP 服务器文档：https://help.splunk.com/en/splunk-cloud-platform/-platform/ mcp-server-for-splunk-splunk-platform about-mcp-server-for
- Splunk 云平台 REST API 的访问要求和限制：https://docs.splunk.com/Documentation/SplunkCloud/latest/RESTTUT/RESTandCloud
- 在 Splunk 云平台中管理身份验证令牌：https://help.splunk.com/en/splunk-cloud-platform/administer/manage-users-and-security/9.3.2411/authenticate-into-the-splunk-platform-with-tokens/manage-or-delete-authentication-tokens
- 使用 Splunk Web 创建和管理角色：https://docs.splunk.com/Documentation/SplunkCloud/latest/Security/Addandeditroles

移除

遥测源在两个级别上连接，分别是代理空间级别和账户级别。要将其完全删除，必须先将其从所有使用它的代理空间中移除，然后才能将其取消注册。

步骤 1：从代理空间中移除

1. 在座席空间页面上，选择一个座席空间，然后按查看详情
2. 选择 “权能” 选项卡
3. 向下滚动到 “遥测” 部分
4. 选择 Splunk
5. 按移除

第 2 步：注销账号

1. 转到 “能力提供者” 页面（可从侧面导航栏访问）
2. 滚动到 “当前注册” 部分。
3. 检查代理空间计数是否为零（如果不是，请在其他代理空间中重复上述步骤 1）
4. 按 Splunk 旁边的 “取消注册”

连接票务和聊天

AWS DevOps Agent 旨在通过参与团队现有的沟通渠道来充当团队的一员。您可以将 DevOps Agent 连接到您的票务和警报系统，例如 ServiceNow 和 PagerDuty，以自动根据事件单启动调查，从而加快现有工作流程中的事件响应，从而缩短平均恢复时间 (MTTR)。您还可以将 DevOps 代理连接到团队协作系统（例如 Slack），以便在聊天频道中接收来自 DevOps 代理的活动摘要。

要了解如何连接工单和聊天集成，请参阅以下内容：

- the section called “正在连接 PagerDuty”
- the section called “正在连接 ServiceNow”
- the section called “连接 Slack”

正在连接 PagerDuty

PagerDuty 集成使 AWS DevOps 代理能够在事件调查和自动响应期间访问和更新您的 PagerDuty 帐户中的事件数据、待命时间表和服务信息。此集成使用 OAuth 2.0 进行安全身份验证。

> **Important**
>
> AWS DevOps 代理仅支持较新的 PagerDuty OAuth 2.0（Scoped OAuth）。不支持 PagerDuty OAuth 带有重定向 uri 的旧版。

PagerDuty 要求

在连接之前 PagerDuty，请确保您具有：

- 包含您的 OAuth 客户 ID 和客户密钥的 PagerDuty 账户
- 您的 PagerDuty 账户子域名（例如，如果您的 PagerDuty 网址是`https://your-company.pagerduty.com`，则子域名是）`your-company`

正在注册 PagerDuty

PagerDuty 在 AWS 账户级别注册，并在该账户中的所有代理空间之间共享。

步骤 1：在中配置访问权限 PagerDuty

1. 登录到 AWS 管理控制台
2. 导航到 AWS DevOps 代理控制台
3. 转到能力提供者页面（可从侧面导航栏访问）
4. PagerDuty在 “沟通” 下的 “可用提供商” 部分中查找，然后单击 “注册”
5. 按照 “配置访问权限 PagerDuty” 页面上的指导设置进行操作：

检查您的服务区域和子域名：

- 账户范围-选择您所在 PagerDuty 的地区（美国或欧盟），然后输入您的 PagerDuty 子域名。例如，如果您的 PagerDuty URL 是`https://your-company.pagerduty.com`，请输入`your-company`。

在以下位置创建新应用程序 PagerDuty：

- 在单独的浏览器选项卡中，登录 PagerDuty 并导航至集成 > 应用程序注册
- 使用 OAuth 2.0 Sco ped 创建新应用程序 OAuth
- 在 “权限” 下，授予以下最低必需范围：`incidents.read`、`incidents.write`、和 `services.read`
- 启用事件集成以允许 AWS DevOps 代理和之间的双向通信 PagerDuty

配置 OAuth 凭证：

- 权限范围-所需的最低权限范围为：`incidents.read`、`incidents.write`、`services.read`
- 客户名称-为您的 OAuth 客户输入描述性名称
- 客户端 ID — 输入您的 PagerDuty 应用程序注册中的 OAuth 客户端 ID
- 客户密钥-输入您的 PagerDuty 应用程序注册中的 OAuth 客户端密钥

第 2 步：查看并提交 PagerDuty 注册

1. 查看所有 PagerDuty 配置细节
2. 单击 “提交” 完成注册
3. 成功注册后， PagerDuty 将显示在 “功能提供者” 页面的 “当前已注册” 部分

PagerDuty 添加到代理空间

在账户 PagerDuty 级别注册后，您可以将其连接到各个代理空间：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “功能” 选项卡
3. 在 “通信” 部分，单击 “添加”
4. PagerDuty从可用提供商列表中选择
5. 单击 “保存”

管理 PagerDuty 连接

- 更新凭证-如果需要更新您的 OAuth 证书，请 PagerDuty 从 “能力提供者” 页面取消注册，然后使用新的凭据重新注册。
- 查看连接-在 AWS DevOps 代理控制台中，选择您的代理空间，然后转到功能选项卡以查看连接的通信集成。
- 删除 PagerDuty-要断开与代理空间 PagerDuty 的连接，请在 “通信” 部分将其选中，然后单击 “删除”。要完全删除注册，请先将其从所有代理空间中删除，然后从 “能力提供者” 页面取消注册。

Webhook 支持

AWS DevOps 代理仅支持 PagerDuty V3 网络挂钩。不支持早期的 webhook 版本。

有关 PagerDuty V3 webhook 订阅的更多信息，请参阅 PagerDuty 开发者文档中的 Webhooks 概述。

正在连接 ServiceNow

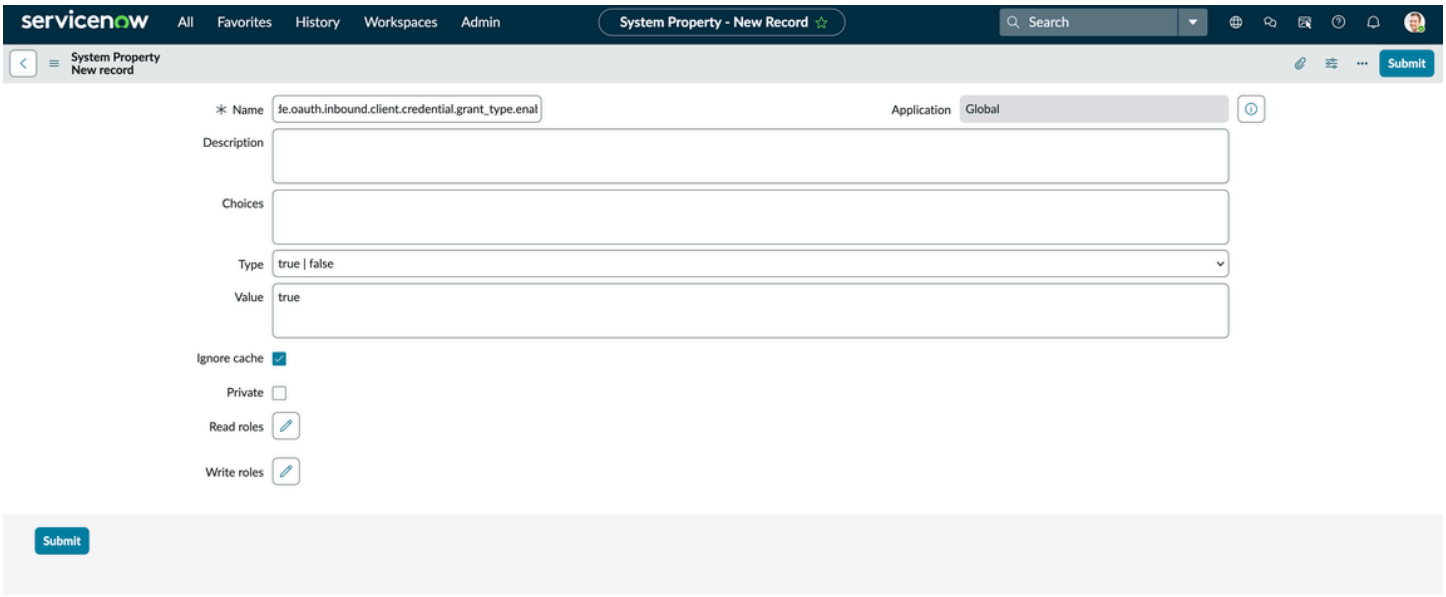
本教程将引导您完成将 ServiceNow 实例连接到 AWS DevOps 代理的过程，使其能够在创建票证时自动启动事件响应调查，并将其关键发现发布到原始票证中。它还包含一些示例，说明如何将您的 ServiceNow 实例配置为仅向 DevOps 代理空间发送特定票证，以及如何协调跨多个 DevOps 代理空间的票证路由。

初始设置

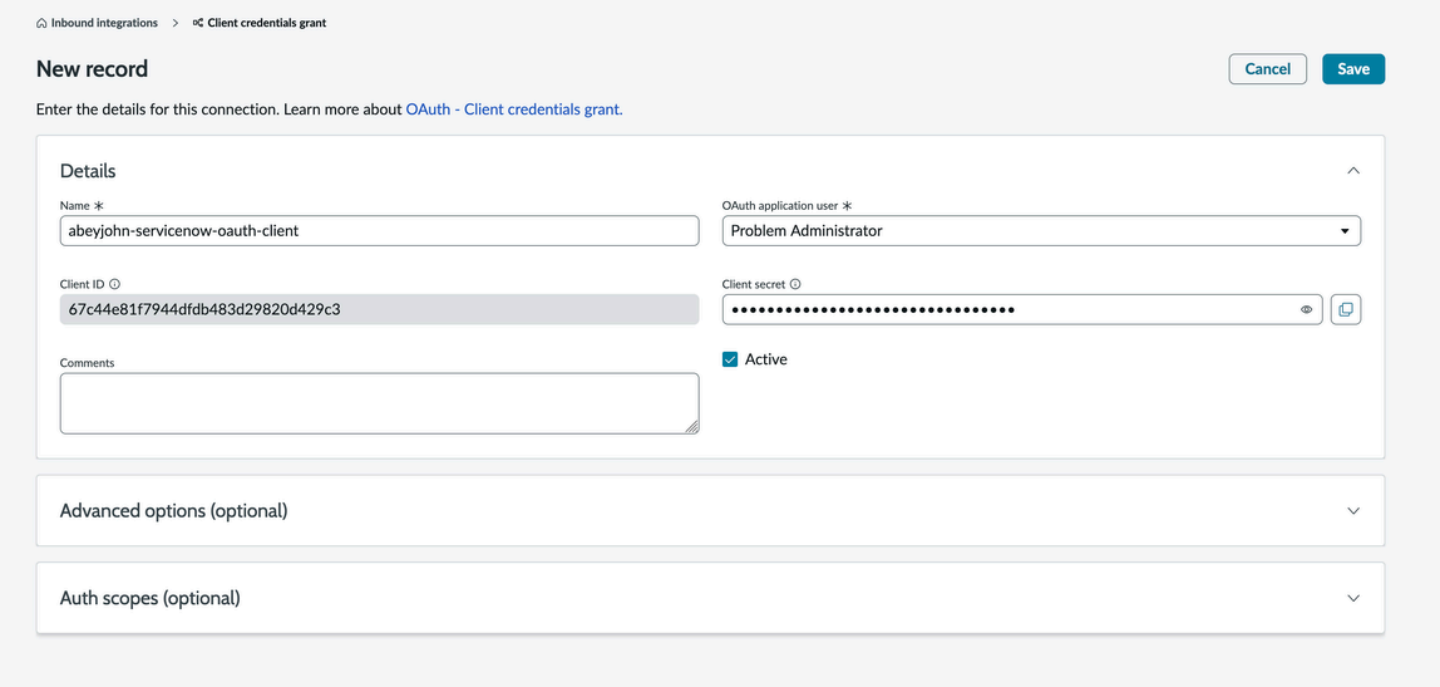
第一步是在 OAuth 应用程序客户端中 ServiceNow 创建 AWS DevOps 可用于访问您的 ServiceNow 实例的客户端。

创建 ServiceNow OAuth 应用程序客户端

1. 启用您的实例的客户端凭证系统属性
   a. `sys_properties.list`在过滤器搜索框中搜索然后按回车键（它不会显示该选项，但按回车起作用）
   b. 选择 “新建”
   c. 将名称添加为，将值添加为 `glide.oauth.inbound.client.credential.grant_type.enabled` true，类型为 true | false



1. 从筛选器搜索框中导航到 “系统” OAuth > “应用程序注册表”
2. 选择 “新建” > “新入站集成体验” > “新集成” > “OAuth -授予客户证书”
3. 选择一个名称并将 OAuth 应用程序用户设置为 “问题管理员”，然后单击 “保存”

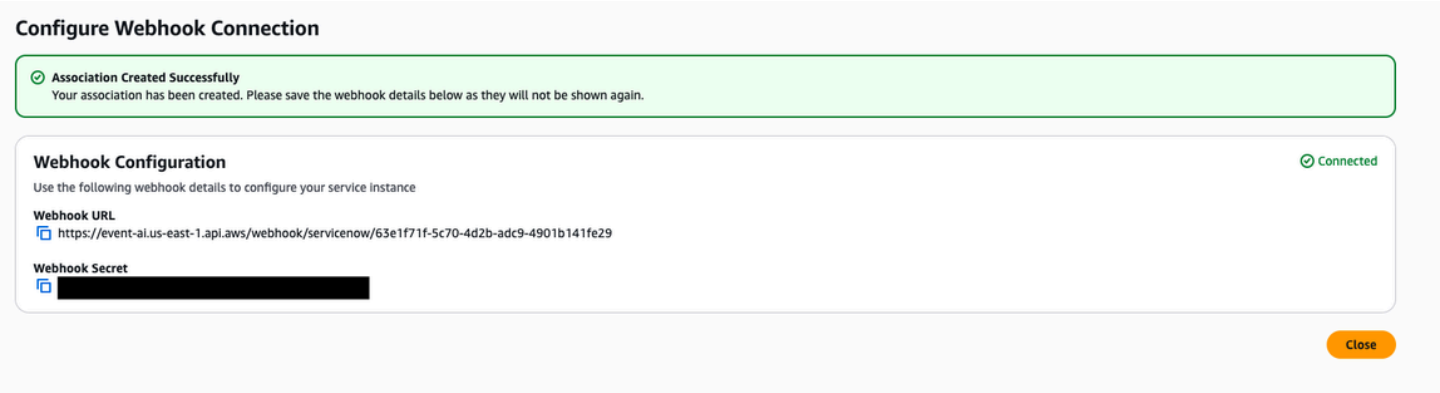


将您的 ServiceNow OAuth 客户机连接到 AWS DevOps 代理

1. 你可以从两个地方开始这个过程。首先，导航到 “能力提供者” 页面并在 “通信” ServiceNow下找到，然后单击 “注册”。或者，您可以选择您可能已创建的任何 DevOps 座席空间，然后导航至权能 → 通信 → 添加 →， ServiceNow 然后单击注册。
2. 接下来，授权 DevOps 代理使用您刚刚创建的 OAuth 应用程序客户端访问您的 ServiceNow 实例。

- 按照以下步骤操作，保存生成的有关 webhook 的信息

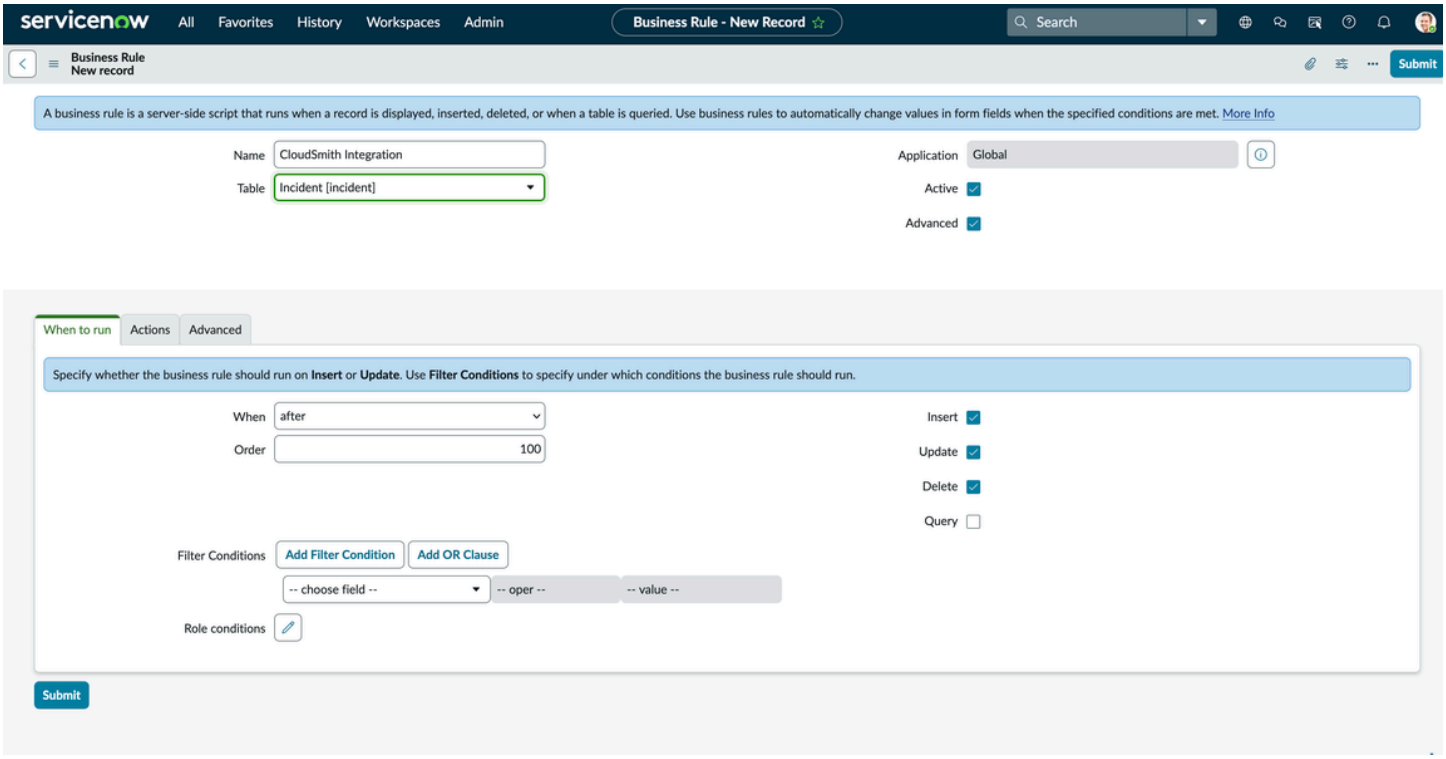
> **Important**
>
> 您将不会再看到此信息



配置您的 ServiceNow 业务规则

建立连接后，您需要在中配置业务规则， ServiceNow 以便将票证发送到您的 DevOps 座席空间。

1. 导航至 “活动订阅” → “管理” → “业务规则”，然后单击 “新建”。
2. 将 “表” 字段设置为 “事件 [事件]”，选中 “高级” 复选框，然后将规则设置为在插入、更新和删除之后运行。



1. 导航到 “高级” 选项卡并添加以下 webhook 脚本，在指示的地方插入您的 webhook 密钥和 URL，然后单击 “提交”。

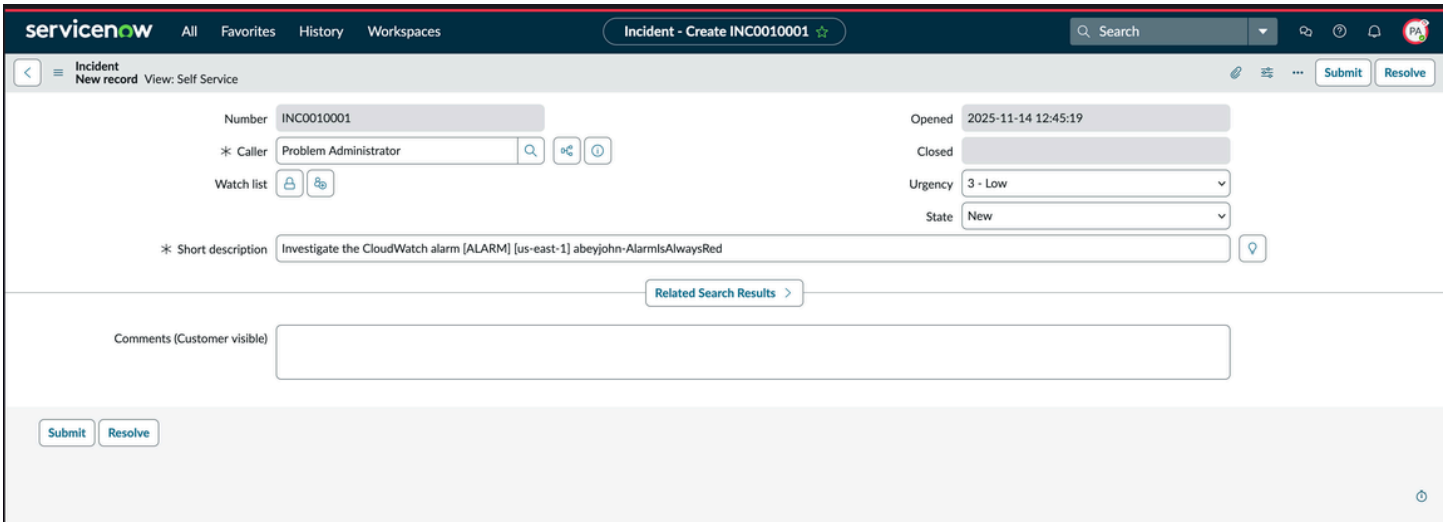
```
(function executeRule(current, previous /*null when async*/ ) {

    var WEBHOOK_CONFIG = {
        webhookSecret: GlideStringUtil.base64Encode('<<< INSERT WEBHOOK SECRET HERE
 >>>'),
        webhookUrl: '<<< INSERT WEBHOOK URL HERE >>>'
    };

    function generateHMACSignature(payloadString, secret) {
        try {
            var mac = new GlideCertificateEncryption();
            var signature = mac.generateMac(secret, "HmacSHA256", payloadString);
            return signature;
        } catch (e) {
            gs.error('HMAC generation failed: ' + e);
            return null;
        }
    }
```

```
    function callWebhook(payload, config) {
        try {
            var timestamp = new Date().toISOString();
            var payloadString = JSON.stringify(payload);
            var payloadWithTimestamp =`${timestamp}:${payloadString}`;

            var signature = generateHMACSignature(payloadWithTimestamp,
config.webhookSecret);

            if (!signature) {
                gs.error('Failed to generate signature');
                return false;
            }

            gs.info('Generated signature: ' + signature);

            var request = new sn_ws.RESTMessageV2();
            request.setEndpoint(config.webhookUrl);
            request.setHttpMethod('POST');

            request.setRequestHeader('Content-Type', 'application/json');
            request.setRequestHeader('x-amzn-event-signature', signature);
            request.setRequestHeader('x-amzn-event-timestamp', timestamp);

            request.setRequestBody(payloadString);

            var response = request.execute();
            var httpStatus = response.getStatusCode();
            var responseBody = response.getBody();

            if (httpStatus >= 200 && httpStatus < 300) {
                gs.info('Webhook sent successfully. Status: ' + httpStatus);
                return true;
            } else {
                gs.error('Webhook failed. Status: ' + httpStatus + ', Response: ' +
responseBody);
                return false;
            }

        } catch (ex) {
            gs.error('Error sending webhook: ' + ex.getMessage());
            return false;
        }
    }
```

```
    function createReference(field) {
        if (!field || field.nil()) {
            return null;
        }

        return {
            link: field.getLink(true),
            value: field.toString()
        };
    }

    function getStringValue(field) {
        if (!field || field.nil()) {
            return null;
        }
        return field.toString();
    }

    function getIntValue(field) {
        if (!field || field.nil()) {
            return null;
        }
        var val = parseInt(field.toString());
        return isNaN(val) ? null : val;
    }

    var eventType = (current.operation() == 'insert') ? "create" : "update";

    var incidentEvent = {
        eventType: eventType.toString(),
        sysId: current.sys_id.toString(),
        priority: getStringValue(current.priority),
        impact: getStringValue(current.impact),
        active: getStringValue(current.active),
        urgency: getStringValue(current.urgency),
        description: getStringValue(current.description),
        shortDescription: getStringValue(current.short_description),
        parent: getStringValue(current.parent),
        incidentState: getStringValue(current.incident_state),
        severity: getStringValue(current.severity),
        problem: createReference(current.problem),
        additionalContext: {}
    };
```

```
    incidentEvent.additionalContext = {
        number: current.number.toString(),
        opened_at: getStringValue(current.opened_at),
        opened_by: current.opened_by.nil() ? null :
 current.opened_by.getDisplayValue(),
        assigned_to: current.assigned_to.nil() ? null :
 current.assigned_to.getDisplayValue(),
        category: getStringValue(current.category),
        subcategory: getStringValue(current.subcategory),
        knowledge: getStringValue(current.knowledge),
        made_sla: getStringValue(current.made_sla),
        major_incident: getStringValue(current.major_incident)
    };

    for (var key in incidentEvent.additionalContext) {
        if (incidentEvent.additionalContext[key] === null) {
            delete incidentEvent.additionalContext[key];
        }
    }

    gs.info(JSON.stringify(incidentEvent, null, 2)); // Pretty print for logging only

    if (WEBHOOK_CONFIG.webhookUrl && WEBHOOK_CONFIG.webhookSecret) {
        callWebhook(incidentEvent, WEBHOOK_CONFIG);
    } else {
        gs.info('Webhook not configured.');
    }

})(current, previous);
```

如果您选择从 “能力提供者” 页面注册 ServiceNow 连接，则现在需要导航到要调查 ServiceNow 事件单的 DevOps 代理空间，选择 “功能” → “通信”，然后在 “能力提供者” 页面上注册的 ServiceNow 实例。现在，一切都应设置完毕，所有将呼叫者设置为 “问题管理员”（模仿您授予 AWS DevOps OAuth 客户端的权限）的事件都将在已配置的 DevOps 代理空间中触发事件响应调查。您可以通过在中创建新事件 ServiceNow 并将事件的 “来电者” 字段设置为 “问题管理员” 来测试这一点。



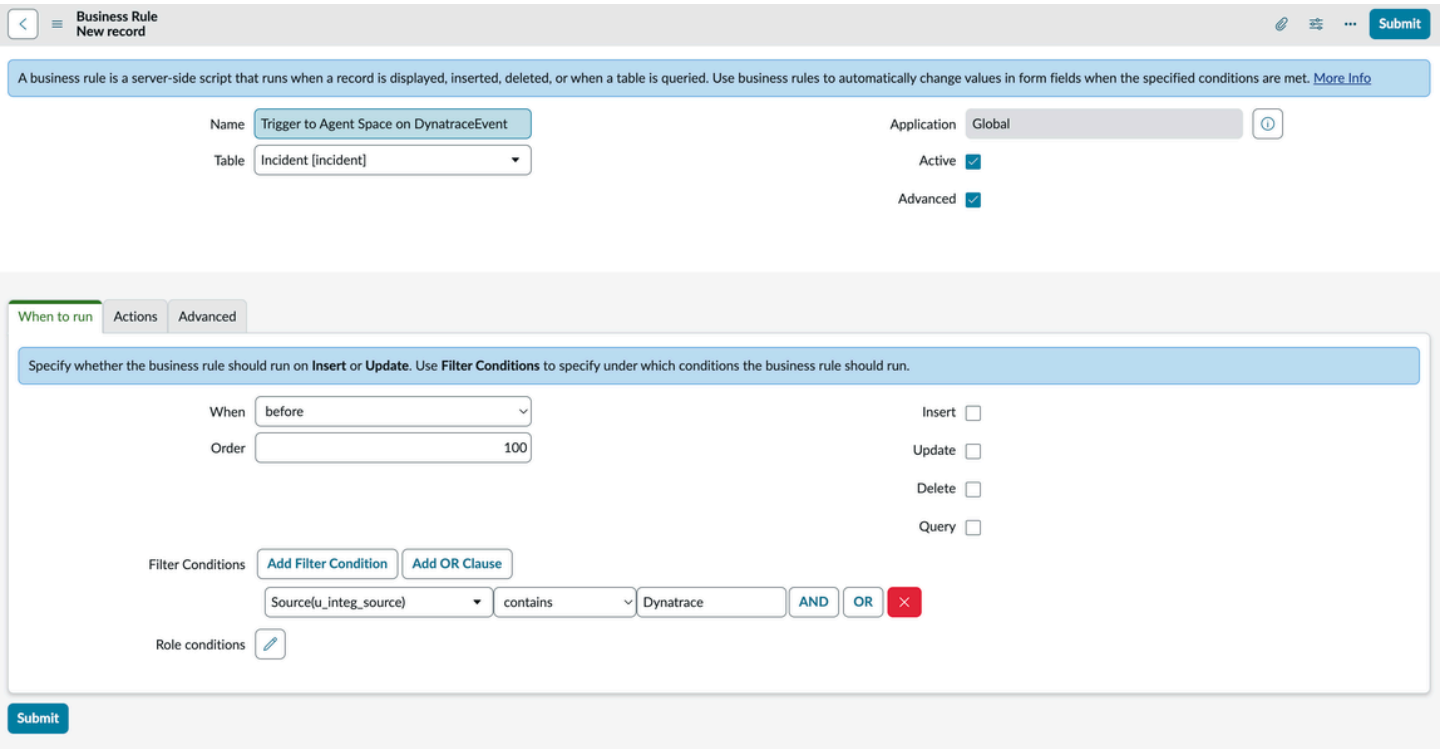
ServiceNow 门票更新

在所有触发的事件响应调查中，您的 DevOps 代理将向原始工单提供其主要发现、根本原因分析和缓解计划的更新。特工的调查结果将发布在事件评论中，我们目前只会发布类型为`finding`、、`causeinvestigation_summary`、`mitigation_summary`、的特工记录和调查状态更新（例如`AWS DevOps Agent started/finished its investigation`）。

工单路由和编排示例

场景：筛选哪些事件被发送到 DevOps 代理空间

这是一个简单的场景，但需要进行一些配置 ServiceNow 才能在中创建 ServiceNow 用于跟踪事件源的字段。在本示例中，使用 SNOW 表单生成器创建一个新的 Source (u_source) 字段。这将允许跟踪事件源，并使用它来将来自特定来源的请求路由到 DevOps 代理空间。路由是通过创建 Service Now Business 规则以及在 “何时运行” 选项卡中设置 “何时” 触发器和 “筛选条件” 来完成的。在此示例中，筛选条件设置如下：

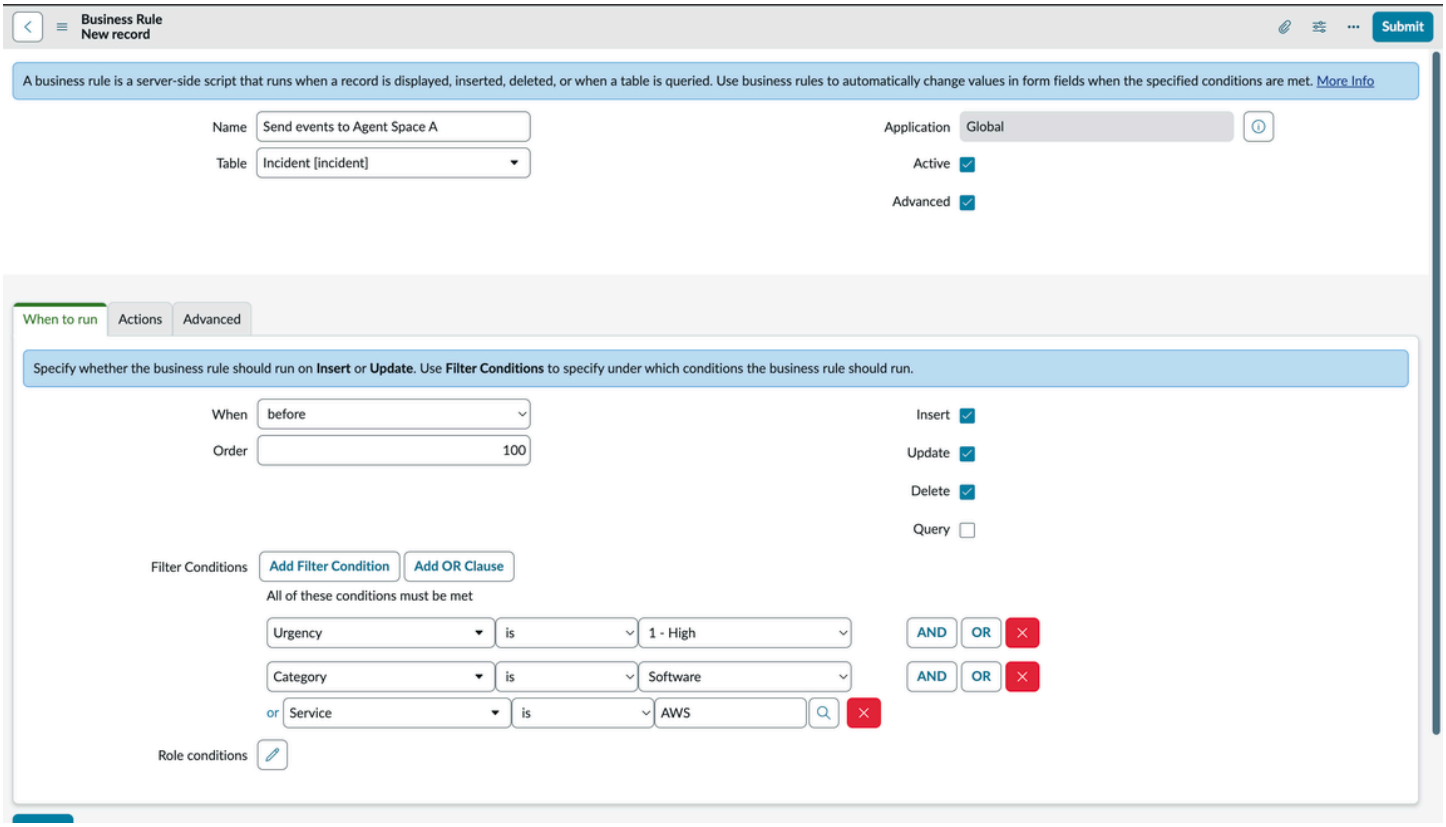


场景：跨多个 DevOps 座席空间路由事件

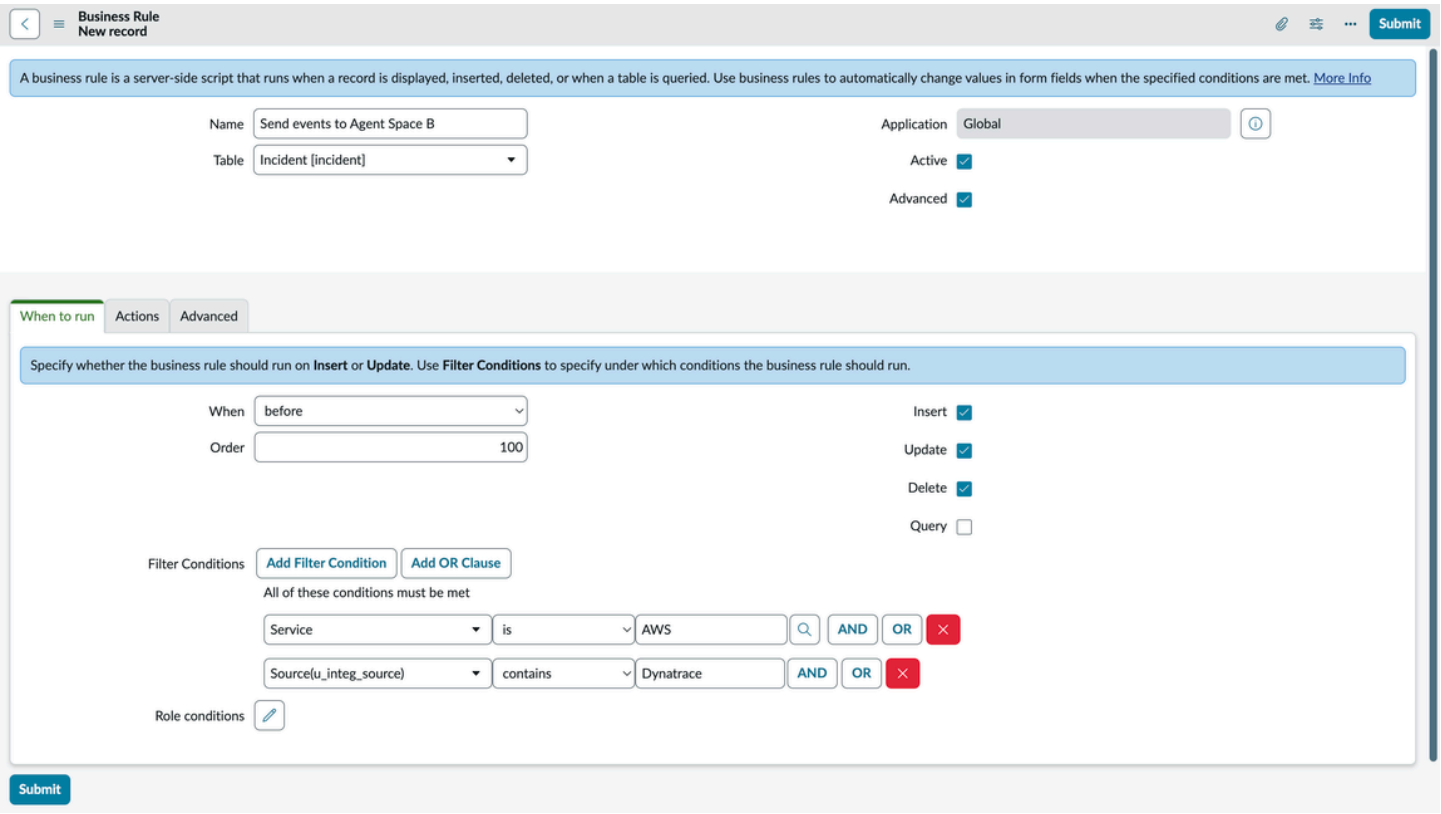
此示例说明当紧急程度为1、类别为或服务为时，如何在 DevOps 代理空间 B 中触发调查，当服务为AWS、来源为时AWS，如何在 DevOps 代理空间 A 中触发调查Dynatrace。Software

这种情况可以通过两种方式实现。可以更新 webhook 脚本本身以包含此业务逻辑。在本场景中，我们将展示如何使用 ServiceNow 业务规则来实现这一目标，以提高透明度并简化调试。路由是通过创建两个 Service Now 业务规则来完成的。

- 在 ServiceNow DevOps Agent Space A 中创建业务规则，然后使用条件生成器创建仅根据我们指定的条件发送事件的条件。



- 接下来，在 AgentSpace B 中 ServiceNow 创建另一条业务规则，只有当服务为 Dynatrace AWS 且来源为 Dynatrace 时，才会触发该业务规则。



现在，当您创建符合指定条件的新事件时，它将触发对 DevOps 代理空间 A 或 DevOps 座席空间 B 的调查，从而为您提供对事件路由的精细控制。

连接 Slack

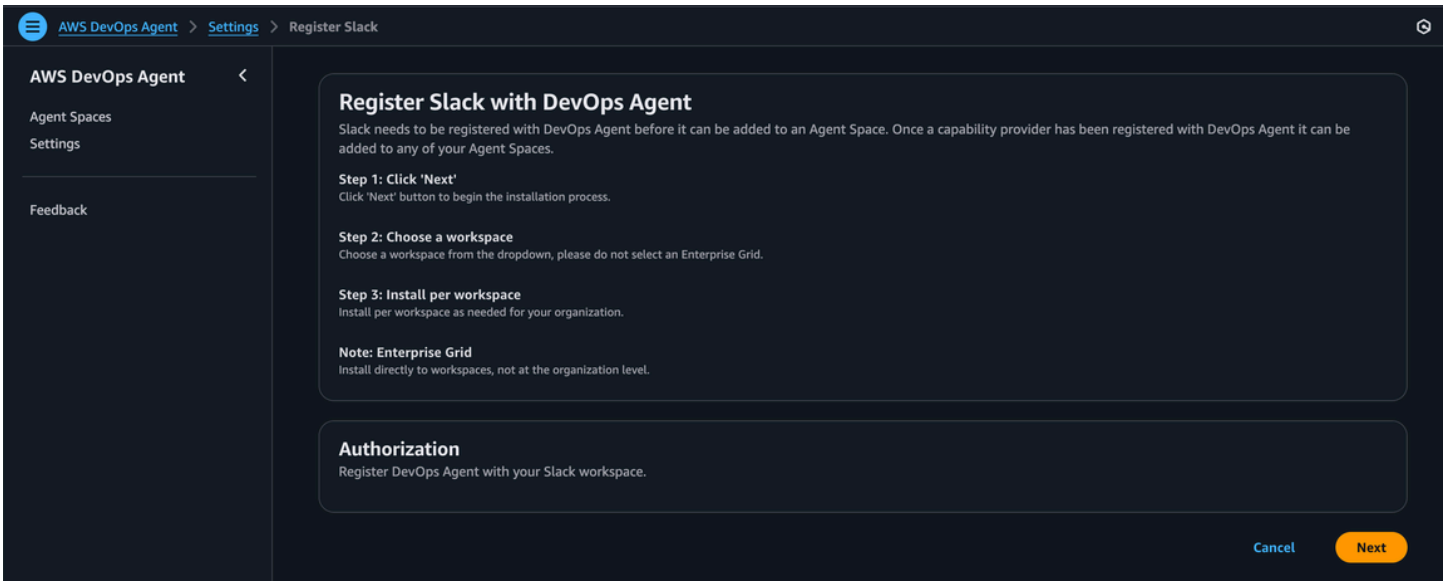
您可以将 AWS DevOps Agent 配置为使用事件响应调查关键发现、根本原因分析和生成的缓解计划来更新您选择的 Slack 频道。

开始前的准备工作

Slack 需要先向 DevOps 代理注册，然后才能将其添加到代理空间。要将 AWS DevOps Agent 与 Slack 集成，您必须满足以下要求：

- 可以访问 Slack 工作区，并能够安装和授权第三方应用程序
- 确定了你想让 AWS DevOps 代理发送通知的 Slack 频道

注册 Slack 与代理的 AWS DevOps 集成



1. 在 AWS DevOps 代理控制台的 “功能提供者” 页面上，在 “通信” 下的 “可用提供者” 部分中找到 Slack，然后单击 “注册”。
2. 选择 “注册” 按钮。
3. 您将被重定向到 Slack，为您的工作空间授权 AWS DevOps 代理应用程序。
4. 在 Slack 授权页面上，直接安装到工作区，而不是在组织级别。
5. 从下拉列表中选择一个工作区。请勿选择企业网格。
6. 根据您的组织需要按工作空间进行安装。
7. 查看请求的范围，然后单击 “允许” 以授权集成。
8. 授权后，您将返回 AWS DevOps 代理控制台。

将 Slack 与你的 DevOps 特工空间关联

在你的 DevOps 特工空间中注册 Slack 后，你可以将其与你的 DevOps 特工空间关联：

1. 在您配置的 “功能” 选项卡中 AgentSpace，导航至 “通信” > “Slack”。
2. 选择 “添加 Slack”
3. 输入频道 ID
4. 选择 “创建” 以完成 Slack 配置。

> **Note**
>
> 代理的机器人用户必须先添加到私人频道，然后才能发布消息。

> **Important**
>
> 卸载 Slack 应用程序可能会导致 Slack 应用程序无法重新安装。请避免卸载 Slack 应用程序。

通过 Webhook 调用 DevOps 代理

Webhook 允许外部系统自动触发 AWS DevOps 代理调查。这样可以与票务系统、监控工具和其他平台集成，这些平台可以在事件发生时发送 HTTP 请求。

先决条件

在配置 webhook 访问权限之前，请确保您具有：

- 在代理中配置的 AWS DevOps 代理空间
- 访问 AWS DevOps 代理控制台
- 将发送 webhook 请求的外部系统

Webhook 类型

AWS DevOps 代理支持以下类型的 Webhook：

- 特定于集成的网络挂钩 — 在配置第三方集成（例如 Dynatrace、Splunk、Datadog、New Relic 或 Slack）时自动生成。 ServiceNow这些 Webhook 与特定的集成相关联，并使用由集成类型确定的身份验证方法
- 通用 webhook — 可以手动创建，用于触发来自特定集成未涵盖的任何来源的调查。通用 webhook 目前使用 HMAC 身份验证（不记名令牌目前不可用）。
- Grafana 警报 webhook — Grafana 可以通过 webhook 联系点直接向代理发送警报通知。 AWS DevOps 有关包括自定义通知模板在内的设置说明，请参阅连接 Grafana。

Webhook 身份验证方法

Webhook 的身份验证方法取决于它与哪个集成关联：

HMAC 身份验证 — 使用者：

- Dynatrace 集成 webhook
- 通用 webhook（未链接到特定的第三方集成）

持有者令牌身份验证 — 使用者：

- Splunk 集成 webhook
- Datadog 集成 webhook
- 全新 Relic 集成 webhook
- ServiceNow 集成 webhook
- Slack 集成 webhook

配置 webhook 访问权限

步骤 1：导航到 webhook 配置

1. 登录 AWS 管理控制台并导航到 AWS DevOps 代理控制台
2. 选择您的代理空间
3. 前往 “功能” 选项卡
4. 在 Webhook 部分中，点击配置

第 2 步：生成 webhook 凭证

对于特定于集成的网络挂钩：

当您完成第三方集成的配置时，系统会自动生成 Webhook。webhook 端点 URL 和凭据是在集成设置过程结束时提供的。

对于通用 Webhook：

1. 点击生成 webhook

2. 系统将生成 HMAC key pair
3. 安全地存储生成的密钥和机密——您将无法再次检索它们
4. 复制提供的 webhook 端点 URL

步骤 3：配置您的外部系统

使用 webhook 端点 URL 和凭据将您的外部系统配置为向 AWS DevOps 代理发送请求。具体的配置步骤取决于您的外部系统。

管理 webhook 凭证

删除凭据 -要删除 webhook 凭据，请转到 webhook 配置部分，然后单击删除。移除凭据后，Webhook 端点将不再接受请求，直到您生成新的凭据。

重新生成凭证-要生成新证书，请先删除现有证书，然后生成新的 key pair 或令牌。

使用 webhook

Webhook 请求格式

要触发调查，您的外部系统应向 webhook 端点 URL 发送 HTTP POST 请求。

对于版本 1（HMAC 身份验证）：

标头：

- `Content-Type: application/json`
- `x-amzn-event-signature: <HMAC signature>`
- `x-amzn-event-timestamp: <+%Y-%m-%dT%H:%M:%S.000Z>`

HMAC 签名是通过使用 SHA-256 使用您的密钥对请求正文进行签名来生成的。

对于版本 2（持有者令牌身份验证）：

标头：

- `Content-Type: application/json`
- `Authorization: Bearer <your-token>`

请求正文：

请求正文应包含有关事件的信息：

```
json

{
  "title": "Incident title",
  "severity": "high",
  "affectedResources": ["resource-id-1", "resource-id-2"],
  "timestamp": "2025-11-23T18:00:00Z",
  "description": "Detailed incident description",
  "data": {
    "metadata": {
        "region": "us-east-1",
        "environment": "production"
    }
  }
}
```

代码示例

版本 1（HMAC 身份验证）- JavaScript:

```
const crypto = require('crypto');

// Webhook configuration
const webhookUrl = 'https://your-webhook-endpoint.amazonaws.com/invoke';
const webhookSecret = 'your-webhook-secret-key';

// Incident data
const incidentData = {
    eventType: 'incident',
    incidentId: 'incident-123',
    action: 'created',
    priority: "HIGH",
    title: 'High CPU usage on production server',
    description: 'High CPU usage on production server host ABC in AWS account 1234
 region us-east-1',
    timestamp: new Date().toISOString(),
    service: 'MyTestService',
    data: {
      metadata: {
```

```
        region: 'us-east-1',
        environment: 'production'
      }
    }
};

// Convert data to JSON string
const payload = JSON.stringify(incidentData);
const timestamp = new Date().toISOString();
const hmac = crypto.createHmac("sha256", webhookSecret);
hmac.update(`${timestamp}:${payload}`, "utf8");
const signature = hmac.digest("base64");

// Send the request
fetch(webhookUrl, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'x-amzn-event-timestamp': timestamp,
    'x-amzn-event-signature': signature
  },
  body: payload
})
.then(res => {
  console.log(`Status Code: ${res.status}`);
  return res.text();
})
.then(data => {
  console.log('Response:', data);
})
.catch(error => {
  console.error('Error:', error);
});
```

版本 1（HMAC 身份验证）-curl：

```
#!/bin/bash

# Configuration
WEBHOOK_URL="https://event-ai.us-east-1.api.aws/webhook/generic/YOUR_WEBHOOK_ID"
SECRET="YOUR_WEBHOOK_SECRET"

# Create payload
```

```
TIMESTAMP=$(date -u +%Y-%m-%dT%H:%M:%S.000Z)
INCIDENT_ID="test-alert-$(date +%s)"

PAYLOAD=$(cat <<EOF
{
"eventType": "incident",
"incidentId": "$INCIDENT_ID",
"action": "created",
"priority": "HIGH",
"title": "Test Alert",
"description": "Test alert description",
"service": "TestService",
"timestamp": "$TIMESTAMP"
}
EOF
)

# Generate HMAC signature
SIGNATURE=$(echo -n "${TIMESTAMP}:${PAYLOAD}" | openssl dgst -sha256 -hmac "$SECRET" -
binary | base64)

# Send webhook
curl -X POST "$WEBHOOK_URL" \
-H "Content-Type: application/json" \
-H "x-amzn-event-timestamp: $TIMESTAMP" \
-H "x-amzn-event-signature: $SIGNATURE" \
-d "$PAYLOAD"
```

版本 2（持有者令牌身份验证）- JavaScript:

```
function sendEventToWebhook(webhookUrl, secret) {
  const timestamp = new Date().toISOString();

  const payload = {
    eventType: 'incident',
    incidentId: 'incident-123',
    action: 'created',
    priority: "HIGH",
    title: 'Test Alert',
    description: 'Test description',
    timestamp: timestamp,
    service: 'TestService',
    data: {}
```

```
  };

  fetch(webhookUrl, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "x-amzn-event-timestamp": timestamp,
      "Authorization": `Bearer ${secret}`,  // Fixed: template literal
    },
    body: JSON.stringify(payload),
  });
}
```

版本 2（持有者令牌身份验证）-cURL：

```
#!/bin/bash

# Configuration
WEBHOOK_URL="https://event-ai.us-east-1.api.aws/webhook/generic/YOUR_WEBHOOK_ID"
SECRET="YOUR_WEBHOOK_SECRET"

# Create payload
TIMESTAMP=$(date -u +%Y-%m-%dT%H:%M:%S.000Z)
INCIDENT_ID="test-alert-$(date +%s)"

PAYLOAD=$(cat <<EOF
{
"eventType": "incident",
"incidentId": "$INCIDENT_ID",
"action": "created",
"priority": "HIGH",
"title": "Test Alert",
"description": "Test alert description",
"service": "TestService",
"timestamp": "$TIMESTAMP"
}
EOF
)

# Send webhook
curl -X POST "$WEBHOOK_URL" \
-H "Content-Type: application/json" \
-H "x-amzn-event-timestamp: $TIMESTAMP" \
```

```
-H "Authorization: Bearer $SECRET" \
-d "$PAYLOAD"
```

网络挂钩疑难解答

如果你没有收到 200

200 和收到的类似于 webhook 的消息表示身份验证已通过，消息已排队等待系统验证和处理。如果你得到的不是200，而是4xx，则很可能是身份验证或标头有问题。尝试使用 curl 选项手动发送以帮助调试身份验证。

如果您收到 200 但调查尚未开始

可能的原因是有效载荷格式不正确。

1. 检查时间戳和事件 ID 是否已更新且唯一。重复的消息会被删除重复。
2. 检查消息是否有效 JSON
3. 检查格式是否正确

如果您收到 200，但调查立即取消

你很可能已经达到了当月的上限。如果合适，请与您的 AWS 联系人联系，要求更改速率限制。

相关主题

- the section called “创建代理空间”
- the section called “什么是 DevOps 代理 Web 应用程序？”
- the section called “DevOps 代理 IAM 权限”

将 AWS DevOps Agent 与亚马逊集成 EventBridge

您可以使用调查和缓解生命周期中发生的事件，将 AWS DevOps Agent 与事件驱动的应用程序集成。AWS DevOps 当调查或缓解措施的状态发生变化 EventBridge 时，代理会向 Amazon 发送事件。然后，您可以创建 EventBridge 规则，根据这些事件采取行动。

例如，您可以创建执行以下操作的规则：

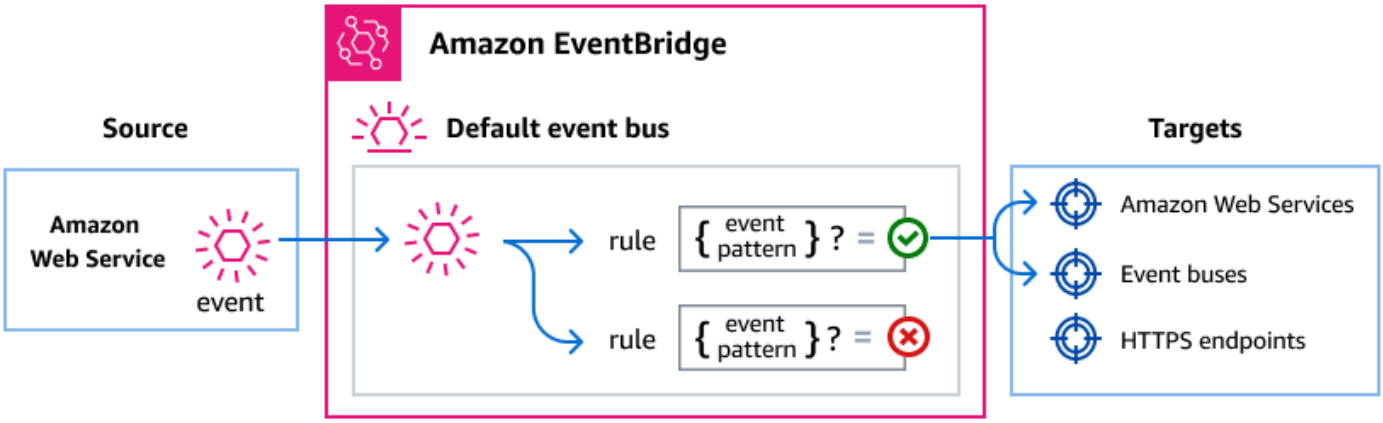
- 调查完成后，调用 AWS Lambda 函数来处理调查结果。

- 当调查失败或超时时，发送 Amazon SNS 通知。
- 创建新调查时更新工单系统。
- 缓解操作完成后，启动 AWS Step Functions 工作流程。

如何 EventBridge 路由 AWS DevOps 代理事件

AWS DevOps 代理将事件发送到 EventBridge 默认事件总线。 EventBridge 然后根据您创建的规则评估事件。当事件与规则的事件模式匹配时， EventBridge 会将该事件发送到指定的目标。

下图显示了如何 EventBridge 路由 AWS DevOps 代理事件。



1. AWS DevOps 当调查或缓解生命周期状态发生变化时，代理会向 EventBridge 默认事件总线发送一个事件。
2. EventBridge 根据您创建的规则评估事件。
3. 如果事件与规则的事件模式匹配，则 EventBridge 将该事件发送到规则中指定的目标。

AWS DevOps 代理事件

AWS DevOps 代理向发送以下事件 EventBridge。所有事件都使用源`aws.aidevops`。

支持的调查事件

detail-type	说明
`Investigation Created`	在特工领域进行了调查。

detail-type	说明
`Investigation Priority Updated`	调查的优先顺序已改变。
`Investigation In Progress`	一项调查开始了积极分析。
`Investigation Completed`	调查成功结束，得出了调查结果。
`Investigation Failed`	调查遇到错误，无法完成。
`Investigation Timed Out`	调查超过了允许的最大持续时间。
`Investigation Cancelled`	一项调查在完成之前被取消。
`Investigation Pending Triage`	在主动分析开始之前，调查正在等待分类。
`Investigation Linked`	调查与相关事件或罚单有关。

支持的缓解事件

detail-type	说明
`Mitigation In Progress`	缓解措施已启动。
`Mitigation Completed`	缓解操作成功完成。
`Mitigation Failed`	缓解操作遇到错误，无法完成。
`Mitigation Timed Out`	缓解措施超过了允许的最大持续时间。
`Mitigation Cancelled`	缓解措施在完成之前被取消。

有关详细的字段描述和示例事件，请参阅the section called “AWS DevOps 代理事件详细信息参考”。

创建与 AWS DevOps 代理事件匹配的事件模式

EventBridge 规则使用事件模式来选择事件并将其路由到目标。事件模式与其处理的事件结构相匹配。您可以创建事件模式以根据事件字段筛选 AWS DevOps 代理事件。

以下示例显示了常见用例的事件模式。

匹配所有 AWS DevOps Agent 事件

以下事件模式匹配来自 AWS DevOps Agent 的所有事件。

```
{
  "source": ["aws.aidevops"]
}
```

仅匹配调查事件

以下事件模式使用前缀匹配来仅选择调查生命周期事件。

```
{
  "source": ["aws.aidevops"],
  "detail-type": [{"prefix": "Investigation"}]
}
```

仅匹配完成和失败事件

以下事件模式与已完成或失败的调查和缓解措施的事件相匹配。

```
{
  "source": ["aws.aidevops"],
  "detail-type": [
    "Investigation Completed",
    "Investigation Failed",
    "Mitigation Completed",
    "Mitigation Failed"
  ]
}
```

匹配特定代理空间的事件

以下事件模式匹配来自特定代理空间的事件。

```
{
  "source": ["aws.aidevops"],
  "detail": {
    "metadata": {
      "agent_space_id": ["your-agent-space-id"]
    }
```

```
  }
}
```

有关事件模式的更多信息，请参阅《亚马逊 EventBridge 用户指南》中的亚马逊 EventBridge 事件模式。

亚马逊 EventBridge 权限

AWS DevOps 代理不需要额外的权限即可将事件传送到 EventBridge。这些事件会自动发送到默认事件总线。

根据您为 EventBridge 规则配置的目标，您可能需要添加特定的权限。有关目标所需权限的更多信息，请参阅《亚马逊 EventBridge 用户指南》 EventBridge中的使用亚马逊基于资源的策略。

其他 EventBridge 资源

有关 EventBridge 概念和配置的更多信息，请参阅 Amazon EventBridge 用户指南中的以下主题：

- EventBridge 活动巴士
- EventBridge 事件
- EventBridge 事件模式
- EventBridge 规则
- EventBridge 目标

AWS DevOps 代理事件详细信息参考

来自 AWS 服务的事件具有通用的元数据字段`source`，包括`detail-type`、`account`、`region`、和`time`。这些事件还包含一个包含特定于该服务的数据的`detail`字段。对于 AWS DevOps 代理事件，始终`source`为`aws.aidevops`，标`detail-type`识特定事件。

调查事件

以下`detail-type`值用于标识调查事件：

- `Investigation Created`
- `Investigation Priority Updated`
- `Investigation In Progress`
- `Investigation Completed`

- `Investigation Failed`
- `Investigation Timed Out`
- `Investigation Cancelled`
- `Investigation Pending Triage`
- `Investigation Linked`

下面包含`source`和`detail-type`字段，因为它们包含 AWS DevOps 代理事件的特定值。有关所有事件中包含的其他元数据字段的定义，请参阅 Amazon Events 参考中的 EventBridge 事件结构。

以下是调查事件的 JSON 结构。

```
{
  . . .,
  "detail-type" : "string",
  "source" : "aws.aidevops",
  . . .,
  "detail" : {
    "version" : "string",
    "metadata" : {
      "agent_space_id" : "string",
      "task_id" : "string",
      "execution_id" : "string"
    },
    "data" : {
      "task_type" : "string",
      "priority" : "string",
      "status" : "string",
      "created_at" : "string",
      "updated_at" : "string",
      "summary_record_id" : "string"
    }
  }
}
```

`detail-type`标识事件的类型。对于调查事件，这是前面列出的事件名称之一。

`source`标识生成事件的服务。对于 AWS DevOps 代理事件，此值为`aws.aidevops`。

`detail`包含事件特定数据的 JSON 对象。该`detail`对象包括以下字段：

- `version`（字符串）-事件详细信息的架构版本。目前`1.0.0`。

- metadata.agent_space_id（字符串）-事件起源的代理空间的唯一标识符。
- metadata.task_id（字符串）-任务的唯一标识符。
- metadata.execution_id（字符串）-执行运行的唯一标识符。当调查被指派执行死刑时在场。
- data.task_type（字符串）-任务的类型。值：INVESTIGATION。
- data.priority（字符串）-优先级。值：CRITICAL、HIGH、MEDIUM、LOW、MINIMAL。
- data.status（字符串）-当前状态。
  值：PENDING_START、IN_PROGRESS、COMPLETED、FAILED、TIMED_OUT、CANCELLED、PENDING_
- data.created_at（字符串）— 创建任务时的 ISO 8601 时间戳。
- data.updated_at（字符串）— 上次更新任务时的 ISO 8601 时间戳。
- data.summary_record_id（字符串）-包含调查结果的摘要记录的标识符。在为已完成的调查生成摘要时包括在内。您可以通过 AWS DevOps 代理 API 检索摘要内容，方法是使用此标识符来查找记录类型为的日记记录investigation_summary_md。

示例：调查已完成事件

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789015",
  "detail-type": "Investigation Completed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:10:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "b2c3d4e5-6789-01ab-cdef-example22222"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "COMPLETED",
```

```
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:10:00Z",
      "summary_record_id": "d4e5f6g7-6789-01ab-cdef-example44444"
    }
  }
}
```

示例：调查失败事件

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789016",
  "detail-type": "Investigation Failed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:10:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "b2c3d4e5-6789-01ab-cdef-example22222"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "FAILED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:10:00Z"
    }
  }
}
```

缓解事件

以下`detail-type`值用于标识缓解事件：

- `Mitigation In Progress`

- `Mitigation Completed`
- `Mitigation Failed`
- `Mitigation Timed Out`
- `Mitigation Cancelled`

下面包含`source`和`detail-type`字段，因为它们包含 AWS DevOps 代理事件的特定值。有关所有事件中包含的其他元数据字段的定义，请参阅 Amazon Events 参考中的 EventBridge 事件结构。

以下是缓解事件的 JSON 结构。

```
{
  . . .,
  "detail-type" : "string",
  "source" : "aws.aidevops",
  . . .,
  "detail" : {
    "version" : "string",
    "metadata" : {
      "agent_space_id" : "string",
      "task_id" : "string",
      "execution_id" : "string"
    },
    "data" : {
      "task_type" : "string",
      "priority" : "string",
      "status" : "string",
      "created_at" : "string",
      "updated_at" : "string",
      "summary_record_id" : "string"
    }
  }
}
```

`detail-type`标识事件的类型。对于缓解事件，这是前面列出的事件名称之一。

`source`标识生成事件的服务。对于 AWS DevOps 代理事件，此值为`aws.aidevops`。

`detail`包含事件特定数据的 JSON 对象。该`detail`对象包括以下字段：

- `version`（字符串）-事件详细信息的架构版本。目前`1.0.0`。

- metadata.agent_space_id（字符串）-事件起源的代理空间的唯一标识符。
- metadata.task_id（字符串）-任务的唯一标识符。
- metadata.execution_id（字符串）-执行运行的唯一标识符。在为缓解措施分配执行时出现。
- data.task_type（字符串）-任务的类型。值：INVESTIGATION。
- data.priority（字符串）-优先级。值：CRITICAL、HIGH、MEDIUM、LOW、MINIMAL。
- data.status（字符串）-当前状态。
  值：IN_PROGRESS、COMPLETED、FAILED、TIMED_OUT、CANCELLED。
- data.created_at（字符串）— 创建任务时的 ISO 8601 时间戳。
- data.updated_at（字符串）— 上次更新任务时的 ISO 8601 时间戳。
- data.summary_record_id（字符串）-包含缓解结果的摘要记录的标识符。在为已完成的缓解措施生成摘要时包括在内。您可以通过 AWS DevOps 代理 API 检索摘要内容，方法是使用此标识符来查找记录类型为的日记记录mitigation_summary_md。

示例：缓解已完成事件

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-12345678901c",
  "detail-type": "Mitigation Completed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:20:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "c3d4e5f6-7890-12ab-cdef-example33333"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "COMPLETED",
      "created_at": "2026-03-12T18:00:00Z",
```

```
      "updated_at": "2026-03-12T18:20:00Z",
      "summary_record_id": "e5f6g7h8-7890-12ab-cdef-example55555"
    }
  }
}
```

示例：缓解失败事件

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-12345678901d",
  "detail-type": "Mitigation Failed",
  "source": "aws.aidevops",
  "account": "123456789012",
  "time": "2026-03-12T18:20:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:aidevops:us-
east-1:123456789012:agentspace/8f6187a7-0388-4926-8217-3a0fe32f757c"
  ],
  "detail": {
    "version": "1.0.0",
    "metadata": {
      "agent_space_id": "8f6187a7-0388-4926-8217-3a0fe32f757c",
      "task_id": "a1b2c3d4-5678-90ab-cdef-example11111",
      "execution_id": "c3d4e5f6-7890-12ab-cdef-example33333"
    },
    "data": {
      "task_type": "INVESTIGATION",
      "priority": "CRITICAL",
      "status": "FAILED",
      "created_at": "2026-03-12T18:00:00Z",
      "updated_at": "2026-03-12T18:20:00Z"
    }
  }
}
```

出售的日志和指标

您可以使用附带的 Amazon CloudWatch 指标和日志来监控您的代理空间和服务操作。本主题介绍 AWS DevOps 代理自动发布到您的账户的 CloudWatch 指标，以及您可以配置的发送到首选目的地的销售日志。

已售指标 CloudWatch

AWS DevOps 代理会自动将指标发布到您的账户 CloudWatch 中的 Amazon。这些指标无需任何配置即可使用。您可以使用它们来监控使用情况、跟踪操作活动和创建警报。

服务相关角色

要在您的账户中发布该服务的亚马逊 CloudWatch 指标， AWS DevOps 代理将自动为您创建与服务相关的角色 AWSService RoleForAIDevOps 服务相关角色。如果调用 API 的 IAM 角色没有适当的权限，则资源创建将失败，并显示为 InvalidParameterException。

> **Important**
>
> 在 2026 年 3 月 13 日 AgentSpace 之前创建的客户需要手动创建 AWSServiceRoleForAIDevOps Service 关联角色，才能在其账户中发布 AWS DevOps 代理 CloudWatch 指标。

手动创建服务相关角色（适用于现有客户）

请执行以下操作之一：

- 在 IAM 控制台中，在AWS DevOps 代理服务下创建 AWSServiceRoleForAIDevOps 角色。
- 从 C AWS LI 中运行以下命令：

```
aws iam create-service-linked-role --aws-service-name aidevops.amazonaws.com
```

命名空间

所有指标都发布在`AWS/AIDevOps`命名空间下。

Dimensions

所有指标都包含以下维度。

维度	说明
AgentSpaceUUID	代理空间的唯一标识符。要汇总账户中所有代理空间的指标，请使用 CloudWatch 数学表达式或省略维度筛选器。

指标参考

指标名称	说明	单位	发布频率	有用的统计数据
ConsumedChatRequests	座席空间消耗的聊天请求数。要获取您账户的总数，请使用所有AgentSpaceUUID 维度的SUM统计数据。	计数	每 5 分钟	总和、平均值
ConsumedInvestigationTime	在代理空间中进行调查所花费的时间。	秒	每 5 分钟	总和、平均值、最大值
ConsumedEvaluationTime	在代理空间中运行评估所花费的时间。	秒	每 5 分钟	总和、平均值、最大值
TopologyCompletionCount	拓扑处理完成次数。 AWS DevOps 无论是从入职期间的初始创建、手动更新还是计划的每日刷新开始，当拓扑完成处理	计数	事件驱动（每次完成时发出）	总和，SampleCount

指标名称	说明	单位	发布频率	有用的统计数据
	时，代理都会发出此指标。			

在 CloudWatch 控制台中查看指标

1. 打开 CloudWatch 控制台。
2. 在导航窗格中，选择 Metrics（指标），然后选择 All metrics（所有指标）。
3. 选择 AWS/O AIDev ps 命名空间。
4. 选择 “依 AgentSpace据” 以查看您的代理空间的指标。

> **Note**
>
> 您可以根据这些指标创建 CloudWatch 警报，以便在使用量超过阈值时收到通知。例如，创建警报`ConsumedChatRequests`以监控聊天请求的使用情况。

先决条件

在配置日志传送之前，请确保具备以下条件：

- 有权访问 AWS DevOps 代理控制台的活跃 AWS 账户
- 具有 CloudWatch 日志传输权限的 IAM 委托人 APIs
- （可选）Amazon S3 存储桶或 Amazon Data Firehose 传输流（如果您计划将其用作日志目标）

Vended logs（已出售日志）

AWS DevOps 代理支持销售日志，这些日志可让您了解您的代理空间和服务注册处理的事件。Vended Logs 使用 Amazon CloudWatch Logs 基础设施将日志传送到您的首选目的地。

要使用已售日志，必须配置传送目的地。支持以下目的地：

- Amazon CloudWatch 日志-您账户中的日志组
- 亚马逊 S3 — 您账户中的 S3 存储桶
- Amazon Data Firehose — 你账户中的 Firehose 传送流

支持的日志类型

支持单一日志类型：APPLICATION_LOGS。此日志类型涵盖服务发出的所有操作事件。

记录事件类型

下表汇总了 AWS DevOps 代理记录的事件。

事件	说明	日志级别
已收到代理入站事件	代理由集成源触发并接收入站事件（例如， PagerDuty 事件事件）。	INFO
代理入站事件已丢弃	入站事件在代理处理之前已将其丢弃。日志中包含原因（例如，格式错误的数据）。	待定
代理出站通信失败	与第三方集成的出站通信失败。日志包括任务 ID 和目标标识符（例如，身份验证错误）。	待定
拓扑创建已排队	拓扑创建作业已排队等候处理。	INFO
拓扑创建已开始	拓扑创建作业已开始处理。	INFO
拓扑创建已完成	拓扑创建任务已完成处理。此事件适用于初始创建、更新和每日刷新。	INFO
资源发现失败	拓扑创建期间的资源发现遇到故障。	ERROR
服务注册失败	服务注册遇到无法恢复的故障	ERROR
Webhook 验证失败	当 Devops 代理收到的 webhook 与预期架构不匹配时	ERROR

事件	说明	日志级别
关联验证状态更新	当代理空间关联（典型 primary/secondary 账户）时，验证状态会从有效变为无效，反之亦然（例如，由于角色格式错误所致，服务无法假设这一点）。	错误/信息

Permissions

AWS DevOps 代理使用CloudWatch 已售日志（V2 权限）来传送日志。要设置日志传输，配置传输的 IAM 角色必须具有以下权限：

- `aidevops:AllowVendedLogDeliveryForResource`— 需要允许代理空间资源的日志传输。
- CloudWatch 日志传送权限 APIs（`logs:PutDeliverySource`、`logs:PutDeliveryDestinationlogs:CreateDelivery`、和相关操作）。
- 特定于您选择的配送目的地的权限。

有关每种目标类型所需的完整 IAM 政策，请参阅 A mazon CloudWatch Logs 用户指南中的以下主题：

- 发送到日志的 CloudWatch 日志
- 发送到 Amazon S3 的日志
- 已发送到 Firehose 的日志

配置日志传输（控制台）

AWS DevOps 代理在 AWS 管理控制台中提供了两个位置来配置日志传输：

- 服务注册设置页面-为服务级别事件配置日志传输。这些日志使用服务 ARN (`arn:aws:aidevops:<region>:<account-id>:service/<account-id>`) 作为资源。
- Agent Space 页面-为特定于单个代理空间的事件配置日志传输。这些日志使用代理空间 ARN (`arn:aws:aidevops:<region>:<account-id>:agentspace/<agent-space-id>`) 作为资源。

为服务注册配置日志传输

1. 在 AWS 管理控制台中打开 AWS DevOps 代理控制台。
2. 在导航窗格中，选择设置。
3. 在 “功能提供者” > “日志” 选项卡中，选择配置。
4. 对于目的地类型，请选择以下选项之一：
5. CloudWatch 日志-选择或创建日志组。
6. 亚马逊 S3 — 输入 S3 存储桶 ARN。
7. Amazon Data Firehose — 选择或创建 Firehose 传送流。
8. 对于其他设置-可选，您可以指定以下选项：
   a. 对于字段选择，请选择要传输到目标的日志字段名称。您可以选择访问日志字段以及实时访问日志字段的子集。
   b. （仅限 Amazon S3）对于分区，请指定日志文件数据分区的路径。
   c. （仅限 Amazon S3）对于与 Hive 兼容的文件格式，您可以选中复选框，使用与 Hive 兼容的 S3 路径。这有助于简化将新数据加载到与 Hive 兼容的工具中的过程。
   d. 对于输出格式，指定您偏好的格式。
   e. 在字段分隔符中，指定如何分隔日志字段。
9. 选择保存。
10确认配送状态显示为 “激活”。

为代理空间配置日志传输

1. 在 AWS 管理控制台中打开 AWS DevOps 代理控制台。
2. 选择要配置的代理空间。
3. 在 “配置” 选项卡中，选择 “配置”。
4. 对于目的地类型，请选择以下选项之一：
5. CloudWatch 日志-选择或创建日志组。
6. 亚马逊 S3 — 输入 S3 存储桶 ARN。
7. Amazon Data Firehose — 选择或创建 Firehose 传送流。
8. 对于其他设置-*可选 *，您可以指定以下选项：
   a. 对于字段选择，请选择要传输到目标的日志字段名称。您可以选择访问日志字段以及实时访问日志字段的子集。

b. （仅限 Amazon S3）对于分区，请指定日志文件数据分区的路径。

c. （仅限 Amazon S3）对于与 Hive 兼容的文件格式，您可以选中复选框，使用与 Hive 兼容的 S3 路径。这有助于简化将新数据加载到与 Hive 兼容的工具中的过程。

d. 对于输出格式，指定您偏好的格式。

e. 在字段分隔符中，指定如何分隔日志字段。

9. 选择保存。

10确认配送状态显示为 “激活”。

配置日志传输 (CloudWatch API)

您还可以使用 CloudWatch 日志 API 以编程方式配置日志传输。工作日志传输由三个元素组成：

- A DeliverySource— 表示生成日志的 AWS DevOps 代理空间资源。
- A DeliveryDestination-表示写入日志的目的地。
- 交付-将传送源连接到传送目的地。

步骤 1：创建交付来源

使用PutDeliverySource操作创建交付来源。传递 AWS DevOps 代理空间资源的 ARN 并指定`APPLICATION_LOGS`为日志类型。

以下示例为代理空间创建交付源：

```
{
    "name": "my-agent-space-delivery-source",
    "resourceArn": "arn:aws:aidevops:us-east-1:123456789012:agentspace/my-agent-space-
id",
    "logType": "APPLICATION_LOGS"
}
```

以下示例为服务创建交付源：

```
{
    "name": "my-service-delivery-source",
    "resourceArn": "arn:aws:aidevops:us-east-1:123456789012:service",
    "logType": "APPLICATION_LOGS"
}
```

步骤 2：创建配送目的地

使用PutDeliveryDestination操作来配置日志的存储位置。你可以选择 Amazon CloudWatch Logs、Amazon S3 或 Amazon Data Firehose。

以下示例创建了一个 CloudWatch 日志目标：

```
{
    "name": "my-cwl-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/
aidevops/my-agent-space"
    },
    "outputFormat": "json"
}
```

以下示例创建了一个 Amazon S3 目的地：

```
{
    "name": "my-s3-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:s3:::my-aidevops-logs-bucket"
    },
    "outputFormat": "json"
}
```

以下示例创建了一个 Amazon Data Firehose 目标：

```
{
    "name": "my-firehose-destination",
    "deliveryDestinationConfiguration": {
        "destinationResourceArn": "arn:aws:firehose:us-
east-1:123456789012:deliverystream/my-aidevops-log-stream"
    },
    "outputFormat": "json"
}
```

> **Note**
>
> 如果您跨账户传送日志，则必须在目标账户PutDeliveryDestinationPolicy中使用来授权传送。

如果你想使用 CloudFormation，你可以使用以下内容：

- Delivery
- DeliveryDestination
- DeliverySource

ResourceArn 是 AgentSpaceArn，LogType 必须是作为支持的日志类型的 APPLICATION_LOGS。

步骤 3：创建配送

使用CreateDelivery操作将传送源链接到传送目的地。

```
{
    "deliverySourceName": "my-agent-space-delivery-source",
    "deliveryDestinationArn": "arn:aws:logs:us-east-1:123456789012:delivery-
destination:my-cwl-destination"
}
```

AWS CloudFormation

您还可以使用 AWS CloudFormation 以下资源来配置日志传输：

- AWS:: 日志:: DeliverySource
- AWS:: 日志:: DeliveryDestination
- AWS:: 日志:: 交付

设置ResourceArn为 AWS DevOps 代理代理空间或服务 ARN，并设置为LogType。APPLICATION_LOGS

日志架构参考

AWS DevOps 代理在所有事件类型中使用共享日志架构。并非每个日志事件都会使用每个字段。

下表描述了日志架构中的字段。

字段	Type	说明
event_timestamp	长整型	事件发生时的 Unix 时间戳

字段	Type	说明
resource_arn	字符串	生成事件的资源的 ARN
可选账号	字符串	AWS 与日志关联的账户 ID。
可选级别	字符串	日志级 别：INFO、WARN、ERROR
可选代理空间标识	字符串	代理空间的标识符。
可选关联_ID	字符串	日志的关联标识符。
可选状态	字符串	拓扑操作的状态。
可选_webhook_id	字符串	Webhook 标识符。
optional_mcp_endpoint_url	字符串	MCP 服务器端点网址
可选服务类型	字符串	服务类型：DYNATRACE 、、DATADOGGITHUB、SLACK、SERV W 。
可选服务端点网址	字符串	第三方集成的终端节点 URL。
可选服务 ID	字符串	来源的标识符。
request_id	字符串	用于关联 AWS CloudTrail 或支持票证的请求标识符。
可选操作	字符串	已执行的操作的名称。
可选任务类型	字符串	代理待办事项任务类 型：INVESTIGATION 或 EVALUATION
可选任务_ID	字符串	代理待办事项任务 IDAgent 积压任务标识符。
可选引用	字符串	来自代理任务（例如 Jira 工单）的参考。

字段	Type	说明
可选错误类型	字符串	错误类型
可选错误消息	字符串	操作失败时的错误描述。
可选详情	字符串 (JSON)	包含操作参数和结果的服务特定事件负载。

管理和禁用日志传输

您可以随时通过 AWS 管理控制台中的 AWS DevOps 代理控制台或使用 CloudWatch 日志 API 修改或删除日志传输。

管理日志传输（控制台）

1. 在 AWS 管理控制台中打开 AWS DevOps 代理控制台。
2. 导航到 “设置” 页面（用于服务级别日志）或特定的 “代理空间” 页面（用于代理空间级别的日志）。
3. 在 “配置” 选项卡（用于代理空间级日志）或 “功能提供商” > “日志” 选项卡（用于服务级别日志）中，选择要修改的传输。
4. 根据需要更新配置，然后选择保存。

> 注意：您无法更改现有配送的目的地类型。要更改目的地类型，请删除当前的配送并创建一个新的配送。

禁用日志传输（控制台）

1. 在 AWS 管理控制台中打开 AWS DevOps 代理控制台。
2. 导航到 “设置” 页面（用于服务级别日志）或特定的 “代理空间” 页面（用于代理空间级别的日志）。
3. 在 “配置” 选项卡（用于代理空间级日志）或 “功能提供商” > “日志” 选项卡（用于服务级别日志）中，选择要删除的传输。
4. 选择删除并确认。

禁用日志传输 (API)

要使用 API 删除日志传输，请按以下顺序删除资源：

1. 使用删除配送DeleteDelivery。
2. 使用删除传送来源DeleteDeliverySource。
3. （可选）如果不再需要配送目的地，请使用将其删除DeleteDeliveryDestination。

> **Important**
>
> 在删除生成日志的代理空间资源之后（例如，在删除代理空间之后），您负责删除日志传输资源。如果您不删除这些资源，则可能会保留孤立的交付配置。

定价

AWS DevOps 代理不为启用已售日志而收费。但是，根据您选择的日志传输目标，可能会产生传输、摄取、存储或访问费用。有关定价详情，请参阅 Amazon CloudWatch P ricing 中 “日志” 选项卡上的 “销售日志”。

有关特定目的地的定价，请参阅以下内容：

- 亚马逊 CloudWatch 日志定价
- Amazon S3 定价
- Amazon Data Firehose 定价

连接到私人托管的工具

私有连接概述

AWS DevOps 可以使用自定义模型上下文协议 (MCP) 工具和其他集成来扩展代理，这些工具允许代理访问内部系统，例如私有包注册表、自托管可观测性平台、内部文档 APIs和源代码管理实例（参见:）。为 AWS DevOps 代理配置功能这些服务通常在 Amazon Virtual Private Cloud（Amazon VPC）内运行，公共互联网访问受限或没有公共访问权限，这意味着默认情况下， AWS DevOps 代理无法访问它们。

通过 AWS DevOps 代理的私有连接，您可以将代理空间安全地连接到 VPC 中运行的服务，而无需将其暴露在公共互联网上。私有连接适用于需要访问私有端点的任何集成，包括 MCP 服务器、自托管 Grafana 或 Splunk 实例，以及源代码控制系统，例如企业服务器和自我管理。 GitHub GitLab

> **Note**
>
> 如果您的私有托管工具从您的 VPC 内部向 AWS DevOps 代理发出出站请求，则也可以使用 VPC 终端节点保护这些流量，使其保持在 AWS 网络中。例如，这可以与通过 webhook 事件触发 DevOps 代理的工具一起使用（请参阅:the section called “通过 Webhook 调用 DevOps 代理”）。有关更多信息，请参阅 the section called “VPC 终端节点 (AWS PrivateLink)”。

私有连接的工作原理

私有连接可在 AWS DevOps 代理和您的 VPC 中的目标资源之间创建安全的网络路径。在幕后， AWS DevOps 代理使用 Amazon VPC Lattice 来建立这条安全的私有连接路径。VPC Lattice 是一项应用网络服务，可让您连接、保护和监控跨 VPCs账户和计算类型的应用程序之间的通信，而无需管理底层网络基础设施。

创建私有连接时，会发生以下情况：

- 您提供与目标服务具有网络连接的 VPC、子网和（可选）安全组。
- AWS DevOps 代理创建服务管理的资源网关，并在您指定的子网中配置其弹性网络接口 (ENIs)。
- 代理使用资源网关通过私有网络路径将流量路由到目标服务的 IP 地址或 DNS 名称。

资源网关完全由 AWS DevOps 代理管理，并在您的账户（名为`aidevops-{your-private-connection-name}`）中显示为只读资源。您无需对其进行配置或维护。在您的 VPC 中创建的唯一资源 ENIs 位于您指定的子网中。 ENIs 它们是私人流量的入口点，完全由该服务管理。他们不接受来自互联网的入站连接，您可以通过自己的安全组完全控制他们的流量。

安全性

私有连接的设计具有多层安全性：

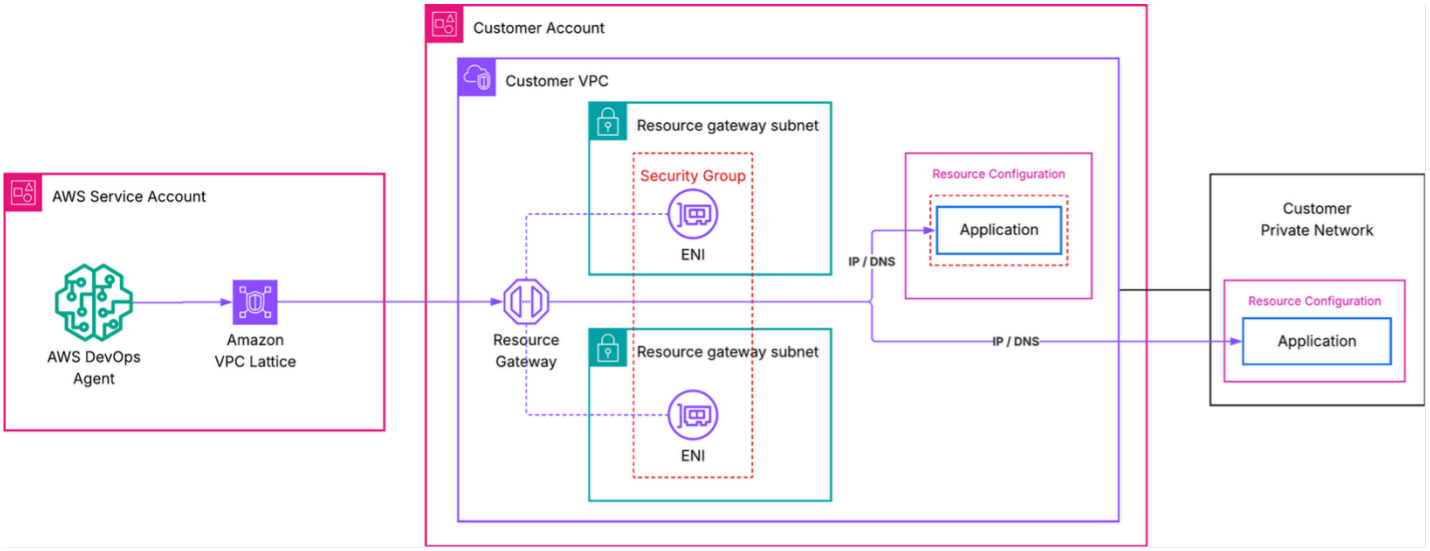
- 没有公共互联网暴露 — AWS DevOps 代理和您的目标服务之间的所有流量都留在 AWS 网络上。您的服务永远不需要公有 IP 地址或互联网网关。
- 服务控制的资源网关-服务管理的资源网关在您的账户中是只读的。它只能由 AWS DevOps 代理使用，其他服务或委托人无法通过它路由流量。您可以在记录所有VPC Lattice API调用的AWS CloudTrail日志中对此进行验证。
- 您的安全组，您的规则-您可以控制流向您拥有和管理的 ENIs 直通安全组的入站和出站流量。如果您未指定安全组， AWS DevOps 代理会创建一个默认安全组，其范围仅限于您定义的端口。

- 权限最低的服务相关角色 — AWS DevOps 代理使用服务相关角色仅创建必要的 VPC Lattice 和 Amazon EC2 资源。此角色仅限于标记为的资源，AWSAIDevOpsManaged并且无法访问您账户中的任何其他资源。

> **Note**
>
> 如果您的组织有限制VPC Lattice API操作的服务控制策略 (SCPs)，则服务管理的资源网关是通过服务相关角色创建的。确保您 SCPs 允许服务相关角色执行必要的操作。

架构

下图显示了专用连接的网络路径。



在此架构中：

- AWS DevOps 代理向您的目标服务发起请求。
- Amazon VPC Lattice 通过您的 VPC 中的服务托管资源网关路由请求。有关使用您自己的 VPC 莱迪思资源的高级设置，请参阅使用现有 VPC 莱迪思资源的高级设置。
- 您的 VPC 中的 ENI 接收流量并将其转发到目标服务的 IP 地址或 DNS 名称。
- 您的安全组控制允许哪些流量通过 ENIs。
- 从目标服务的角度来看，请求来自您的 VPC ENIs 内的私有 IP 地址。

创建私有连接

您可以使用 AWS 管理控制台或 AWS CLI 创建私有连接。

> **Note**
>
> VPC Lattice 不支持以下可用区：`use1-az3`、`usw1-az2`、`apne1-az3`、`apne2-az2`、、`euc1-az2`、`euw1-az4cac1-az3`、`ilc1-az2`。

先决条件

在创建私有连接之前，请确认您已具备以下条件：

- 活跃的代理空间-您的账户中需要一个现有的代理空间。如果没有，请参阅开始使用 AWS DevOps 代理。
- 可私密访问的目标服务 — 您的 MCP 服务器、可观察性平台或其他服务必须能够通过已知的私有 IP 地址或 DNS 名称从部署资源网关的 VPC 访问。只要该服务可以从资源网关子网进行路由，就可以在同一 VPC、对等 VPC 或本地运行。该服务必须在您在创建连接时指定的端口上提供 TLS 最低版本为 1.2 的 HTTPS 流量。
- 您的 VPC 中的子网 — 确定要在其中创建 1-20 个子网。 ENIs 我们建议在多个可用区中选择子网以实现高可用性。这些子网必须与您的目标服务建立网络连接。VPC Lattice 每个可用区只能使用一个子网。
- （可选）安全组-如果您想使用特定规则控制流量，请准备最多五个安全组 IDs 以附加到 ENIs。如果省略安全组，则 AWS DevOps 代理会创建默认安全组。

私有连接是账户级别的资源。创建私有连接后，可以在需要访问同一主机的多个集成和代理空间中重复使用该连接。

使用控制台创建私有连接

1. 打开 AWS DevOps 代理控制台。
2. 在导航窗格中，选择能力提供商，然后选择专用连接。
3. 选择建立新的连接。
4. 在名称中，输入连接的描述性名称，例如`my-mcp-tool-connection`。
5. 对于 VP C，选择 ENIs 将部署资源网关的 VPC。
6. 对于子网，请选择一个或多个子网（最多 20 个）。我们建议在至少两个可用区中选择子网。

7. 在 IP 地址类型中，选择目标服务的 IP 地址类型（`IPv4`、`IPv6`或`DualStack`）。
8. （可选）对于 IPv4 地址数量，如果您选择了 IP 地址类型 IPv4 或 Dualstack，则可以输入资源网关每 IPv4 个 ENI 的地址数。默认为每 IPv4 个 ENI 有 16 个地址。
9. （可选）对于安全组，请选择现有安全组（最多 5 个），以限制允许哪些流量到达您的目标服务。如果未选择任何安全组，则会创建默认安全组。
10. （可选）对于端口范围，请指定目标应用程序监听的 TCP 端口（例如`443`或`8080-8090`）。您最多可以指定 11 个端口范围。
11. 在主机地址中，输入目标服务的 IP 地址或 DNS 名称（例如，`mcp.internal.example.com`或`10.0.1.50`）。必须可以从选定的 VPC 访问该服务。如果您选择 DNS 名称，则该名称必须可以从选定的 VPC 中解析。
12. （可选）对于证书公钥，如果您指定的主机地址使用私有证书颁发机构颁发的 TLS 证书，请输入证书的 PEM 编码公钥。这允许 AWS DevOps 代理信任与目标服务的 TLS 连接。
13. 选择创建连接。

连接状态更改为“正在创建”。此过程最多可能需要 10 分钟。当状态更改为“活动”时，网络路径已准备就绪。

如果状态更改为“创建失败”，请验证以下内容：

- 您指定的子网具有可用的 IP 地址。
- 您的账户未达到 VPC 莱迪思服务配额。
- 没有任何限制性的 IAM 策略阻止服务相关角色创建资源。

> **Note**
>
> 这些步骤也可以通过在注册功能提供者`Create a new private connection`期间进行选择来执行。有关更多信息，请参阅使用与能力提供者的私有连接。

使用 AWS CLI 创建私有连接

运行以下命令创建私有连接。用自己的占位符值替换占位符值。

```
aws devops-agent create-private-connection \
    --name my-mcp-tool-connection \
    --mode '{
```

```
        "serviceManaged": {
            "hostAddress": "mcp.internal.example.com",
            "vpcId": "vpc-0123456789abcdef0",
            "subnetIds": [
                "subnet-0123456789abcdef0",
                "subnet-0123456789abcdef1"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
            ],
            "portRanges": ["443"]
        }
    }'
```

响应包括连接名称和状态CREATE_IN_PROGRESS：

```
{
    "name": "my-mcp-tool-connection",
    "status": "CREATE_IN_PROGRESS",
    "resourceGatewayId": "rgw-0123456789abcdef0",
    "hostAddress": "mcp.internal.example.com",
    "vpcId": "vpc-0123456789abcdef0"
}
```

要检查连接状态，请使用以下describe-private-connection命令：

```
aws devops-agent describe-private-connection \
    --name my-mcp-tool-connection
```

当状态为时ACTIVE，您的私人连接已准备就绪，可以使用。

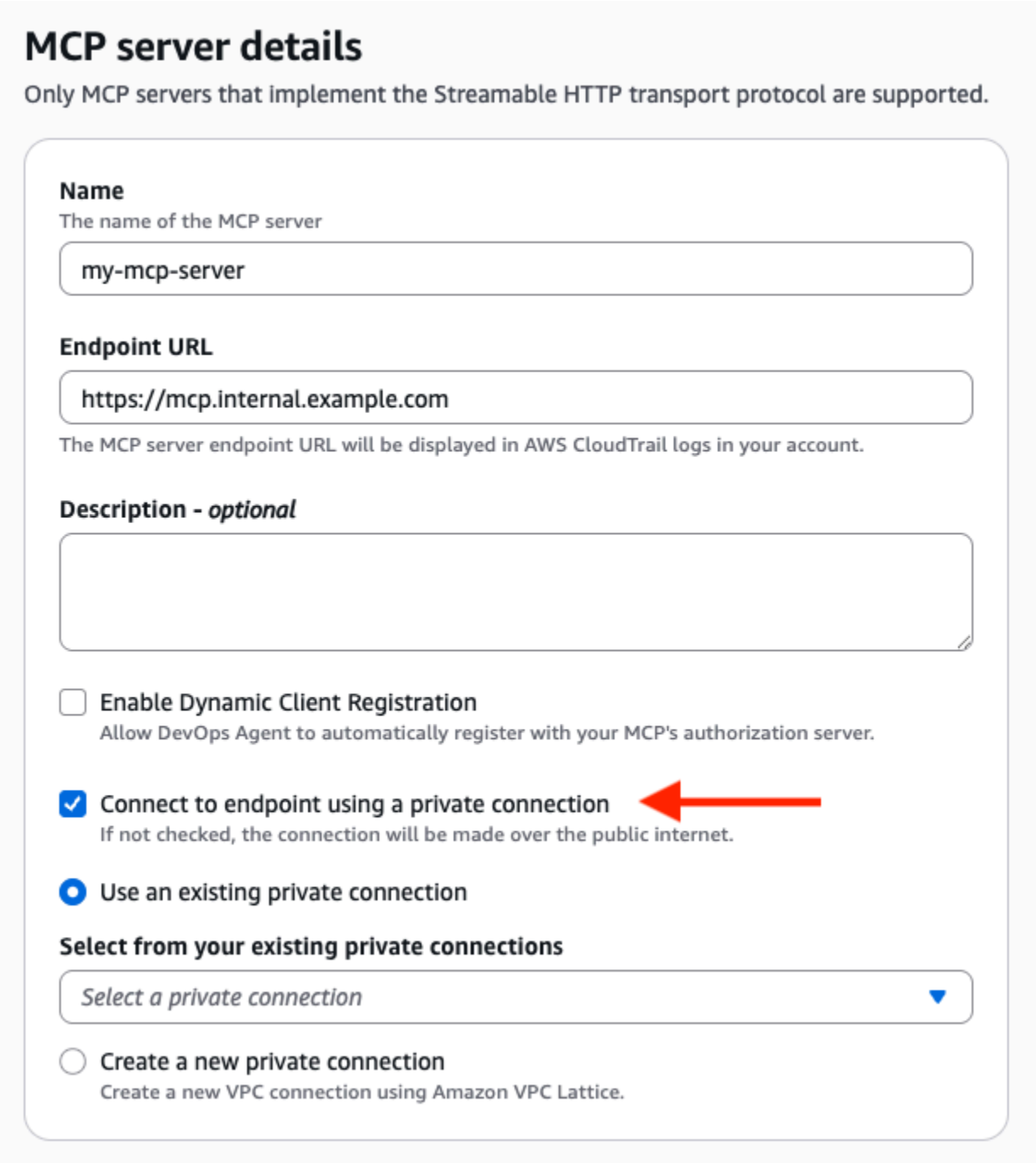
使用与功能提供商的私有连接

要使用私有连接，可以在注册功能提供商的过程中链接到该私有连接。可用于私有连接的支持功能包括：GitHub、GitLabMCP Server、和Grafana。您可以使用 AWS 管理控制台或 AWS CLI 执行此步骤。

> **Note**
>
> 注册功能提供程序时， AWS DevOps 代理会验证端点是否可访问并已响应。在完成注册之前，请确保您的目标服务正在运行并接受连接。

使用控制台与能力提供者建立私有连接

在 AWS DevOps 代理控制台中，通过选择 “使用专用连接连接到端点” 选项，可以在注册期间将私有连接链接到功能。



1. 打开 AWS DevOps 代理控制台并导航到您的代理空间。
2. 在 “能力提供者” 部分中，选择注册。
3. 为要用于私有连接的功能类型选择 “注册”。

4. 在注册详细信息视图中，输入要使用私有连接连接的终端节点 URL（例如`https://mcp.internal.example.com`）。
5. 选择 “使用私有连接连接到端点”。
6. 要么选择与您要连接的终端节点 URL 相对应的现有私有连接，要么选择创建新的私有连接来创建私有连接。
7. 完成能力提供者的注册流程。

使用 C AWS LI 与功能提供者建立私有连接

您可以通过包含`private-connection-name`参数向私有连接注册权能。以下是使用`my-mcp-tool-connection`私有连接注册具有 API 密钥授权的 MCP 服务器的示例。用自己的占位符值替换占位符值。

```
aws devops-agent register-service \
    --service mcpserver \
    --private-connection-name my-mcp-tool-connection \
    --service-details '{
        "mcpserver": {
            "name": "my-mcp-tool",
            "endpoint": "https://mcp.internal.example.com",
            "authorizationConfig": {
                "apiKey": {
                    "apiKeyName": "api-key",
                    "apiKeyValue": "secret-value",
                    "apiKeyHeader": "x-api-key"
                }
            }
        }
    }' \
    --region us-east-1
```

验证私有连接

在专用连接达到 “活动” 状态并已被功能提供商使用后，请验证 AWS DevOps 代理是否可以访问您的目标服务：

1. 打开 AWS DevOps 代理控制台并导航到您的代理空间。
2. 开始新的聊天会话。

3. 调用使用由您的私有连接支持的集成的命令。例如，如果您的 MCP 工具提供对内部知识库的访问权限，请向代理询问一个需要该知识库的问题。
4. 确认代理返回来自私有服务的结果。

如果连接失败，请检查以下内容：

- VPC Lattice 限制-确认您尚未达到任何资源网关或其他 VPC 莱迪思配额限制
- 安全组规则-验证附加到的安全组是否 ENIs 允许您的服务监听的端口上的出站流量。此外，请验证您的服务的安全组是否允许目标端口上的入站流量。流量从您的 VPC CIDR IPs 范围内的 VPC 莱迪思数据平面到达。您可以使用安全组引用（允许 ENI 安全组作为来源），也可以允许从 VPC CIDR 入站。
- 子网连接-验证您选择的子网是否可以将流量路由到您的服务。如果服务在不同的子网中运行，请确认路由表允许它们之间的流量。
- 服务可用性-确认您的服务正在运行并接受预期端口上的连接。
- 不支持的可用区-验证您的子网是否位于支持的可用区中。运行`aws ec2 describe-subnets --subnet-ids <your-subnet-ids> --query 'Subnets[*].[SubnetId,AvailabilityZoneId]'`并检查上面列出的不支持的可用区。

删除私有连接

您可以使用 AWS 管理控制台或 AWS CLI 删除未使用的私有连接。

使用控制台删除私有连接

1. 打开 AWS DevOps 代理控制台。
2. 在导航窗格中，选择能力提供商，然后选择专用连接。
3. 选择要删除的专用连接的 “操作” 菜单，然后选择 “删除”。

当 AWS DevOps 代理 ENIs 从您的 VPC 中移除托管资源网关时，私有连接的状态将显示为 “正在删除连接”。删除完成后，该连接将不再出现在您的专用连接列表中。

使用 AWS CLI 删除私有连接

```
aws devops-agent delete-private-connection \
    --name my-mcp-tool-connection
```

响应返回的状态为DELETE_IN_PROGRESS。 AWS DevOps 代理会 ENIs 从您的 VPC 中移除托管资源网关。删除完成后，该连接将不再出现在您的专用连接列表中。

使用现有 VPC 莱迪思资源进行高级设置

如果您的组织已经在使用 Amazon VPC Lattice 并管理自己的资源配置，则可以在自我管理模式下创建私有连接。您无需让 AWS DevOps 代理为您创建资源网关，而是提供指向您的目标服务的现有资源配置的 Amazon 资源名称 (ARN)。

这种方法在以下情况下很有用：

- 想要完全控制资源网关和资源配置生命周期。
- 需要跨多个 AWS 账户或服务共享资源配置。
- 需要使用 VPC Lattice 访问日志进行详细的流量监控。
- 运行 hub-and-spoke网络架构。

要使用 AWS CLI 创建自行管理的私有连接，请执行以下操作：

```
aws devops-agent create-private-connection \
    --name my-advanced-connection \
    --mode '{
        "selfManaged": {
            "resourceConfigurationId": "arn:aws:vpc-lattice:us-
east-1:123456789012:resourceconfiguration/rcfg-0123456789abcdef0"
        }
    }'
```

有关设置 VPC 莱迪思资源网关和资源配置的更多详细信息，请参阅 Amazon VPC 莱迪思用户指南。

相关主题

- the section called “VPC 终端节点 (AWS PrivateLink)”
- the section called “连接 MCP 服务器”
- 为 AWS DevOps 代理配置功能
- AWS DevOps 代理安全
- the section called “DevOps 代理 IAM 权限”

AWS DevOps 代理安全

本文档提供有关 AWS DevOps 代理的安全注意事项、数据保护、访问控制和合规性功能的信息。使用此信息来了解 AWS DevOps Agent 是如何设计来满足您的安全和合规性要求的。

多层安全

AWS DevOps 代理在多层实现安全性。即使向代理的 IAM 角色授予了更广泛的权限，代理也会强制执行自己的内部访问控制以限制其操作范围。例如，如果客户向代理的 IAM 角色添加了完整的 Amazon S3 访问 IAM 策略，则 AWS DevOps 代理将确保仅读取AWSLogs前缀之后的日志以进行故障排除。

我们建议在为 AWS DevOps 代理配置 IAM 权限和实现多层安全性时遵循最低权限原则。深度防御可确保任何错误配置都不会危及您环境的安全。

代理空间

代理空间是 AWS DevOps 代理中的主要安全边界。每个代理空间：

- 使用自己的配置和权限独立运行
- 定义代理可以访问哪些 AWS 账户和资源
- 建立与第三方平台的连接

Agent Spaces 保持严格的隔离，以确保安全并防止跨不同环境或团队的意外访问。

区域处理和数据流

AWS DevOps Agent 在全球范围内运营，具有区域处理能力。代理从已配置的代理空间内被授予访问权限的所有 AWS 账户的 AWS 区域中检索操作数据。这种多区域跨账户数据收集可确保全面的事件分析，同时尊重推理处理的地理边界。

Amazon Bedrock 使用情况和跨区域推理

AWS DevOps 代理将自动选择您所在地理区域内的最佳区域来处理您的推理请求。这样可以最大限度地提高可用计算资源和模型可用性，并提供最佳的客户体验。您的数据将仅存储在创建代理空间的区域中，但是，输入提示和输出结果可能会在该区域之外进行处理，如下表所述。所有数据都将通过 Amazon 的安全网络进行加密传输。

AWS DevOps 代理会将您的推理请求安全地路由到发出请求的地理区域内的可用计算资源，如下所示：

- 来自欧盟的推理请求将在欧盟内部处理。
- 来自美国的推理请求将在美国境内处理。
- 来自澳大利亚的推理请求将在澳大利亚境内处理。
- 来自日本境内的推理请求将在日本国内处理。
- 如果推理请求来自未列出的区域，则默认情况下将在美国境内进行处理。
- DevOps Agent 和 Bedrock 不受服务控制政策 (SCPs) 或 Control Tower 中将客户内容限制在特定区域的客户政策的影响
- Bedrock 可能会使用您所在地理区域内的原始区域以外的区域来执行无状态推理，以优化性能和可用性

Identity and access management

身份验证方法

AWS DevOps 代理提供了两种登录 AWS DevOps 代理空间 Web 应用程序的身份验证方法：

- AWS 身份中心集成 — 主要身份验证方法使用 OAuth 2.0，使用仅限 HTTP 的 Cookie 进行基于会话的身份验证。 AWS 身份中心可以通过标准的 OIDC 和 SAML 协议与外部身份提供商联合，包括 Okta、Ping Identity 和 Microsoft Entra ID 等提供商。此方法支持通过您的身份提供商进行多因素身份验证。 AWS Identity Center 的会话持续时间默认为最长 12 小时，并且可以配置为所需的持续时间。
- IAM 身份验证链接 — 另一种方法允许使用从现有 AWS 管理控制台会话中衍生的基于 JWT 的令牌从 AWS 管理控制台直接访问 Web 应用程序。此选项可用于在实现完整 Identity Center 集成之前评估 AWS DevOps 代理，以及在无法通过基于 Identity Center 的身份验证访问 AWS DevOps 代理 Web 应用程序时获得管理访问权限。会话限制在 10 分钟以内。

IAM 角色

AWS DevOps 代理使用 IAM 角色来定义访问权限：

- 主账户角色-授予代理访问您在其中创建代理空间的 AWS 账户中的资源的访问权限以及次要账户角色的访问权限。

- 次要账户角色-授予代理访问连接到代理空间的其他 AWS 账户中的资源的权限。
- Web 应用程序角色-授予用户在 Web 应用程序中访问 AWS DevOps 代理调查数据和结果的权限。

这些角色应按照最低权限原则进行配置，仅授予调查所需的必要只读权限。

数据保护

数据加密

AWS DevOps 代理对所有客户数据进行加密：

- 静@@ 态加密-所有数据均使用 AWS托管密钥加密。
- 传输中的加密-所有检索到的日志、指标、知识项目、工单元数据和其他数据在传输到代理的专用网络和外部网络时都经过加密。

数据存储和保留

数据存储在创建代理空间的区域，而推理处理可能在您所在的地理区域内进行，如上面的 Amazon Bedrock 使用情况部分所述。

个人身份信息 (PII)

AWS DevOps 在汇总调查、建议评估或聊天回复期间收集的数据时，代理不会过滤 PII 信息。建议先对 PII 数据进行编辑，然后再将其存储在可观察性日志中。

代理日志和审计日志

代理日记

事件调查和预防部门都维护详细的日志，这些日记包括：

- 记录每一个推理步骤和采取的行动
- 实现代理决策过程的完全透明
- 一旦记录下来，代理就无法对其进行修改，从而最大限度地减少了诸如提示注入之类的攻击，使其无法隐藏重要操作

- 包含 “调查” 页面上的所有聊天消息

AWS CloudTrail 整合

所有 AWS DevOps 代理 API 调用 AWS CloudTrail 均由主机 AWS 账户自动捕获。使用收集的信息 CloudTrail，您可以确定：

- 向代理人提出的请求
- 发出请求的 IP 地址
- 发出请求的人员
- 发出请求的时间

即时注射保护

当攻击者将恶意指令嵌入外部数据（例如网页或文档）中时，就会发生提示注入攻击，生成式 AI 系统稍后将处理这些指令。 AWS DevOps 作为其正常操作的一部分，Agent 本机会使用许多数据源，包括日志、资源标签和其他操作数据。 AWS DevOps 代理通过以下保护措施防止即时注入攻击，但重要的是要确保所有连接的数据源和用户对这些数据源的访问都是可信的。有关更多信息，请参见分担责任模型部分。

及时注射的保障措施：

- 写入能力有限 — 除了打开工单和支持案例外，代理可用的工具无法改变资源。这样可以防止恶意指令修改您的基础设施或应用程序。
- 账户边界强制执行 — AWS DevOps 代理只能在主账户和关联的次要 AWS 账户中分配给代理的角色所允许的边界内运作。代理无法访问或修改其配置范围之外的资源。
- AI 安全保护 — AWS DevOps 代理使用具有 AI 安全等级 3 (ASL-3) 保护的模型。这些保护措施包括分类器，这些分类器可在即时注入攻击影响代理行为之前对其进行检测和防止。
- 不可变的审计跟踪 — 代理日志记录每一个推理步骤和采取的行动。记录日记条目后，代理就无法修改日记条目，从而防止提示注入攻击隐藏恶意行为。

虽然 AWS DevOps Agent 提供了针对即时注入攻击的多层保护，但某些配置可能会增加风险：

- 自定义 MCP 服务器工具 — bring-your-own MCP 功能允许您向代理引入自定义工具，这可以为即时注入提供更多机会。自定义工具可能没有与原生 AWS DevOps 代理工具相同的安全控制，恶意指令可能会以意想不到的方式利用这些工具。有关更多信息，请参见分担责任模型部分。

- 授权用户攻击 — 有权在 AWS 账户边界内或关联工具内操作的用户尝试攻击代理的几率更高。这些用户可能能够修改代理使用的数据源，例如日志或资源标签，从而更容易嵌入代理将要处理的恶意指令。

要降低这些风险，请执行以下操作：

1. 在将自定义 MCP 服务器部署到代理空间之前，请仔细检查和测试它们。
   a. 确保他们只能执行只读操作
   b. 验证 MCP 服务器访问的外部工具的用户是否为可信实体，因为与 MCP 接口的 AWS DevOps 代理依赖于这些工具用户和代理之间建立的隐式信任关系 AWS DevOps
2. 在授予用户访问向代理提供数据的系统的权限时，应用最低权限原则
3. 定期审核哪些 MCP 服务器已连接到您的代理空间
4. 由于从许可名单中检索到的任何内容都 URLs 可能试图操纵代理的行为，因此您的许可名单中仅包含可信来源。

集成安全

AWS DevOps 代理支持多种集成类型，每种类型都有自己的安全模型：

- 本机双向集成-内置集成，可以向代理发送数据并从代理接收更新。这使用供应商的身份验证方法
- MCP 服务器 — 使用 OAuth 2.0 身份验证流程和 API 密钥与外部系统安全通信的远程模型上下文协议服务器。
- Webhook 触发器 — 来自远程服务（例如票证或可观测性系统）的调查触发器。Webhook 使用基于哈希的消息身份验证码 (HMAC) 来确保安全。
- 出站通信 — 诸如 Slack 和票务系统之类的集成会从代理接收更新，但尚不支持双向通信。

注册提供商

一些外部工具在账户级别进行身份验证，并在账户中的所有代理空间之间共享。注册这些工具时，您只需在帐户级别进行一次身份验证，然后每个代理空间都可以连接到该注册连接中的特定资源。

以下工具使用账户级注册：

- GitHub— 使用 OAuth 流程进行身份验证。在账户 GitHub 级别注册后，每个 Agent Space 都可以连接到 GitHub 组织内的特定存储库。

- Dynatrac e — 使用 OAuth 令牌身份验证。在账户级别注册 Dynatrace 后，每个代理空间都可以连接到特定的 Dynatrace 环境或监控配置。
- Slack — 使用 OAuth 令牌身份验证。在账户级别注册 Slack 后，每个 Agent Space 都可以连接到特定的 Slack 频道频道。
- Datadog — 使用带有 OAuth 流程的 MCP 进行身份验证。在账户级别注册 Datadog 后，每个代理空间都可以连接到特定的 Datadog 监控资源。
- 新遗物-使用 API 密钥身份验证。在账户级别注册 New Relic 后，每个 Agent Space 都可以连接到特定的 New Relic 监控配置。
- Splunk — 使用不记名令牌身份验证。在账户级别注册 Splunk 后，每个代理空间都可以连接到特定的 Splunk 数据源。
- GitLab— 使用访问令牌身份验证。在账户 GitLab 级别注册后，每个 Agent Space 都可以连接到特定的 GitLab 存储库。
- ServiceNow— 使用 OAuth 客户端 key/token 身份验证。在账户 ServiceNow 级别注册后，每个 Agent Space 都可以连接到特定的 ServiceNow 实例或工单队列。
- 普通公众可访问的远程 MCP 服务器-使用 OAuth 流程进行身份验证。在帐户级别注册远程 MCP 服务器后，每个代理空间都可以连接到该服务器公开的特定资源。

网络连接

AWS DevOps 代理连接到您的第三方系统和远程 MCP 服务器以执行调查和其他操作。

从 AWS DevOps 代理到您的系统的入站流量

AWS DevOps 代理启动与您的第三方系统和远程 MCP 服务器的出站连接，这些连接以入站流量形式到达您的基础架构。如何保护这些流量取决于您的工具托管方式：

- 私有托管工具 — 如果您的工具可以从 AWS VPC 内访问，则可以使用 AWS DevOps 代理私有连接将流量隔离到 AWS 网络，并与公共 Internet 隔离。有关更多信息，请参阅 the section called “连接到私人托管的工具”。
- 公开托管的工具-如果您的工具可通过公共 Internet 访问并使用 IP 许可名单或防火墙规则，则必须允许来自以下 AWS DevOps 代理源 IP 地址的入站流量：
  - 亚太地区（悉尼）(ap-southeast-2)
    - `13.237.95.197`
    - `13.238.84.102`

- 亚太地区（东京）(ap-northeast-1)
  - 13.192.12.233
  - 35.74.181.230
  - 57.183.50.158
- 欧洲地区（法兰克福）(eu-central-1)
  - 18.158.110.140
  - 52.57.96.160
  - 52.59.55.56
- 欧洲地区（爱尔兰）(eu-west-1)
  - 34.251.85.24
  - 52.30.157.157
  - 52.51.192.222
- 美国东部（弗吉尼亚州北部）(us-east-1)
  - 34.228.181.128
  - 44.219.176.187
  - 54.226.244.221
- 美国西部（俄勒冈州）(us-west-2)
  - 34.212.16.133
  - 52.89.67.212
  - 54.187.135.61

从您的 VPC 到 AWS DevOps 代理的出站流量

对于从您的 AWS VPC 到 AWS DevOps 代理的出站流量（例如，使用the section called “通过 Webhook 调用 DevOps 代理”），您可以使用 VPC 终端节点将此网络流量与 AWS 网络隔离。有关更多信息，请参阅 the section called “VPC 终端节点 (AWS PrivateLink)”。

责任共担模式

AWS 责任

AWS 负责：

- 维护代理检索到的数据的安全性
- 保护可供代理使用的本机工具
- 保护运行 AWS DevOps 代理的基础架构

客户责任

客户负责：

- 管理用户对代理空间的访问权限
- 限制向代理提供输入的外部系统的可信用户的访问权限，例如生成日志、 CloudTrail 事件、票证等的服务和资源，这些服务和资源可能被用来尝试恶意提示注入。
- 确保所有连接的数据源都有不太可能被用来尝试提示注入攻击的可信数据
- 确保 bring-your-own MCP 服务器集成安全运行
- 确保分配给代理的 IAM 角色范围正确
- 在存储到可观测性日志和其他代理数据源中之前，先修改 PII 数据
- 遵循建议的做法，即仅向连接的数据源（包括 bring-your-own MCP 服务器）授予只读权限

数据使用情况

AWS 不使用代理数据、聊天消息或来自集成数据源的数据来训练模型或改进产品。 AWS DevOps Agent Space 使用客户的产品内反馈来改善代理的响应和调查，但 AWS 不使用它来改善服务本身。

合规

在预览版中， AWS DevOps Agent 不符合 SOC 2、PCI-DSS、ISO 27001 或 FedRAMP 等标准。 AWS 稍后将宣布将提供哪些合规认证。

DevOps 代理 IAM 权限

AWS DevOps 代理使用特定于服务的 AWS 身份和访问管理 (IAM) Access Management 操作来控制对其功能和功能的访问。这些操作决定了用户在 AWS DevOps 代理控制台和操作员 Web 应用程序中可以执行的操作。这与代理本身用于调查您的资源的 AWS 服务 API 权限是分开的。

有关限制代理访问权限的更多信息，请参阅限制 AWS 账户中的代理访问权限。

代理空间管理操作

以下操作控制对代理空间配置和管理的访问：

- aidevops: GetAgentSpace — 允许用户查看有关代理空间的详细信息，包括其配置、状态和关联账户。用户需要此权限才能访问 AWS 管理控制台中的代理空间。
- aidevops: GetAssociation — 允许用户查看有关特定账户关联的详细信息，包括 IAM 角色配置和连接状态。
- aidevops: ListAssociations — 允许用户列出为代理空间配置的所有 AWS 账户关联，包括主账户和次要账户。

调查和执行行动

以下操作控制对事件调查功能的访问：

- aidevops: ListExecutions — 允许用户查看执行元数据（包括 ID、状态等），用于与任务相关的调查、缓解措施、评估和聊天对话。
- aidevops：ListJournalRecords— 允许用户访问详细日志，这些日志显示了代理的推理步骤、采取的操作以及调查、缓解、评估和聊天对话期间咨询的数据源。这对于了解代理如何得出结论很有用。

聊天管理操作

聊天需要以下 IAM 权限才能运行：

- aidevops: ListChats — 允许用户列出和访问聊天对话历史记录。
- aidevops: CreateChat — 允许用户创建新的聊天对话。
- aidevops: SendMessage — 允许用户提交查询和接收直播回复。

拓扑和发现操作

以下操作控制对应用程序资源映射功能的访问：

- aidevops: DiscoverTopology — 允许用户触发代理空间的拓扑发现和映射。此操作启动扫描 AWS 帐户和构建应用程序资源拓扑的过程。

预防和建议行动

以下操作控制对 “预防” 功能的访问权限：

- aidevops：ListGoals— 允许用户根据最近的事件模式查看代理正在努力实现的预防目标。
- aidevops：ListRecommendations— 允许用户查看 “预防” 功能生成的所有建议，包括其优先级和类别。
- aidevops：GetRecommendation— 允许用户查看有关特定建议的详细信息，包括该建议本可以防止的事件和实施指南。

待办事项任务管理操作

这些操作控制将推荐作为待办事项任务进行管理的能力：

- aidevops: CreateBacklogTask — 允许用户创建事件调查或预防评估任务。
- aidevops：UpdateBacklogTask— 允许用户批准缓解计划或取消正在进行的调查或评估。
- aidevops: GetBacklogTask — 允许用户检索有关特定任务的详细信息。
- aidevops: ListBacklogTasks — 允许用户列出代理空间的任务，按任务类型、状态、优先级或创建时间进行筛选。

知识管理行动

这些操作控制了代理在调查期间可以使用的添加和管理自定义知识的能力：

- aidevops: CreateKnowledgeItem — 允许用户添加自定义知识项，例如技能、故障排除指南或代理应参考的特定于应用程序的信息。
- aidevops: ListKnowledgeItems — 允许用户查看为代理空间配置的所有知识项目。
- aidevops: GetKnowledgeItem — 允许用户检索特定知识项的详细信息。
- aidevops: UpdateKnowledgeItem — 允许用户修改现有知识项目以保持信息最新。
- aidevops: DeleteKnowledgeItem — 允许用户删除不再相关的知识项目。

AWS Support 集成操作

以下操作可控制与 Su AWS pport 案例的集成：

- aidevops：InitiateChatForCase— 允许用户直接通过调查开始与 Su AWS pport 的聊天会话，自动提供有关事件的背景信息。
- aidevops: EndChatForCase — 允许用户结束活跃的 Su AWS pport 案例聊天会话。
- aidevops：DescribeSupportLevel— 允许用户查看账户的 AWS 支持计划级别，以确定可用的支持选项。

使用情况和监控操作

以下操作控制对使用信息的访问权限：

- aidevops: GetAccountUsage — 允许用户查看 AWS DevOps 代理每月的调查时间、预防评估时间和聊天请求配额以及当月的使用情况。

常见的 IAM 策略示例

管理员策略

此政策授予对所有 AWS DevOps 代理功能的完全访问权限：

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

运营商政策

此策略允许访问调查和预防功能，但无需管理权限：

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

```
      "Action": [
        "aidevops:GetAgentSpace",
        "aidevops:InvokeAgent",
        "aidevops:ListExecutions",
        "aidevops:ListJournalRecords",
        "aidevops:ListAssociations",
        "aidevops:GetAssociation",
        "aidevops:DiscoverTopology",
        "aidevops:ListRecommendations",
        "aidevops:GetRecommendation",
        "aidevops:CreateBacklogTask",
        "aidevops:UpdateBacklogTask",
        "aidevops:GetBacklogTask",
        "aidevops:ListBacklogTasks",
        "aidevops:ListKnowledgeItems",
        "aidevops:GetKnowledgeItem",
        "aidevops:InitiateChatForCase",
        "aidevops:EndChatForCase",
        "aidevops:ListChats",
        "aidevops:CreateChat",
        "aidevops:SendMessage",
        "aidevops:ListGoals",
        "aidevops:CreateKnowledgeItem",
        "aidevops:UpdateKnowledgeItem",
        "aidevops:DescribeSupportLevel",
        "aidevops:ListPendingMessages"
      ],
      "Resource": "*"
    }
  ]
}
```

只读策略

此政策授予对调查和建议的访问权限，仅供查看：

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

```
        "aidevops:ListAssociations",
        "aidevops:GetAssociation",
        "aidevops:ListExecutions",
        "aidevops:ListJournalRecords",
        "aidevops:ListRecommendations",
        "aidevops:GetRecommendation",
        "aidevops:ListBacklogTasks",
        "aidevops:GetBacklogTask",
        "aidevops:ListKnowledgeItems",
        "aidevops:GetKnowledgeItem",
        "aidevops:GetAccountUsage"
      ],
      "Resource": "*"
    }
  ]
}
```

为 AWS DevOps 代理使用服务相关角色

AWS DevOps 代理使用 AWS 身份和访问管理 (IAM) Access Management 服务相关角色。服务相关角色是一种独特的 IAM 角色，直接链接到 AWS DevOps 代理。服务相关角色由 AWS DevOps 代理预定义，包括该服务代表您调用其他 AWS 服务所需的所有权限。

服务相关角色权限

AWSServiceRoleForAIDevOps 服务相关角色信任 aidevops.amazonaws.com 服务委托人担任该角色。

该角色使用AWSServiceRoleForAIDevOpsPolicy具有以下权限的托管策略：

- cloudwatch:PutMetricData— 将使用情况指标发布到AWS/AIDevOps CloudWatch 命名空间。由一个cloudwatch:namespace条件限定为仅允许AWS/AIDevOps命名空间。
- vpc-lattice:CreateResourceGateway— 为私有连接创建 VPC 莱迪思资源网关。按aws:RequestTag/AWSAIDevOpsManaged条件设定范围，因此该服务只能创建带有该AWSAIDevOpsManaged标签的资源网关。
- vpc-lattice:TagResource— 标记 VPC 莱迪思资源网关。按aws:RequestTag/AWSAIDevOpsManaged条件划分范围。
- vpc-lattice:DeleteResourceGateway— 删除 VPC 莱迪思资源网关。按aws:ResourceTag/AWSAIDevOpsManaged条件设定范围，因此服务只能删除其创建的资源网关。

- vpc-lattice:GetResourceGateway— 检索有关 VPC 莱迪思资源网关的信息。按aws:ResourceTag/AWSAIDevOpsManaged条件设定范围，因此服务只能读取其创建的资源网关。
- ec2:DescribeVpcs，ec2:DescribeSubnets，ec2:DescribeSecurityGroups— 检索有关配置资源网关所需的 VPC 网络资源的信息。这些只读操作适用于所有 VPC 资源，因为 EC2 API 不支持描述调用的资源级权限。
- iam:CreateServiceLinkedRole— 创建资源网关操作所需的VPC Lattice服务相关角色。此权限仅限于vpc-lattice.amazonaws.com服务主体，不能用于为任何其他服务创建服务相关角色。

创建服务关联角色

无需手动创建 AWSServiceRoleForAIDevOps 服务关联角色。当您开始使用 AWS DevOps 代理时，该服务会为您创建服务相关角色。

要允许服务代表您创建角色，您必须拥有iam:CreateServiceLinkedRole权限。我们建议aidevops.amazonaws.com以遵循最小权限原则为iam:AWSServiceName条件来界定此权限的范围。有关更多信息，请参阅服务相关角色权限。

编辑 服务相关角色

您无法编辑 AWSServiceRoleForAIDevOps 服务相关角色。创建角色后，您无法更改角色的名称，因为各种实体可能会按名称引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅编辑服务相关角色。

删除 服务相关角色

如果您不再需要使用 AWS DevOps 代理，我们建议您删除AWSServiceRoleForAIDevOps服务相关角色。在删除角色之前，必须先删除在代理空间中配置的所有专用连接。删除服务相关角色不会自动移除之前由该服务创建的带有AWSAIDevOpsManaged标签的 VPC Lattice 资源网关。如果不再需要这些资源网关，则应手动将其删除。有关更多信息，请参阅删除服务相关角色。

AWS AWS DevOps 代理的托管策略

AWS 通过提供由创建和管理的独立 IAM 策略来解决许多常见用例 AWS。这些 AWS 托管策略为常见用例授予必要的权限，这样您就可以不必调查需要哪些权限。有关更多信息，请参阅 _IAM 用户指南_ 中的AWS 托管策略。

以下 AWS 托管策略特定于 A AWS DevOps gent，您可以将其附加到账户中的用户。

AIDevOpsAgentReadOnlyAccess

通过 AWS 管理控制台提供对 Amazon A DevOps gent 的只读访问权限

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AIDevOpsAgentReadOnlyAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:Get*",
    "aidevops:List*",
    "aidevops:SearchServiceAccessibleResource"
   ],
   "Resource": "*"
  }
 ]
}
```

AIDevOpsAgentFullAccess

通过 AWS 管理控制台提供对 Amazon A DevOps gent 的完全访问权限

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AIDevOpsAgentSpaceAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateAgentSpace",
    "aidevops:DeleteAgentSpace",
    "aidevops:GetAgentSpace",
    "aidevops:ListAgentSpaces",
    "aidevops:UpdateAgentSpace"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsServiceAccess",
   "Effect": "Allow",
   "Action": [
```

```
    "aidevops:DeregisterService",
    "aidevops:GetService",
    "aidevops:ListServices",
    "aidevops:RegisterService",
    "aidevops:SearchServiceAccessibleResource"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsAssociationAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:AssociateService",
    "aidevops:DisassociateService",
    "aidevops:GetAssociation",
    "aidevops:ListAssociations",
    "aidevops:UpdateAssociation",
    "aidevops:ValidateAwsAssociations"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsWebhookAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:ListWebhooks"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsOperatorAppAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:DisableOperatorApp",
    "aidevops:EnableOperatorApp",
    "aidevops:GetOperatorApp",
    "aidevops:UpdateOperatorAppIdpConfig"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsKnowledgeAccess",
   "Effect": "Allow",
   "Action": [
```

```
    "aidevops:CreateKnowledgeItem",
    "aidevops:DeleteKnowledgeItem",
    "aidevops:GetKnowledgeItem",
    "aidevops:ListKnowledgeItems",
    "aidevops:ListKnowledgeItemVersions",
    "aidevops:UpdateKnowledgeItem"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsBacklogAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateBacklogTask",
    "aidevops:GetBacklogTask",
    "aidevops:ListBacklogTasks",
    "aidevops:ListGoals",
    "aidevops:UpdateBacklogTask",
    "aidevops:UpdateGoal"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsRecommendationAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetRecommendation",
    "aidevops:ListRecommendations",
    "aidevops:UpdateRecommendation"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AIDevOpsAgentChatAccess",
   "Effect": "Allow",
   "Action": [
    "aidevops:CreateChat",
    "aidevops:ListChats",
    "aidevops:ListPendingMessages",
    "aidevops:SendMessage"
   ],
   "Resource": "*"
  },
  {
```

```
    "Sid": "AIDevOpsJournalAccess",
    "Effect": "Allow",
    "Action": [
     "aidevops:ListExecutions",
     "aidevops:ListJournalRecords"
    ],
    "Resource": "*"
   },
   {
    "Sid": "AIDevOpsTopologyAccess",
    "Effect": "Allow",
    "Action": [
     "aidevops:DiscoverTopology"
    ],
    "Resource": "*"
   },
   {
    "Sid": "AIDevOpsSupportAccess",
    "Effect": "Allow",
    "Action": [
     "aidevops:DescribeSupportLevel",
     "aidevops:EndChatForCase",
     "aidevops:InitiateChatForCase"
    ],
    "Resource": "*"
   },
   {
    "Sid": "AIDevOpsUsageAccess",
    "Effect": "Allow",
    "Action": [
     "aidevops:GetAccountUsage"
    ],
    "Resource": "*"
   },
   {
    "Sid": "AIDevOpsTaggingAccess",
    "Effect": "Allow",
    "Action": [
     "aidevops:ListTagsForResource",
     "aidevops:TagResource",
     "aidevops:UntagResource"
    ],
    "Resource": "*"
   },
```

```
  {
   "Sid": "AIDevOpsVendedLogs",
   "Effect": "Allow",
   "Action": [
    "aidevops:AllowVendedLogDeliveryForResource"
   ],
   "Resource": "*"
  }
 ]
}
```

AIDevOpsOperatorAppAccessPolicy

提供使用 AWS DevOps 操作员 Web 应用程序访问代理空间的权限。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowOperatorAgentSpaceActions",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetAgentSpace",
    "aidevops:GetAssociation",
    "aidevops:ListAssociations",
    "aidevops:CreateBacklogTask",
    "aidevops:GetBacklogTask",
    "aidevops:UpdateBacklogTask",
    "aidevops:ListBacklogTasks",
    "aidevops:ListJournalRecords",
    "aidevops:DiscoverTopology",
    "aidevops:ListGoals",
    "aidevops:ListRecommendations",
    "aidevops:ListExecutions",
    "aidevops:GetRecommendation",
    "aidevops:UpdateRecommendation",
    "aidevops:CreateKnowledgeItem",
    "aidevops:ListKnowledgeItems",
    "aidevops:ListKnowledgeItemVersions",
    "aidevops:GetKnowledgeItem",
    "aidevops:UpdateKnowledgeItem",
    "aidevops:DeleteKnowledgeItem",
    "aidevops:ListPendingMessages",
```

```
    "aidevops:InitiateChatForCase",
    "aidevops:EndChatForCase",
    "aidevops:DescribeSupportLevel",
    "aidevops:ListChats",
    "aidevops:CreateChat",
    "aidevops:SendMessage"
   ],
   "Resource": "arn:aws:aidevops:*:*:agentspace/${aws:PrincipalTag/AgentSpaceId}",
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  },
  {
   "Sid": "AllowOperatorAccountActions",
   "Effect": "Allow",
   "Action": [
    "aidevops:GetAccountUsage"
   ],
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  },
  {
   "Sid": "AllowSupportOperatorActions",
   "Effect": "Allow",
   "Action": [
    "support:DescribeCases",
    "support:InitiateChatForCase",
    "support:DescribeSupportLevel"
   ],
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  }
 ]
```

```
}
```

AIDevOpsAgentAccessPolicy

提供 AWS DevOps 代理对客户 AWS 资源进行调查和分析所需的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AIOPSServiceAccess",
            "Effect": "Allow",
            "Action": [
                "access-analyzer:GetAnalyzer",
                "access-analyzer:List*",
                "acm-pca:Describe*",
                "acm-pca:GetCertificate",
                "acm-pca:GetCertificateAuthorityCertificate",
                "acm-pca:GetCertificateAuthorityCsr",
                "acm-pca:List*",
                "acm:DescribeCertificate",
                "acm:GetAccountConfiguration",
                "aidevops:GetKnowledgeItem",
                "aidevops:ListKnowledgeItems",
                "airflow:List*",
                "amplify:GetApp",
                "amplify:GetBranch",
                "amplify:GetDomainAssociation",
                "amplify:List*",
                "aoss:BatchGetCollection",
                "aoss:BatchGetLifecyclePolicy",
                "aoss:BatchGetVpcEndpoint",
                "aoss:GetAccessPolicy",
                "aoss:GetSecurityConfig",
                "aoss:GetSecurityPolicy",
                "aoss:List*",
                "appconfig:GetApplication",
                "appconfig:GetConfigurationProfile",
                "appconfig:GetEnvironment",
                "appconfig:GetHostedConfigurationVersion",
                "appconfig:List*",
                "appflow:Describe*",
                "appflow:List*",
```

```
                "application-autoscaling:Describe*",
                "application-signals:BatchGetServiceLevelObjectiveBudgetReport",
                "application-signals:GetService",
                "application-signals:GetServiceLevelObjective",
                "application-signals:List*",
                "applicationinsights:Describe*",
                "applicationinsights:List*",
                "apprunner:Describe*",
                "apprunner:List*",
                "appstream:Describe*",
                "appstream:List*",
                "appsync:GetApiAssociation",
                "appsync:GetDataSource",
                "appsync:GetDomainName",
                "appsync:GetFunction",
                "appsync:GetGraphqlApi",
                "appsync:GetGraphqlApiEnvironmentVariables",
                "appsync:GetIntrospectionSchema",
                "appsync:GetResolver",
                "appsync:GetSourceApiAssociation",
                "appsync:List*",
                "aps:Describe*",
                "aps:List*",
                "arc-zonal-shift:GetManagedResource",
                "arc-zonal-shift:List*",
                "athena:GetCapacityAssignmentConfiguration",
                "athena:GetCapacityReservation",
                "athena:GetDataCatalog",
                "athena:GetNamedQuery",
                "athena:GetPreparedStatement",
                "athena:GetWorkGroup",
                "athena:List*",
                "auditmanager:GetAssessment",
                "auditmanager:List*",
                "autoscaling:Describe*",
                "backup-gateway:GetHypervisor",
                "backup-gateway:List*",
                "backup:Describe*",
                "backup:GetBackupPlan",
                "backup:GetBackupSelection",
                "backup:GetBackupVaultAccessPolicy",
                "backup:GetBackupVaultNotifications",
                "backup:GetRestoreTestingPlan",
                "backup:GetRestoreTestingSelection",
```

```
                "backup:List*",
                "batch:DescribeComputeEnvironments",
                "batch:DescribeJobQueues",
                "batch:DescribeSchedulingPolicies",
                "batch:List*",
                "bedrock:GetAgent",
                "bedrock:GetAgentActionGroup",
                "bedrock:GetAgentAlias",
                "bedrock:GetAgentKnowledgeBase",
                "bedrock:GetDataSource",
                "bedrock:GetGuardrail",
                "bedrock:GetKnowledgeBase",
                "bedrock:List*",
                "budgets:Describe*",
                "budgets:List*",
                "ce:Describe*",
                "ce:GetAnomalyMonitors",
                "ce:GetAnomalySubscriptions",
                "ce:List*",
                "chatbot:Describe*",
                "chatbot:GetMicrosoftTeamsChannelConfiguration",
                "chatbot:List*",
                "cleanrooms-ml:GetTrainingDataset",
                "cleanrooms-ml:List*",
                "cleanrooms:GetAnalysisTemplate",
                "cleanrooms:GetCollaboration",
                "cleanrooms:GetConfiguredTable",
                "cleanrooms:GetConfiguredTableAnalysisRule",
                "cleanrooms:GetConfiguredTableAssociation",
                "cleanrooms:GetMembership",
                "cleanrooms:List*",
                "cloudformation:Describe*",
                "cloudformation:GetResource",
                "cloudformation:GetStackPolicy",
                "cloudformation:GetTemplate",
                "cloudformation:List*",
                "cloudfront:Describe*",
                "cloudfront:GetCachePolicy",
                "cloudfront:GetCloudFrontOriginAccessIdentity",
                "cloudfront:GetContinuousDeploymentPolicy",
                "cloudfront:GetDistribution",
                "cloudfront:GetDistributionConfig",
                "cloudfront:GetFunction",
                "cloudfront:GetKeyGroup",
```

```
                "cloudfront:GetMonitoringSubscription",
                "cloudfront:GetOriginAccessControl",
                "cloudfront:GetOriginRequestPolicy",
                "cloudfront:GetPublicKey",
                "cloudfront:GetRealtimeLogConfig",
                "cloudfront:GetResponseHeadersPolicy",
                "cloudfront:List*",
                "cloudtrail:Describe*",
                "cloudtrail:GetChannel",
                "cloudtrail:GetEventConfiguration",
                "cloudtrail:GetEventDataStore",
                "cloudtrail:GetEventSelectors",
                "cloudtrail:GetInsightSelectors",
                "cloudtrail:GetQueryResults",
                "cloudtrail:GetResourcePolicy",
                "cloudtrail:GetTrail",
                "cloudtrail:GetTrailStatus",
                "cloudtrail:List*",
                "cloudtrail:LookupEvents",
                "cloudtrail:StartQuery",
                "cloudwatch:Describe*",
                "cloudwatch:GenerateQuery",
                "cloudwatch:GetDashboard",
                "cloudwatch:GetInsightRuleReport",
                "cloudwatch:GetMetricData",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:GetMetricStream",
                "cloudwatch:GetService",
                "cloudwatch:GetServiceLevelObjective",
                "cloudwatch:List*",
                "codeartifact:Describe*",
                "codeartifact:GetDomainPermissionsPolicy",
                "codeartifact:GetRepositoryPermissionsPolicy",
                "codeartifact:List*",
                "codebuild:BatchGetFleets",
                "codebuild:List*",
                "codecommit:GetRepository",
                "codecommit:GetRepositoryTriggers",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetDeploymentTargets",
                "codedeploy:GetApplication",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:List*",
```

```
                "codeguru-profiler:Describe*",
                "codeguru-profiler:GetNotificationConfiguration",
                "codeguru-profiler:GetPolicy",
                "codeguru-profiler:List*",
                "codeguru-reviewer:Describe*",
                "codeguru-reviewer:List*",
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:List*",
                "codestar-connections:GetConnection",
                "codestar-connections:GetRepositoryLink",
                "codestar-connections:GetSyncConfiguration",
                "codestar-connections:List*",
                "codestar-notifications:Describe*",
                "codestar-notifications:List*",
                "cognito-identity:DescribeIdentityPool",
                "cognito-identity:GetIdentityPoolRoles",
                "cognito-identity:ListIdentityPools",
                "cognito-identity:ListTagsForResource",
                "cognito-idp:AdminListGroupsForUser",
                "cognito-idp:DescribeIdentityProvider",
                "cognito-idp:DescribeResourceServer",
                "cognito-idp:DescribeRiskConfiguration",
                "cognito-idp:DescribeUserImportJob",
                "cognito-idp:DescribeUserPool",
                "cognito-idp:DescribeUserPoolDomain",
                "cognito-idp:GetGroup",
                "cognito-idp:GetLogDeliveryConfiguration",
                "cognito-idp:GetUICustomization",
                "cognito-idp:GetUserPoolMfaConfig",
                "cognito-idp:GetWebACLForResource",
                "cognito-idp:ListGroups",
                "cognito-idp:ListIdentityProviders",
                "cognito-idp:ListResourceServers",
                "cognito-idp:ListUserPoolClients",
                "cognito-idp:ListUserPools",
                "cognito-idp:ListTagsForResource",
                "comprehend:Describe*",
                "comprehend:List*",
                "config:Describe*",
                "config:GetStoredQuery",
                "config:List*",
                "connect:Describe*",
                "connect:GetTaskTemplate",
```

```
                "connect:List*",
                "databrew:Describe*",
                "databrew:List*",
                "datapipeline:Describe*",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:List*",
                "datasync:Describe*",
                "datasync:List*",
                "deadline:GetFarm",
                "deadline:GetFleet",
                "deadline:GetLicenseEndpoint",
                "deadline:GetMonitor",
                "deadline:GetQueue",
                "deadline:GetQueueEnvironment",
                "deadline:GetQueueFleetAssociation",
                "deadline:GetStorageProfile",
                "deadline:List*",
                "detective:GetMembers",
                "detective:List*",
                "devicefarm:GetDevicePool",
                "devicefarm:GetInstanceProfile",
                "devicefarm:GetNetworkProfile",
                "devicefarm:GetProject",
                "devicefarm:GetTestGridProject",
                "devicefarm:GetVPCEConfiguration",
                "devicefarm:List*",
                "devops-guru:Describe*",
                "devops-guru:GetResourceCollection",
                "devops-guru:List*",
                "dms:Describe*",
                "dms:List*",
                "ds:Describe*",
                "dynamodb:Describe*",
                "dynamodb:GetResourcePolicy",
                "dynamodb:List*",
                "ec2:Describe*",
                "ec2:GetAssociatedEnclaveCertificateIamRoles",
                "ec2:GetIpamPoolAllocations",
                "ec2:GetIpamPoolCidrs",
                "ec2:GetManagedPrefixListEntries",
                "ec2:GetNetworkInsightsAccessScopeContent",
                "ec2:GetSnapshotBlockPublicAccessState",
                "ec2:GetTransitGatewayMulticastDomainAssociations",
                "ec2:GetTransitGatewayRouteTableAssociations",
```

```
                "ec2:GetTransitGatewayRouteTablePropagations",
                "ec2:GetVerifiedAccessEndpointPolicy",
                "ec2:GetVerifiedAccessGroupPolicy",
                "ec2:GetVerifiedAccessInstanceWebAcl",
                "ec2:SearchLocalGatewayRoutes",
                "ec2:SearchTransitGatewayRoutes",
                "ecr:Describe*",
                "ecr:GetLifecyclePolicy",
                "ecr:GetRegistryPolicy",
                "ecr:GetRepositoryPolicy",
                "ecr:List*",
                "ecs:Describe*",
                "ecs:List*",
                "eks:AccessKubernetesApi",
                "eks:Describe*",
                "eks:List*",
                "elasticache:Describe*",
                "elasticache:List*",
                "elasticbeanstalk:Describe*",
                "elasticbeanstalk:List*",
                "elasticfilesystem:Describe*",
                "elasticloadbalancing:GetResourcePolicy",
                "elasticloadbalancing:GetTrustStoreCaCertificatesBundle",
                "elasticloadbalancing:GetTrustStoreRevocationContent",
                "elasticloadbalancing:Describe*",
                "elasticmapreduce:Describe*",
                "elasticmapreduce:List*",
                "emr-containers:Describe*",
                "emr-containers:List*",
                "emr-serverless:GetApplication",
                "emr-serverless:List*",
                "es:Describe*",
                "es:List*",
                "events:Describe*",
                "events:List*",
                "evidently:GetExperiment",
                "evidently:GetFeature",
                "evidently:GetLaunch",
                "evidently:GetProject",
                "evidently:GetSegment",
                "evidently:List*",
                "firehose:Describe*",
                "firehose:List*",
                "fis:GetExperimentTemplate",
```

```
                "fis:GetTargetAccountConfiguration",
                "fis:List*",
                "fms:GetNotificationChannel",
                "fms:GetPolicy",
                "fms:List*",
                "forecast:Describe*",
                "forecast:List*",
                "frauddetector:BatchGetVariable",
                "frauddetector:Describe*",
                "frauddetector:GetDetectors",
                "frauddetector:GetDetectorVersion",
                "frauddetector:GetEntityTypes",
                "frauddetector:GetEventTypes",
                "frauddetector:GetExternalModels",
                "frauddetector:GetLabels",
                "frauddetector:GetListElements",
                "frauddetector:GetListsMetadata",
                "frauddetector:GetModelVersion",
                "frauddetector:GetOutcomes",
                "frauddetector:GetRules",
                "frauddetector:GetVariables",
                "frauddetector:List*",
                "fsx:Describe*",
                "gamelift:Describe*",
                "gamelift:List*",
                "globalaccelerator:Describe*",
                "globalaccelerator:List*",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetJob",
                "glue:GetRegistry",
                "glue:GetSchema",
                "glue:GetSchemaVersion",
                "glue:GetTable",
                "glue:GetTags",
                "glue:GetTrigger",
                "glue:List*",
                "glue:querySchemaVersionMetadata",
                "grafana:Describe*",
                "grafana:List*",
                "greengrass:Describe*",
                "greengrass:GetDeployment",
                "greengrass:List*",
                "groundstation:GetConfig",
```

```
                "groundstation:GetDataflowEndpointGroup",
                "groundstation:GetMissionProfile",
                "groundstation:List*",
                "guardduty:GetDetector",
                "guardduty:GetFilter",
                "guardduty:GetIPSet",
                "guardduty:GetMalwareProtectionPlan",
                "guardduty:GetMasterAccount",
                "guardduty:GetMembers",
                "guardduty:GetThreatIntelSet",
                "guardduty:List*",
                "health:DescribeEvents",
                "health:DescribeEventDetails",
                "healthlake:Describe*",
                "healthlake:List*",
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetInstanceProfile",
                "iam:GetLoginProfile",
                "iam:GetOpenIDConnectProvider",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetSAMLProvider",
                "iam:GetServerCertificate",
                "iam:GetServiceLinkedRoleDeletionStatus",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListOpenIDConnectProviders",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListServerCertificates",
                "iam:ListVirtualMFADevices",
                "identitystore:DescribeGroup",
                "identitystore:DescribeGroupMembership",
                "identitystore:ListGroupMemberships",
                "identitystore:ListGroups",
                "imagebuilder:GetComponent",
                "imagebuilder:GetContainerRecipe",
                "imagebuilder:GetDistributionConfiguration",
                "imagebuilder:GetImage",
                "imagebuilder:GetImagePipeline",
```

```
                "imagebuilder:GetImageRecipe",
                "imagebuilder:GetInfrastructureConfiguration",
                "imagebuilder:GetLifecyclePolicy",
                "imagebuilder:GetWorkflow",
                "imagebuilder:List*",
                "inspector2:List*",
                "inspector:Describe*",
                "inspector:List*",
                "internetmonitor:GetMonitor",
                "internetmonitor:List*",
                "iot:Describe*",
                "iot:GetPackage",
                "iot:GetPackageVersion",
                "iot:GetPolicy",
                "iot:GetThingShadow",
                "iot:GetTopicRule",
                "iot:GetTopicRuleDestination",
                "iot:GetV2LoggingOptions",
                "iot:List*",
                "iotanalytics:Describe*",
                "iotanalytics:List*",
                "iotevents:Describe*",
                "iotevents:List*",
                "iotsitewise:Describe*",
                "iotsitewise:List*",
                "iotwireless:GetDestination",
                "iotwireless:GetDeviceProfile",
                "iotwireless:GetFuotaTask",
                "iotwireless:GetMulticastGroup",
                "iotwireless:GetNetworkAnalyzerConfiguration",
                "iotwireless:GetServiceProfile",
                "iotwireless:GetWirelessDevice",
                "iotwireless:GetWirelessGateway",
                "iotwireless:GetWirelessGatewayTaskDefinition",
                "iotwireless:List*",
                "ivs:GetChannel",
                "ivs:GetEncoderConfiguration",
                "ivs:GetPlaybackRestrictionPolicy",
                "ivs:GetRecordingConfiguration",
                "ivs:GetStage",
                "ivs:List*",
                "ivschat:GetLoggingConfiguration",
                "ivschat:GetRoom",
                "ivschat:List*",
```

```
                "kafka:Describe*",
                "kafka:GetClusterPolicy",
                "kafka:List*",
                "kafkaconnect:Describe*",
                "kafkaconnect:List*",
                "kendra:Describe*",
                "kendra:List*",
                "kinesis:Describe*",
                "kinesis:GetResourcePolicy",
                "kinesis:List*",
                "kinesisanalytics:Describe*",
                "kinesisanalytics:List*",
                "kinesisvideo:Describe*",
                "kms:DescribeKey",
                "kms:ListResourceTags",
                "kms:ListKeys",
                "kms:GetKeyPolicy",
                "kms:GetKeyRotationStatus",
                "kms:ListAliases",
                "kms:ListKeyRotations",
                "lakeformation:Describe*",
                "lakeformation:GetLFTag",
                "lakeformation:GetResourceLFTags",
                "lakeformation:List*",
                "lambda:GetAlias",
                "lambda:GetCodeSigningConfig",
                "lambda:GetEventSourceMapping",
                "lambda:GetFunctionCodeSigningConfig",
                "lambda:GetFunctionConfiguration",
                "lambda:GetFunctionEventInvokeConfig",
                "lambda:GetFunctionRecursionConfig",
                "lambda:GetFunctionUrlConfig",
                "lambda:GetLayerVersion",
                "lambda:GetLayerVersionPolicy",
                "lambda:GetPolicy",
                "lambda:GetProvisionedConcurrencyConfig",
                "lambda:GetRuntimeManagementConfig",
                "lambda:List*",
                "launchwizard:GetDeployment",
                "launchwizard:List*",
                "license-manager:GetLicense",
                "license-manager:List*",
                "lightsail:GetAlarms",
                "lightsail:GetBuckets",
```

```
                "lightsail:GetCertificates",
                "lightsail:GetContainerServices",
                "lightsail:GetDisk",
                "lightsail:GetDisks",
                "lightsail:GetInstance",
                "lightsail:GetInstances",
                "lightsail:GetLoadBalancer",
                "lightsail:GetLoadBalancers",
                "lightsail:GetLoadBalancerTlsCertificates",
                "lightsail:GetStaticIp",
                "lightsail:GetStaticIps",
                "logs:Describe*",
                "logs:FilterLogEvents",
                "logs:GetDataProtectionPolicy",
                "logs:GetDelivery",
                "logs:GetDeliveryDestination",
                "logs:GetDeliveryDestinationPolicy",
                "logs:GetDeliverySource",
                "logs:GetLogAnomalyDetector",
                "logs:GetLogDelivery",
                "logs:GetLogGroupFields",
                "logs:GetQueryResults",
                "logs:List*",
                "logs:StartQuery",
                "logs:StopLiveTail",
                "logs:StopQuery",
                "logs:TestMetricFilter",
                "m2:GetApplication",
                "m2:GetEnvironment",
                "m2:List*",
                "macie2:GetAllowList",
                "macie2:GetCustomDataIdentifier",
                "macie2:GetFindingsFilter",
                "macie2:GetMacieSession",
                "macie2:List*",
                "mediaconnect:Describe*",
                "mediaconnect:List*",
                "medialive:Describe*",
                "medialive:GetCloudWatchAlarmTemplate",
                "medialive:GetCloudWatchAlarmTemplateGroup",
                "medialive:GetEventBridgeRuleTemplate",
                "medialive:GetEventBridgeRuleTemplateGroup",
                "medialive:GetSignalMap",
                "medialive:List*",
```

```
                "mediapackage-vod:Describe*",
                "mediapackage-vod:List*",
                "mediapackage:Describe*",
                "mediapackage:List*",
                "mediapackagev2:GetChannel",
                "mediapackagev2:GetChannelGroup",
                "mediapackagev2:GetChannelPolicy",
                "mediapackagev2:GetOriginEndpoint",
                "mediapackagev2:GetOriginEndpointPolicy",
                "mediapackagev2:List*",
                "memorydb:Describe*",
                "memorydb:List*",
                "mobiletargeting:GetInAppTemplate",
                "mobiletargeting:List*",
                "mq:Describe*",
                "mq:List*",
                "network-firewall:Describe*",
                "network-firewall:List*",
                "networkmanager:Describe*",
                "networkmanager:GetConnectAttachment",
                "networkmanager:GetConnectPeer",
                "networkmanager:GetCoreNetwork",
                "networkmanager:GetCoreNetworkPolicy",
                "networkmanager:GetCustomerGatewayAssociations",
                "networkmanager:GetDevices",
                "networkmanager:GetLinkAssociations",
                "networkmanager:GetLinks",
                "networkmanager:GetSites",
                "networkmanager:GetSiteToSiteVpnAttachment",
                "networkmanager:GetTransitGatewayPeering",
                "networkmanager:GetTransitGatewayRegistrations",
                "networkmanager:GetTransitGatewayRouteTableAttachment",
                "networkmanager:GetVpcAttachment",
                "networkmanager:List*",
                "oam:GetLink",
                "oam:GetSink",
                "oam:GetSinkPolicy",
                "oam:List*",
                "omics:GetAnnotationStore",
                "omics:GetReferenceStore",
                "omics:GetRunGroup",
                "omics:GetSequenceStore",
                "omics:GetVariantStore",
                "omics:GetWorkflow",
```

```
                "omics:List*",
                "organizations:Describe*",
                "organizations:List*",
                "osis:GetPipeline",
                "osis:List*",
                "payment-cryptography:GetAlias",
                "payment-cryptography:GetKey",
                "payment-cryptography:List*",
                "pca-connector-ad:GetConnector",
                "pca-connector-ad:GetDirectoryRegistration",
                "pca-connector-ad:GetServicePrincipalName",
                "pca-connector-ad:GetTemplate",
                "pca-connector-ad:GetTemplateGroupAccessControlEntry",
                "pca-connector-ad:List*",
                "pca-connector-scep:GetChallengeMetadata",
                "pca-connector-scep:GetConnector",
                "pca-connector-scep:List*",
                "personalize:Describe*",
                "personalize:List*",
                "pi:DescribeDimensionKeys",
                "pi:GetResourceMetadata",
                "pi:GetResourceMetrics",
                "pi:ListAvailableResourceDimensions",
                "pi:ListAvailableResourceMetrics",
                "pipes:Describe*",
                "pipes:List*",
                "proton:GetEnvironmentTemplate",
                "proton:GetServiceTemplate",
                "proton:List*",
                "qbusiness:GetApplication",
                "qbusiness:GetDataSource",
                "qbusiness:GetIndex",
                "qbusiness:GetPlugin",
                "qbusiness:GetRetriever",
                "qbusiness:GetWebExperience",
                "qbusiness:List*",
                "ram:GetPermission",
                "ram:GetResourceShares",
                "ram:List*",
                "rds:Describe*",
                "rds:List*",
                "redshift-serverless:GetNamespace",
                "redshift-serverless:GetWorkgroup",
                "redshift-serverless:List*",
```

```
                "redshift:Describe*",
                "refactor-spaces:GetApplication",
                "refactor-spaces:GetEnvironment",
                "refactor-spaces:GetRoute",
                "refactor-spaces:List*",
                "rekognition:Describe*",
                "rekognition:List*",
                "resiliencehub:Describe*",
                "resiliencehub:List*",
                "resource-explorer-2:GetDefaultView",
                "resource-explorer-2:GetIndex",
                "resource-explorer-2:GetView",
                "resource-explorer-2:List*",
                "resource-explorer-2:Search",
                "resource-groups:GetGroup",
                "resource-groups:GetGroupConfiguration",
                "resource-groups:GetGroupQuery",
                "resource-groups:GetTags",
                "resource-groups:List*",
                "route53-recovery-control-config:Describe*",
                "route53-recovery-control-config:List*",
                "route53-recovery-readiness:GetCell",
                "route53-recovery-readiness:GetReadinessCheck",
                "route53-recovery-readiness:GetRecoveryGroup",
                "route53-recovery-readiness:GetResourceSet",
                "route53-recovery-readiness:List*",
                "route53:GetDNSSEC",
                "route53:GetHealthCheck",
                "route53:GetHealthCheckStatus",
                "route53:GetHostedZone",
                "route53:List*",
                "route53profiles:GetProfile",
                "route53profiles:GetProfileAssociation",
                "route53profiles:GetProfileResourceAssociation",
                "route53profiles:List*",
                "route53resolver:GetFirewallDomainList",
                "route53resolver:GetFirewallRuleGroup",
                "route53resolver:GetFirewallRuleGroupAssociation",
                "route53resolver:GetOutpostResolver",
                "route53resolver:GetResolverConfig",
                "route53resolver:GetResolverQueryLogConfig",
                "route53resolver:GetResolverQueryLogConfigAssociation",
                "route53resolver:GetResolverRule",
                "route53resolver:GetResolverRuleAssociation",
```

```
                "route53resolver:List*",
                "rum:GetAppMonitor",
                "rum:List*",
                "s3-outposts:ListEndpoints",
                "s3-outposts:ListOutpostsWithS3",
                "s3:GetAccessGrant",
                "s3:GetAccessGrantsInstance",
                "s3:GetAccessGrantsLocation",
                "s3:GetAccessPoint",
                "s3:GetAccessPointConfigurationForObjectLambda",
                "s3:GetAccessPointForObjectLambda",
                "s3:GetAccessPointPolicy",
                "s3:GetAccessPointPolicyForObjectLambda",
                "s3:GetAccessPointPolicyStatusForObjectLambda",
                "s3:GetBucketAbac",
                "s3:GetBucketAcl",
                "s3:GetBucketCORS",
                "s3:GetBucketLocation",
                "s3:GetBucketLogging",
                "s3:GetBucketMetadataTableConfiguration",
                "s3:GetBucketNotification",
                "s3:GetBucketObjectLockConfiguration",
                "s3:GetBucketOwnershipControls",
                "s3:GetBucketPolicy",
                "s3:GetBucketPublicAccessBlock",
                "s3:GetBucketTagging",
                "s3:GetBucketVersioning",
                "s3:GetEncryptionConfiguration",
                "s3:GetLifecycleConfiguration",
                "s3:GetMultiRegionAccessPoint",
                "s3:GetMultiRegionAccessPointPolicy",
                "s3:GetMultiRegionAccessPointPolicyStatus",
                "s3:GetReplicationConfiguration",
                "s3:GetStorageLensConfiguration",
                "s3:GetStorageLensConfigurationTagging",
                "s3:GetStorageLensGroup",
                "s3:ListAllMyBuckets",
                "sagemaker:Describe*",
                "sagemaker:List*",
                "scheduler:GetSchedule",
                "scheduler:GetScheduleGroup",
                "scheduler:List*",
                "schemas:Describe*",
                "schemas:GetResourcePolicy",
```

```
                "schemas:List*",
                "secretsmanager:Describe*",
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:List*",
                "securityhub:BatchGetAutomationRules",
                "securityhub:BatchGetSecurityControls",
                "securityhub:Describe*",
                "securityhub:GetConfigurationPolicy",
                "securityhub:GetConfigurationPolicyAssociation",
                "securityhub:GetEnabledStandards",
                "securityhub:GetFindingAggregator",
                "securityhub:GetInsights",
                "securityhub:List*",
                "securitylake:GetSubscriber",
                "securitylake:List*",
                "servicecatalog:Describe*",
                "servicecatalog:GetApplication",
                "servicecatalog:GetAttributeGroup",
                "servicecatalog:List*",
                "servicequotas:GetServiceQuota",
                "ses:Describe*",
                "ses:GetAccount",
                "ses:GetAddonInstance",
                "ses:GetAddonSubscription",
                "ses:GetArchive",
                "ses:GetConfigurationSet",
                "ses:GetConfigurationSetEventDestinations",
                "ses:GetContactList",
                "ses:GetDedicatedIpPool",
                "ses:GetDedicatedIps",
                "ses:GetEmailIdentity",
                "ses:GetEmailTemplate",
                "ses:GetIngressPoint",
                "ses:GetRelay",
                "ses:GetRuleSet",
                "ses:GetTemplate",
                "ses:GetTrafficPolicy",
                "ses:List*",
                "shield:Describe*",
                "shield:List*",
                "signer:GetSigningProfile",
                "signer:List*",
                "sns:GetDataProtectionPolicy",
                "sns:GetSubscriptionAttributes",
```

```
                "sns:GetTopicAttributes",
                "sns:List*",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:List*",
                "ssm-contacts:GetContact",
                "ssm-contacts:GetContactChannel",
                "ssm-contacts:List*",
                "ssm-incidents:GetReplicationSet",
                "ssm-incidents:GetResponsePlan",
                "ssm-incidents:List*",
                "ssm-sap:GetApplication",
                "ssm-sap:List*",
                "ssm:Describe*",
                "ssm:GetDefaultPatchBaseline",
                "ssm:GetDocument",
                "ssm:GetParameters",
                "ssm:GetPatchBaseline",
                "ssm:GetResourcePolicies",
                "ssm:List*",
                "sso:GetInlinePolicyForPermissionSet",
                "sso:GetManagedApplicationInstance",
                "sso:GetPermissionsBoundaryForPermissionSet",
                "sso:GetSharedSsoConfiguration",
                "sso:ListAccountAssignments",
                "sso:ListApplicationAssignments",
                "sso:ListApplications",
                "sso:ListCustomerManagedPolicyReferencesInPermissionSet",
                "sso:ListInstances",
                "sso:ListManagedPoliciesInPermissionSet",
                "sso:ListTagsForResource",
                "states:GetExecutionHistory",
                "states:Describe*",
                "states:List*",
                "support:CreateCase",
                "support:DescribeCases",
                "synthetics:Describe*",
                "synthetics:GetCanary",
                "synthetics:GetCanaryRuns",
                "synthetics:GetGroup",
                "synthetics:List*",
                "tag:GetResources",
                "timestream:Describe*",
                "timestream:List*",
```

```
                "transfer:Describe*",
                "transfer:List*",
                "verifiedpermissions:GetIdentitySource",
                "verifiedpermissions:GetPolicy",
                "verifiedpermissions:GetPolicyStore",
                "verifiedpermissions:GetPolicyTemplate",
                "verifiedpermissions:GetSchema",
                "verifiedpermissions:List*",
                "vpc-lattice:GetAccessLogSubscription",
                "vpc-lattice:GetAuthPolicy",
                "vpc-lattice:GetListener",
                "vpc-lattice:GetResourcePolicy",
                "vpc-lattice:GetRule",
                "vpc-lattice:GetService",
                "vpc-lattice:GetServiceNetwork",
                "vpc-lattice:GetServiceNetworkServiceAssociation",
                "vpc-lattice:GetServiceNetworkVpcAssociation",
                "vpc-lattice:GetTargetGroup",
                "vpc-lattice:List*",
                "wafv2:GetIPSet",
                "wafv2:GetLoggingConfiguration",
                "wafv2:GetRegexPatternSet",
                "wafv2:GetRuleGroup",
                "wafv2:GetWebACL",
                "wafv2:GetWebACLForResource",
                "wafv2:List*",
                "workspaces-web:GetBrowserSettings",
                "workspaces-web:GetIdentityProvider",
                "workspaces-web:GetNetworkSettings",
                "workspaces-web:GetPortal",
                "workspaces-web:GetPortalServiceProviderMetadata",
                "workspaces-web:GetTrustStore",
                "workspaces-web:GetUserAccessLoggingSettings",
                "workspaces-web:GetUserSettings",
                "workspaces-web:List*",
                "workspaces:Describe*",
                "xray:BatchGetTraces",
                "xray:GetGroup",
                "xray:GetGroups",
                "xray:GetSamplingRules",
                "xray:GetServiceGraph",
                "xray:GetTraceSummaries",
                "xray:List*"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "AIOPSAPIGatewayAccess",
            "Effect": "Allow",
            "Action": [
                "apigateway:GET"
            ],
            "Resource": [
                "arn:aws:apigateway:*::/restapis",
                "arn:aws:apigateway:*::/restapis/*",
                "arn:aws:apigateway:*::/restapis/*/deployments",
                "arn:aws:apigateway:*::/restapis/*/deployments/*",
                "arn:aws:apigateway:*::/restapis/*/resources/*/methods/*/integrations",
                "arn:aws:apigateway:*::/restapis/*/resources/*/methods/*/integrations/
*",
                "arn:aws:apigateway:*::/restapis/*/stages",
                "arn:aws:apigateway:*::/restapis/*/stages/*",
                "arn:aws:apigateway:*::/apis",
                "arn:aws:apigateway:*::/apis/*",
                "arn:aws:apigateway:*::/apis/*/deployments",
                "arn:aws:apigateway:*::/apis/*/deployments/*",
                "arn:aws:apigateway:*::/apis/*/integrations",
                "arn:aws:apigateway:*::/apis/*/integrations/*",
                "arn:aws:apigateway:*::/apis/*/stages",
                "arn:aws:apigateway:*::/apis/*/stages/*",
                "arn:aws:apigateway:*::/domainnames/*"
            ]
        }
    ]
}
```

限制 AWS 账户中的代理访问权限

AWS DevOps 代理使用 IAM 角色在事件调查和预防性评估期间发现和描述 AWS 资源。您可以通过配置附加到这些角色的 IAM 策略来控制代理的访问级别。应用程序拓扑不会显示代理可以访问的所有内容，IAM 策略是真正限制代理可以访问的 AWS APIs 服务和资源的唯一方法。

了解 AWS DevOps 代理的 IAM 角色

AWS DevOps 代理使用 IAM 角色访问两种账户中的资源：

- 主账户角色-授予代理访问您在其中创建代理空间的 AWS 账户中的资源的访问权限。
- 次要账户角色-授予代理访问您连接到代理空间的其他 AWS 账户中的资源的权限。

无论哪种类型的账户，您都可以限制代理可以访问哪些 AWS 服务，限制对这些服务中特定资源的访问以及控制代理可以在哪些区域中运行。

选择您的资源边界

限制资源访问权限时，您需要为代理提供足够的权限才能成功调查应用程序事件。这包括：

- 代理应监控和调查的范围内应用程序的所有资源
- 这些应用程序所依赖的所有支持基础架构

支持基础设施可能包括：

- 网络组件（子网VPCs、负载均衡器、API 网关）
- 数据存储（数据库、缓存、对象存储）
- 计算资源（EC2 实例、Lambda 函数、容器）
- 监控和记录服务 (CloudWatch, CloudTrail)
- 了解权限所需的身份和访问管理资源

如果您限制访问权限的范围过于狭窄，则代理可能无法确定源于您定义的边界之外的支持基础架构的根本原因。

限制服务访问

您可以通过修改附加到代理角色的 IAM 策略来限制代理可以访问的 AWS 服务。创建自定义策略时，请遵循以下最佳实践：

- 仅授予只读权限-代理需要在调查期间读取资源配置、指标和日志。避免授予允许代理修改或删除资源的权限。
- 仅@@ 限于必要的服务-仅包括包含与您的应用程序相关的资源的 AWS 服务。例如，如果您的应用程序不使用 Amazon RDS，则不要在策略中包含 RDS 权限。
- 使用特定操作代替通配符-与其授予`service:*`权限，不如指定单个操作，例如`cloudwatch:GetMetricData`或`ec2:DescribeInstances`。

限制特定服务的政策示例：

```
json

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:GetMetricData",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:DescribeAlarms",
        "logs:GetLogEvents",
        "logs:FilterLogEvents",
        "ec2:DescribeInstances",
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": "*"
    }
  ]
}
```

限制资源访问权限

要将代理限制为服务中的特定资源，请在 IAM 策略中使用资源级权限。这允许您仅向符合特定模式的资源授予访问权限。

使用资源 ARN 模式：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration"
      ],
      "Resource": "arn:aws:lambda:*:*:function:production-*"
    }
```

```
  ]
}
```

此示例限制代理只能访问名称以 “生产-” 开头的 Lambda 函数。

使用基于标签的限制：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeInstanceStatus"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Environment": "production"
        }
      }
    }
  ]
}
```

此示例限制代理只能访问标记为的 EC2 实例`Environment=production`。

限制区域访问

要限制代理可以访问哪些 AWS 区域，请使用您的 IAM 策略中的`aws:RequestedRegion`条件密钥：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:Describe*",
        "lambda:Get*",
        "cloudwatch:Get*"
      ],
```

```
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": [
            "us-east-1",
            "us-west-2"
          ]
        }
      }
    }
  ]
}
```

此示例限制代理只能访问 us-east-1 和 us-west-2 区域的资源。

创建自定义 IAM 策略

创建代理空间或添加辅助账户时，您可以选择使用策略模板创建自定义 IAM 角色。这使您可以实现最小权限原则。

创建代理空间时

从 AWS 管理控制台中的 DevOps 代理控制台...

- 选择 “使用策略文档创建新的 DevOps 代理角色”，然后按照说明进行操作

编辑代理空间时

从 AWS 管理控制台中的 DevOps 代理控制台...

- 选择 “权能” 选项卡
- 从 “云端” 部分选择要编辑的辅助账户，然后单击 “编辑”
- 选择使用模板创建新的 DevOps 代理策略，然后按照说明进行操作

自定义策略最佳实践

- 仅授予只读权限-避免允许修改或删除资源的权限
- 尽可能使用资源级权限 — 使用 ARN 模式或标签限制对特定资源的访问
- 定期审查和审计权限 — 定期审查代理的 IAM 政策，确保它们仍然符合您的安全要求

设置 IAM 身份中心身份验证

IAM Identity Center 身份验证提供了一种集中方式来管理用户对 A AWS DevOps gent Space Web 应用程序的访问权限。本指南介绍如何配置 IAM Identity Center 身份验证和管理用户。

先决条件

在设置 IAM 身份中心身份验证之前，请确保您具有：

- 您的组织或账户已启用 IAM 身份中心
- AWS DevOps 代理中的管理员权限
- 已配置或准备创建的代理空间

身份验证选项

AWS DevOps 代理提供两种用于访问 Agent Space Web 应用程序的身份验证方法：

IAM 身份中心身份验证-建议在生产环境中使用。提供集中式用户管理、与外部身份提供商的集成以及长达 12 小时的会话。

管理员访问权限（IAM 身份验证）-在初始设置和配置期间为管理员提供快速访问权限。会话限制在 30 分钟以内。

在创建代理空间期间配置 IAM 身份中心

创建代理空间时，您可以在 A c cess 选项卡上配置 IAM 身份中心身份验证：

步骤 1：导航到 Web 应用程序配置

1. 配置您的代理空间详细信息和 AWS 帐户访问权限后，进入访问选项卡
2. 您将看到两个部分：“Connect IAM 身份中心” 和 “管理员访问权限”

步骤 2：配置 IAM 身份中心集成

在 Connect [代理空间] 到 IAM 身份中心部分：

1. 验证 IAM 身份中心实例 — 控制台显示哪个 Identity Center 实例将管理 Web App 用户访问权限（例如`ssoins-7223a9580931edbe`）。系统将自动预填充离您最近的 IAM 身份中心实例。

2. 选择 IAM Identity Center 应用程序角色名称选项 — 选择三个选项之一：

自动创建新的 DevOps 代理角色（推荐）：

- 系统会自动创建具有适当权限的新服务角色
- 这是最简单的选项，适用于大多数用例

分配现有角色：

- 使用您已经创建的现有 IAM 角色
- 系统将验证该角色是否具有所需的权限
- 如果您的组织已为 AWS DevOps 代理预先创建了角色，请选择此选项

使用策略模板创建新的 DevOps 代理角色：

- 使用提供的策略详细信息在 IAM 控制台中创建自己的自定义角色
- 如果您需要自定义角色权限，请选择此选项

单击 Connect 后，系统会自动：

- 创建或配置指定的 IAM 角色
- 为您的代理空间设置 IAM 身份中心应用程序
- 在 IAM 身份中心和 Agent Space Web 应用程序之间建立信任关系
- 配置 OAuth 2.0 身份验证流程，确保用户访问安全

备选方案：使用管理员访问权限

如果您想在不设置 IAM 身份中心的情况下立即访问 Agent Space Web 应用程序，请执行以下操作：

1. 在管理员访问权限部分，记下提供管理员访问权限的 IAM 角色 ARN（例如）`arn:aws:iam::440491339484:role/service-role/DevOpsAgentRole-WebappAdmin-15ppoc42`
2. 点击蓝色的管理员访问权限按钮启动采用 IAM 身份验证的 Agent Space Web 应用程序
3. 使用此方法的会话限制为 30 分钟

> **Note**
>
> 管理员访问权限用于初始设置和配置。对于生产用途和持续操作，请配置 IAM 身份中心身份验证。

添加用户和组。

配置 IAM Identity Center 身份验证后，您需要向特定用户和群组授予对 Agent Space Web 应用程序的访问权限：

步骤 1：访问用户管理

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “访问权限” 选项卡
3. 在 “用户访问权限” 下，单击 “管理用户和群组”

步骤 2：添加用户或群组

1. 选择 “添加用户或群组”
2. 在 IAM 身份中心目录中搜索用户或群组
3. 选中要添加的用户或群组旁边的复选框
4. 单击 “添加” 授予他们访问权限

选定的用户现在可以使用他们的 IAM 身份中心证书访问 Agent Space Web 应用程序。

与外部身份提供商合作

如果你使用外部身份提供商（例如 Okta、Microsoft Entra ID 或 Ping 身份）和 IAM 身份中心：

- 用户和群组从您的外部身份提供商同步到 IAM 身份中心
- 将用户和组添加到 Agent Space Web 应用程序时，即从同步目录中进行选择
- 用户属性和群组成员资格由您的外部身份提供商维护
- 同步后，您的身份提供商的更改会自动反映在 IAM 身份中心中

用户如何访问 Agent Space Web 应用程序

将用户添加到代理空间后：

1. 与授权用户共享 Agent Space Web 应用程序 URL
2. 当用户导航到该 URL 时，他们会被重定向到 IAM 身份中心登录页面
3. 输入他们的凭证（如果已配置，则完成 MFA）后，他们将被重定向回 Agent Space 网络应用程序
4. 默认情况下，他们的会话有效期为 8 小时（可由 Identity Center 管理员配置）

管理用户访问权限

您可以随时更新用户访问权限：

添加更多用户或群组：

- 按照上述相同步骤添加其他用户或群组

正在删除访问权限：

1. 在 “用户访问权限” 部分，找到要移除的用户或组
2. 点击他们名字旁边的移除按钮
3. 确认移除

被移除的用户将立即失去访问权限，但活动会话可能会持续到到期为止。

会话管理

Agent Space Web 应用程序的 IAM 身份中心会话具有以下特征：

- 默认会话时长 — 8 小时
- 会话安全 — 仅限 HTTP 的 Cookie 可增强保护
- 多重身份验证 — 在 IAM 身份中心配置时支持
- API 凭证 — 针对 API 调用颁发短期（15 分钟）Sigv4 凭证并自动续订

要配置会话持续时间：

1. 导航到 IAM 身份中心控制台
2. 前往 “设置” > “身份验证”
3. 在 “会话持续时间” 下，配置您的首选持续时间（从 1 小时到 12 小时）
4. 选择保存更改

断开身份中心的连接

1. 在 Agent Space 的控制台中，点击右上角的操作，然后选择断开与 IAM 身份中心的连接
2. 在确认对话框中确认

设置外部身份提供商 (IdP) 身份验证

外部身份提供商 (IdP) 身份验证允许您的组织使用与 OIDC 兼容的现有身份提供商（例如 Okta 或 Microsoft Entra ID）来管理用户对 Agent Space Web 应用程序的访问权限。 AWS DevOps 用户直接通过您的 IdP 使用其公司证书登录，无需 AWS 使用 IAM 身份中心。

先决条件

在设置外部 IdP 身份验证之前，请确保：

- 兼容 OIDC 的身份提供商（Okta 或 Microsoft Entra ID）
- 管理员对您的身份提供商的访问权限
- 访问 AWS DevOps 代理控制台的管理员权限
- 已配置或准备创建的代理空间

工作原理

配置外部 IdP 身份验证时：

- 用户导航到 Agent Space Web 应用程序 URL
- 它们会被重定向到您的身份提供商的登录页面
- 使用公司凭证进行身份验证后，他们会被重定向回 Web 应用程序
- Web 应用程序将身份验证令牌交换为限于 Agent Space 的短期 AWS 凭证

会话的有效期最长为 8 小时。使用 OIDC 刷新令牌自动刷新凭证，无需用户重新进行身份验证。

配置外部 IdP 身份验证

步骤 1：在您的身份提供商中注册应用程序

选择您的身份提供商，然后按照相应的设置说明进行操作。

选项 A：Okta

1. 在 Okta 管理员控制台中，导航到 “应用程序” > “应用程序”，然后选择 “创建应用程序集成”
2. 选择 OIDC-OpenID Connect 作为登录方法，选择 Web 应用程序作为应用程序类型。选择下一步。
3. 为应用程序设置描述性名称（例如，`AWS DevOps Agent`）
4. 在 “拨款类型” 下，确保选中以下各项：
   - 授权码（默认）
   - 刷新令牌-这是会话刷新所必需的。如果未启用，用户将无法维护会话。

> **Note**
>
> 默认情况下，Okta 不启用刷新令牌授权类型。您必须明确启用它。

1. 暂时将登录重定向 URIs保留为默认值 — 您将在配置代理空间后对其进行更新
2. 在 “分配” 下，分配应具有访问权限的用户或群组
3. 选择保存
4. 在应用程序的 “常规” 选项卡上，记下以下值：
   - 客户端 ID
   - 客户机密钥-选择 “复制” 以安全地保存此值
5. 记下你的 Okta 域名 ——这是你的发行商网址（例如`https://dev-12345678.okta.com`）。

> **Note**
>
> 在 “登录” 选项卡上，确认颁发者已设置为 Okta URL（不是动态）。这样可以确保发行者 URL 稳定。

> **Note**
>
> 请勿在授权服务器的 “声明” 选项卡中的 ID 令牌中添加群组声明。 AWS DevOps 代理不使用您的 IdP 的群组成员资格。

选项 B：微软 Entra ID

1. 在 Azure 门户中，导航到微软 Entra ID > 应用程序注册 > 新注册
2. 设置描述性名称（例如，`AWS DevOps Agent`）
3. 在 “支持的账户类型” 下，选择适合您的组织的选项（通常仅限此组织目录中的帐户）
4. 暂时将重定向 URI 留空。选择 “注册”
5. 在应用程序概述页面上，记下以下值：
   - 应用程序（客户端）ID — 在配置代理空间时用作客户端 ID
   - 目录（租户）ID — 用于构造发行者 URL
6. 导航到 “证书和密钥” > “新建客户机密”
   - 设置描述和到期时间
   - 选择 “添加” 并立即复制密钥值 — 它不会再次显示
7. Entra ID 的发行者网址遵循此格式。`{tenant-id}`替换为步骤 5 中的目录（租户）ID：
   - `https://login.microsoftonline.com/{tenant-id}/v2.0`

> **Note**
>
> 请勿在令牌配置中启用群组的可选声明。 AWS DevOps 代理不使用您的 IdP 的群组成员资格。

步骤 2：启用带有 IdP 身份验证的操作员应用程序

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “访问权限” 选项卡
3. 在 “用户访问权限” 下，选择 “外部身份提供商”
4. 在配置表单中，配置以下内容：
   - 身份提供商-选择您的身份提供商（Okta 或 Microsoft Entra ID）

- 颁发者网址-您的身份提供商提供的 OIDC 颁发者网址
- 客户端 ID — 来自您创建的 OIDC 应用程序的客户端 ID
- 客户端密钥 — 来自 OIDC 应用程序的客户端密钥

5. 在 “身份提供商应用程序角色名称” 下，选择以下三个选项之一：
   - 自动创建新的 DevOps 代理角色（推荐）-创建具有适当权限的新服务角色
   - 分配现有角色-使用您已经创建的现有 IAM 角色
   - 使用策略模板创建新的 DevOps 代理角色 — 使用提供的详细信息在 IAM 控制台中创建自己的角色
6. 查看表单底部显示的回调 URL 警告提醒。复制此 URL — 您需要将其添加到身份提供商允许的重定向中， URIs 然后用户才能登录。
7. 选择 Connect (连接)。

选择 Connect 后，控制台将显示包含以下详细信息的外部身份提供者配置：

- 提供商-您选择的身份提供商
- 颁发者网址-已配置的 OIDC 颁发者网址
- 客户端 ID-配置的客户端 ID
- IAM 角色 ARN — 用于用户访问的 IAM 角色
- 回调 URL — 在您的身份提供商中将此 URL 配置为允许的重定向 URI
- 登录 URL — 使用此 URL 通过您的身份提供商访问 Web 应用程序

第 3 步：将回传 URL 添加到您的身份提供商

Okta

1. 在 Okta 管理员控制台中，导航到应用程序的 “常规” 选项卡
2. 在 “登录” 下，选择 “编辑”
3. 将回调 URL 添加为登录重定向 URI：
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/callback`
4. （可选）设置启动登录 URI 以启用 IdP 从 Okta 控制面板启动的登录：
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/login`
5. （推荐）添加注销重定向 URI，以便在注销后将用户重定向回 Web 应用程序。否则，用户在注销时可能会看到错误页面：

- `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`

6. 选择保存

Microsoft Entra ID

1. 在 Azure 门户中，导航到应用程序的身份验证页面
2. 在平台配置下，选择添加平台 > Web
3. 输入回传 URL 作为重定向 URI：
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/callback`
4. （可选）添加注销重定向 URI，以便在注销后将用户重定向回 Web 应用程序：
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`
5. 选择 “配置”

步骤 4：验证配置

1. 导航到控制台中显示的登录 URL：
   - `https://{agentSpaceId}.aidevops.global.app.aws/authorizer/idp/login`
2. 您应该被重定向到身份提供商的登录页面
3. 使用您的公司凭证登录
4. 成功进行身份验证后，您将被重定向回 Agent Space Web 应用程序

更新 IdP 配置

你可以在不断开连接的情况下轮换客户端密钥：

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “访问权限” 选项卡
3. 在 “外部身份提供商配置” 下，选择 “轮换客户端密钥”
4. 输入新的客户密钥
5. 选择保存

要更改任何其他 IdP 配置字段（例如颁发者 URL、客户端 ID 或身份提供者），必须断开现有 IdP 的连接并配置一个新的 IdP。

用户如何访问 Agent Space Web 应用程序

配置外部 IdP 身份验证后：

- 与授权用户共享 Agent Space Web 应用程序 URL
- 当用户导航到该 URL 时，他们会被重定向到您的身份提供商的登录页面
- 输入他们的凭证（如果由你的 IdP 配置，则完成 MFA）后，他们将被重定向回 Agent Space 网络应用程序
- 会话会自动刷新 — 有关详细信息，请参阅会话管理

会话管理

Agent Space Web 应用程序的外部 IdP 会话具有以下特征：

- 会话时长-浏览器会话持续长达 8 小时。这在 AWS DevOps 代理中不可配置。如果您的 IdP 会话生命周期超过 8 小时，则可以在用户下次访问时自动重新进行身份验证，而无需输入凭据。根据组织的安全要求配置 IdP 的会话和令牌生命周期。
- 凭据刷新 — 使用 OIDC 刷新令牌自动刷新会话，无需用户重新进行身份验证
- 多重身份验证-在您的身份提供商中配置时支持。IdP 在登录期间处理 MFA — 无需在代理中进行其他配置 AWS DevOps

注销行为

当用户在 Web 应用程序中单击 “注销” 时：

1. 所有会话 cookie 都会立即被清除
2. 用户被重定向到身份提供商的 OIDC 注销端点以终止 SSO 会话
3. 如果配置了注销重定向 URI，则用户将被重定向回 Web 应用程序欢迎页面

撤消用户访问权限

要立即撤消用户的访问权限，您可以直接在身份提供商的管理门户中撤消他们的会话：

- Okta — 在 Okta 管理员控制台中，导航到 “目录” > “人员”，选择用户，选择 “更多操作” > “清除用户会话”
- Microsoft Entra ID — 在 Azure 门户中，导航到 “用户”，选择用户，然后选择 “撤消会话”

安全注意事项

客户端密钥存储 — 如果您在创建代理空间时提供了客户管理的 KMS 密钥，则使用客户管理的 KMS 密钥进行加密，否则使用服务拥有的密钥进行加密。初始配置后，它永远不会在 API 响应中返回，也不会显示在控制台中。

客户端密钥轮换 — Entra 客户端密钥具有可配置的过期时间。使用 AWS DevOps 代理控制台中的 “轮换客户端密钥” 选项设置提醒，以便在密钥到期之前轮换密钥。如果密钥过期，则在轮换密钥之前，用户将无法登录。

令牌生命周期管理 — 身份提供商颁发的令牌（访问令牌、刷新令牌）的生命周期由您的 IdP 的配置控制。我们建议在您的 IdP 中配置适当的令牌生命周期：

- Okta — 在 “安全” > “API” > “授权服务器” > “访问策略” 下配置令牌生命周期
- Microsoft Entra ID — 使用令牌生命周期策略配置令牌生命周期

群组声明 — 请勿在身份提供商的令牌配置中启用群组声明。 AWS DevOps 代理目前不使用您的 IdP 的群组成员资格。

用户标识符- AWS DevOps 代理使用特定于提供商的声明来唯一识别用户：

- Okta — 使用 ID 令牌中的`sub`声明
- Microsoft Entra ID — 使用 ID 令牌中的`oid`（对象标识符）声明

这些标识符是不可变的，会出现在 CloudTrail 日志中以供审计。

断开外部 IdP 的连接

1. 在 AWS DevOps 代理控制台中，选择您的代理空间
2. 前往 “访问权限” 选项卡
3. 在 “用户访问权限” 下，选择断开连接
4. 查看确认对话框中列出的影响并确认

断开连接将：

- 从代理空间中删除 IdP 配置

- 阻止用户通过外部身份提供商登录
- 移除与 IdP 用户帐户关联的个人聊天和构件历史记录

活跃的用户会话将持续到其过期或下一次凭据刷新失败。

问题排查

- 重定向到 IdP 失败 — 验证颁发者网址是否与您的 IdP 的 OIDC 发现端点相匹配。对于 Okta，请确保在 “登录” 选项卡上将发行者设置为 Okta URL（不是动态）。对于 Entra，请使用格式`https://login.microsoftonline.com/{tenant-id}/v2.0`。
- 访问被拒绝或策略错误 (Okta)-在 “分配” 下验证用户或其组已分配给应用程序。选中 “登录” > “登录政策” 规则。
- 登录后出现 IdP 配置错误-您的身份提供商未返回刷新令牌。确保已启用`offline_access`范围和刷新令牌授予类型：
  - Okta — 前往应用程序的 “常规” 选项卡并启用 “授权类型” 下的 “刷新令牌” 复选框
  - Entra — 前往 API 权限并确保列`offline_access`在委托权限下
- 身份验证成功但网络应用显示错误 — 验证您的 IdP 中的重定向 URI 与代理控制台中 AWS DevOps 显示的回调 URL 完全匹配。
- 身份验证失败-如果您的 IdP 中启用了群组可选声明，请将其禁用。 AWS DevOps 代理不使用团体索赔。
- IdP 身份验证后登录失败 — 对于 Entra，应用程序清单**`null`**中未将 ver **`requestedAccessTokenVersion`** ify 设置为。对于 Okta，请验证发行者网址是否正确。
- 单击 “注销” 后出现错误页面 (Okta) — 如果您在注销后看到**`post_logout_redirect_uri`**错误，请在 Okta 应用程序的 “常规” 选项卡中添加**`https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`**为注销重定向 URI。
- 用户在注销后会留在身份提供者页面上 (Entra) — 要在注销后将用户重定向回 Web 应用程序，请在您的 Entra 应用程序的身份验证页面中添加**`https://{agentSpaceId}.aidevops.global.app.aws/authorizer/welcome`**为重定向 URI。

AWS DevOps 代理的静态加密

AWS DevOps 代理对所有静态客户数据进行加密。默认情况下， AWS DevOps 代理使用 AWS 自有密钥自动加密您的数据，无需额外付费。您无法查看、管理或审核 AWS 自有密钥的使用情况。但是，您无需采取任何措施来保护这些密钥。您的数据将自动受到保护。

您可以选择使用您在密钥管理服务 (AWS KMS) 中创建、拥有和管理的对称客户托管 AWS 密钥来加密数据。由于您可以完全控制此加密层，因此可以执行以下任务：

- 制定和维护关键策略
- 启用和禁用密钥策略
- 轮换加密材料
- 添加 标签
- 创建密钥别名
- 安排密钥删除

有关更多信息，请参阅《密钥管理服务开发人员指南》中的客户托管AWS密钥。

> **Note**
>
> AWS DevOps 代理使用 AWS 自有密钥自动启用静态加密，从而免费保护客户数据。当您使用客户托管密钥时，将收取标准 AWS KMS 费用。有关定价的更多信息，请参阅AWS 密钥管理服务定价。

客户自主管理型密钥

客户托管密钥是您在 AWS 账户中创建、拥有和管理的 KMS 密钥。您可以完全控制这些 KMS 密钥，包括建立和维护其密钥策略。

当您配置客户托管密钥时， AWS DevOps 代理会使用它来保护敏感的资源数据。 AWS DevOps 代理使用带有加密 SDK 分层密钥环的信封 AWS 加密。您的 KMS 密钥用于生成分支密钥，从而保护您的数据。

在创建以下资源时，您可以指定客户托管密钥：

- Agent Sp ace — 加密代理空间详细信息和通过 DevOps 代理 Web 应用程序创建的与调查、技能和聊天相关的内容。
- 服务-加密静态的第三方服务凭证。

要在 AWS DevOps 代理中配置客户托管密钥，请按照以下步骤操作。

步骤 1：创建客户托管式密钥

您可以使用 AWS KMS 控制台或 KMS API 创建对称客户托管 AWS 密钥。密钥必须满足以下要求：

属性	要求
密钥类型	对称
密钥规范	SYMMETRIC_DEFAULT
密钥用法	ENCRYPT_DECRYPT

> **Note**
>
> AWS DevOps 代理仅支持带有密钥规格和密钥用法的SYMMETRIC_DEFAULT对称加密 KMS ENCRYPT_DECRYPT 密钥。目前不支持多区域密钥和非对称密钥。

有关更多信息，请参阅《密钥管理服务开发人员指南》中的创建对称客户托管AWS密钥。

步骤 2：设置密钥策略

密钥政策控制对客户托管密钥的访问。每个客户托管式密钥必须只有一个密钥策略，其中包含确定谁可以使用密钥以及如何使用密钥的声明。

您的密钥策略必须向调用委托人（您的 IAM 身份）和 AWS DevOps 代理服务授予权限。 AWS DevOps 代理使用两组凭据访问您的密钥：

1. 您的呼叫者凭证 — 用于所有同步操作，包括密钥验证、资源创建时的加密以及向调用者返回直接响应的任何 API 调用。
2. AWS DevOps 代理服务主体 — 用于在后台运行的异步操作，例如运营调查、事件分析、事件关联和根本原因分析生成。

下表列出了所需的 KMS 操作：

KMS 操作	说明
kms:DescribeKey	在创建资源时验证密钥配置

KMS 操作	说明
kms:GenerateDataKey	生成用于信封加密的数据加密密钥
kms:Decrypt	解密数据
kms:Encrypt	加密数据
kms:ReEncrypt	使用相同或不同的密钥重新加密数据

AWS DevOps 代理在配置时使用试运行操作验证所有这些权限。如果缺少任何权限，则请求会失败，但会出现异常。

以下是示例密钥策略。用自己的占位符值替换占位符值。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCallerAccessViaService",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/DevOpsAgentUserRole"
      },
      "Action": [
        "kms:DescribeKey",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "aidevops.us-east-1.amazonaws.com"
        }
      }
    },
    {
      "Sid": "AllowDevOpsAgentServiceDescribeKeyAccess",
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowDevOpsAgentAccessForAgentSpace",
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:aidevops:us-east-1:111122223333:agentspace/*"
        },
        "StringLike": {
          "kms:EncryptionContext:aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:us-
east-1:111122223333:agentspace/*"
        }
      }
    },
    {
      "Sid": "AllowDevOpsAgentAccessForService",
      "Effect": "Allow",
      "Principal": {
        "Service": "aidevops.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "ArnLike": {
          "aws:SourceArn": "arn:aws:aidevops:us-east-1:111122223333:service/*"
        },
        "StringLike": {
          "kms:EncryptionContext:aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:us-
east-1:111122223333:service/*"
        }
      }
    }
  ]
}
```

该政策包含以下声明：

- AllowKeyAdministration— 授予账户 root 对密钥的完全管理权限。111122223333替换为您的 AWS 账户 ID。
- AllowCallerAccessViaService— 向您的 IAM 委托人授予所有同步 AWS DevOps 代理操作所需的 KMS 权限。这包括在资源创建时进行密钥验证，以及对向调用方返回直接响应的任何 API 调用的加密和解密操作。该kms:ViaService条件可确保您只能通过 AWS DevOps 代理服务使用密钥。111122223333替换为您的 AWS 账户 ID 和us-east-1您所在 AWS 的地区。
- AllowDevOpsAgentServiceAccessForAgentSpace/AllowDevOpsAgentServiceAccessForService— 向aidevops.amazonaws.com服务主体授予异步操作所需的 KMS 权限。 AWS DevOps 代理在执行后台操作（例如运营调查、分析事件、跨服务关联事件以及生成根本原因分析）时，使用此服务主体来加密和解密您的数据。如果没有此访问权限， AWS DevOps 代理将无法读取代表您进行调查所需的加密数据。该aws:SourceArn条件限制了对来自您的 AWS DevOps 代理资源的请求的访问权限，并且该kms:EncryptionContext条件可确保加密上下文与您的资源 ARNs相匹配。111122223333替换为您的 AWS 账户 ID 和us-east-1您所在 AWS 的地区。

有关密钥策略的更多信息，请参阅《密钥管理服务开发人员指南》中的 AWS KMS 中的AWS 密钥策略。

步骤 3：在创建资源时指定密钥

创建密钥并配置密钥策略后，可以在创建 AWS DevOps 代理资源时指定密钥。

控制台

要在控制台中创建代理空间时配置客户托管密钥，请执行以下操作：

1. 打开 AWS DevOps 代理控制台。

2. 选择 “创建代理空间” 或 “注册服务”。
3. 输入代理空间的详细信息（名称、描述和 IAM 角色）。
4. 展开 “高级配置” 部分。
5. 在加密密钥类型下，选择客户管理的密钥。
6. 从下拉列表中选择 KMS 密钥，或输入 KMS 密钥 ARN。
7. 查看密钥策略可扩展部分中显示的密钥策略。确保您已将此策略附加到您的 KMS 密钥。您可以使用复制按钮来复制策略。
8. 完成剩余的配置并选择创建。

> **Note**
>
> 如果您在下拉列表中看不到您的 KMS 密钥，请验证该密钥是否符合步骤 1 中的要求以及您是否拥有`kms:ListKeys`和`kms:DescribeKey`权限。

API

使用客户管理的密钥创建座席空间

创建代理空间时指定`kmsKeyArn`参数。该值必须是完整的 KMS 密钥 ARN。

```
{
  "name": "my-agent-space",
  "description": "An encrypted agent space",
  "kmsKeyArn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

使用客户管理的密钥注册服务

注册服务时指定`kmsKeyArn`参数。该值必须是完整的 KMS 密钥 ARN。所有服务类型都支持此参数，包括 Dynatrace、、 ServiceNow、 PagerDuty GitLab GitHub、和 MCP 服务器。

```
{
  "service": "dynatrace",
  "kmsKeyArn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "serviceDetails": { ... }
```

```
}
```

> **Note**
>
> 您必须在创建资源时指定客户管理的密钥。您不能为现有资源添加或更改客户托管密钥。

AWS DevOps 代理加密上下文

加密上下文是一组非秘密密钥值对，其中包含有关数据的其他上下文信息。 AWS KMS 使用加密上下文作为额外的经过身份验证的数据来支持经过身份验证的加密。当您在加密数据的请求中包含加密上下文时， AWS KMS 会将加密上下文绑定到加密数据。要解密数据，您必须在请求中包含相同的加密上下文。

AWS DevOps 代理在所有加密操作中使用以下加密上下文：

```
{
  "aws-crypto-ec:aws:aidevops:arn": "arn:aws:aidevops:{region}:{accountId}:
{resourceType}/{resourceId}"
}
```

加密上下文值是正在加密的 AWS DevOps 代理资源的 ARN。您可以在密钥策略条件和 AWS CloudTrail 日志中使用此加密上下文来审核密钥的使用情况。

密钥管理

如果您禁用或计划删除您的 KMS 密钥， AWS DevOps 代理将无法解密您的数据。这会导致读取加密数据的操作出`AccessDeniedException`错。

> **Important**
>
> 如果您选择使用客户管理的密钥，则应负责管理该密钥及其权限。如果密钥被禁用或删除，或者 AWS DevOps 代理失去使用密钥的权限，则您将无法访问加密数据。

下表描述了常见的故障场景：

Action	影响
密钥策略权限已撤销	`AccessDeniedException` 关于加密和解密操作
KMS 密钥已禁用	`DisabledException` 关于加密和解密操作
KMS 密钥已计划删除	`KMSInvalidStateException` 关于加密和解密操作
KMS 密钥已删除	永久丢失数据-无法恢复加密数据

在禁用或删除密钥之前：

1. 确认密钥不依赖任何活跃的 AWS DevOps 代理资源。
2. 考虑先禁用密钥以测试其影响，然后再安排删除时间。
3. AWS KMS 强制执行删除密钥前的最短等待时间，让您在需要时有时间取消。

> 注意：： AWS DevOps 代理不会自动重新加密使用新密钥的数据。如果您需要轮换到新的客户托管密钥，则必须使用新密钥创建新资源。

监控您的加密密钥

当您将客户托管密钥与 AWS DevOps 代理一起使用时，您可以使用AWS CloudTrail来跟踪 AWS DevOps 代理发送到 AWS KMS 的请求。

您可以按以下方式筛选 CloudTrail 事件：

- 事件源 — `kms.amazonaws.com`
- 加密上下文密钥 — `aws-crypto-ec:aws:aidevops:arn`
- 密钥 ARN — 您的客户在请求参数中管理密钥 ARN

有关更多信息，请参阅AWS 密AWS 钥管理服务开发人员指南 AWS CloudTrail中的使用记录 KMS API 调用。

VPC 终端节点 (AWS PrivateLink)

您可以使用 AWS PrivateLink 在您的 VPC 和 AWS DevOps 代理之间创建私有连接。您可以像访问您的 VPC 一样访问 AWS DevOps 代理，无需使用互联网网关、NAT 设备、VPN 连接或 Direct Connect 连接。您的 VPC 中的实例不需要公有 IP 地址即可访问 AWS DevOps 代理。

您可以通过创建由提供支持的接口端点来建立此私有连接 AWS PrivateLink。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。这些是请求者管理的网络接口，用作发往代理的流量的入口点。 AWS DevOps

有关更多信息，请参阅AWS PrivateLink 指南 AWS PrivateLink中的通过访问 AWS 服务。

AWS DevOps 代理 VPC 终端节点的注意事项

在为 AWS DevOps 代理设置接口端点之前，请查看AWS PrivateLink 指南中的注意事项。

AWS DevOps 代理支持通过以下 VPC 终端节点进行 API 调用。

类别	端点后缀
AWS DevOps 代理控制平面 API 操作	`aidevops`
AWS DevOps 代理运行时操作	`aidevops-dataplane`
AWS DevOps 代理 Webhook 事件	`event-ai`

为 AWS DevOps 代理创建接口终端节点

您可以使用 Amazon VPC 控制台或 AWS 命令行界面 (AWS CLI) 为 AWS DevOps 代理创建接口终端节点。有关更多信息，请参阅《AWS PrivateLink 指南》中的创建接口端点。

使用以下服务名称为 AWS DevOps Agent 创建接口终端节点：

- com.amazonaws。 {region} .aidevops
- com.amazonaws。 {region} .aidevops-dataplane
- com.amazonaws。 {region} .event-ai

在创建端点后，您可以选择启用私有 DNS 主机名。在创建 VPC 端点时，通过在 VPC 控制台中选择启用私有 DNS 名称，可启用此设置。

如果您为接口终端节点启用私有 DNS，则可以使用代理的默认区域 DNS 名称向 AWS DevOps 代理发出 API 请求。以下示例显示了默认区域 DNS 名称的格式。

- aidevops。 {region} .api.aws
- aidevops-dataplane。 {区域} .amazonaws.com
- event-ai。 {region} .api.aws

为接口端点创建端点策略

端点策略是一种 IAM 资源，您可以将其附加到接口端点。默认端点策略允许通过接口端点对 AWS DevOps 代理进行完全访问权限。要控制允许 AWS DevOps 代理从您的 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：

- 可以执行操作的委托人（AWS 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《AWS PrivateLink 指南》中的使用端点策略控制对服务的访问权限。

配额

AWS DevOps 代理配额包括代理空间数量、并行调查等。您可以请求增大某些配额，但并非所有配额都可以增大。这些加薪不会立即获得批准，因此您的上调可能需要几个小时到几天才能生效。除非另有说明，否则每个配额都是特定于区域的。

下表描述了 AWS DevOps 代理的配额。

Name	默认值	可调整	说明
每个区域每个账户的代理空间	10	是	您可以在每个 AWS 区域中为每个账户创建的最大代理空间数量。
每个代理空间的并行调查	3	是	在单个座席空间中可以同时运行的最大事件解决调查数量。
每个代理空间的并行评估	1	否	在单个座席空间中可以同时运行的最大事件预防评估次数。
每个代理空间的并发按需调用	10	是	在单个代理空间中可以同时运行的按需 DevOps 调用的最大数量。

请求提高配额

您可以使用以下选项之一申请增加配额：

- 从 AWS 管理控制台 — 打开 S ervice Quotas 控制台。在导航窗格中，选择 AWS 服务。选择DevOps 代理，选择配额，然后按照说明申请增加配额。有关更多信息，请参阅《服务配额用户指南》中的 Requesting a quota increase。
- 从 AWS CLI 中 — 使用 C request-service-quota-increase AWS LI 命令。有关更多信息，请参阅《服务配额用户指南》中的 Requesting a quota increase。

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。